

Proceedings of the Austrian Robotics Workshop 2025

FH Salzburg
Campus Urstein
Urstein Süd 1, 5412 Puch

Wilfried Kubinger, Simon Kranzer and Markus Vincze (Eds.)

Puch, May 2025

Preface

The **Austrian Robotics Workshop (ARW)** is Austria's central annual platform for exchanging ideas and presenting innovations in the field of robotics. Since its beginnings in 2006, ARW has evolved into a vibrant event that connects researchers, students, professionals, and industry representatives from across Austria and beyond. It fosters collaboration, sparks new ideas, and showcases the latest developments in robotic systems, intelligent automation, and human-machine interaction.

The **2025 edition of ARW** is hosted by the **Department of Information Technologies and Digitalisation at Salzburg University of Applied Sciences**. Nested in a vibrant region of innovative robotics and automation industry, this year's workshop brings together a broad and dynamic community to discuss current research, share experiences, and explore future trends.

This year's **call for papers** covered a wide range of topics in robotics and automation, including:

- **Mechatronic Design, Kinematics & Dynamics, and Embedded Systems**
- **Control and Machine Learning, Planning, Reasoning & AI**
- **Robot Perception, Computer Vision, Navigation & Manipulation**
- **Cybersecurity, Safety & Resilience, and Systems Engineering**
- **Human-Robot Interaction, Collaborative Robotics, Ethics & Sustainability**
- **Field Robotics, UAVs, UGVs, and Agricultural/Rescue Robots**
- **and novel Robotic Applications in Real-World Environments**

The accepted contributions reflect this diversity, ranging from Sim2Real transfer for grasp verification and low-cost real-time communication, to drone-based perception in agriculture, autonomous navigation, and inclusive workstations for industrial collaboration. A dedicated student session provides young researchers with a platform to present early ideas and engage in interdisciplinary discussions with peers and experts.

The **Austrian Robotics Workshop** is organized under the auspices of **GMAR** – Gesellschaft für Mess-, Automatisierungs- und Robotertechnik – and is supported by the **Austrian Electrotechnical Association (OVE)** and the **Austrian Research Promotion Agency (FFG)**. Their contributions and continuous support are instrumental in ensuring the ongoing success and development of ARW.

We sincerely thank all authors, reviewers, speakers, and attendees for their valuable input and engagement. Our special thanks go to the local organizing committee at FH Salzburg and to all partners who have helped shape this year's event.

We wish all participants an inspiring workshop, meaningful discussions, and a wonderful stay in Salzburg!

Simon Kranzer, Stefan Huber, Simon Hoher, Dorian Prill, Matthäus Horn and Hanna Trenkler (on behalf of the organizing committee)

Puch, May 2025

Program Committee

- Aburaia Mohamed, FH Technikum Wien
- Bader Markus, TU Wien
- Böhm Christoph, FH Technikum Wien
- Brandstötter Mathias, FH Kärnten
- Fischer Clara, JR Robotics
- Hoher Simon, FH Salzburg
- Horauer Martin, FH Technikum Wien
- Huber Stefan, FH Salzburg
- Kubinger Wilfried, FH Technikum Wien
- Müller Andreas, JKU Linz
- Neureiter Christian, FH Salzburg
- Piater Justus, Uni Innsbruck
- Rathmair Michael, JR Robotics
- Rehrl Jakob, FH Salzburg
- Silberbauer Lukas, taurob
- Steinbauer-Wagner Gerald, TU Graz
- Thalhammer Stefan, FH Technikum Wien
- Vincze Markus, TU Wien
- Waclawek Hannes, FH Salzburg
- Werth Wolfgang, FH Kärnten
- Wöber Wilfried, FH Technikum Wien
- Zangl Hubert, AAU Klagenfurt

Keynote Speakers

- Prof. Dr.-Ing. Dirk Jacob, Vice-President Teaching and Professional Development, University of Applied Sciences Kempten: Humanoid robots - the new hype in robotics
- Assistant Prof. Dr.in phil. Mag.a phil. Astrid Weiss, TU Wien: Exploring Robots Through a Human-Centered Lens – Research Challenges, Design Issues, and Innovative Approaches

Contents

1	Update of the SCARAB robot to sort valuable items in containers of residual waste by Daniel Reischl et al.	7
2	Mechanical Design Optimization of a Pneumatically Actuated Parallel Kinematic Machine by Klemens Springer et al.	13
3	Analysis and tuning of PID controller gains for DC servo drives using Garpinger's trade-off plots by Simon Hoher et al.	19
4	Towards Automated Handling and Sorting of Garments combining Visual Language Models and Convolutional Neural Networks by Serkan Ergun et al.	25
5	Towards Inclusive and Accessible Industrial Workstations by Shaping Safe and Adaptive Human-Robot Collaboration by Mara Vukadinovic et al.	31
6	An Adaptable Multi-Robot Support System for Disaster Response by Laurent Frering et al.	37
7	DOPS: Drone Optimized Performance Score for Evaluating Real-Time Tomato Ripeness Detection by Ylli Rexhaj et al.	43
8	Sim2Real Transfer for Vision-Based Grasp Verification by Pau Amargant Alvarez et al.	49
9	LLM-Empowered Embodied Agent for Memory-Augmented Task Planning in Household Robotics by Marc Glocker et al.	55
10	Low-Cost Open-Source Real-Time Communication in Industrial IoT: Using the Raspberry Pi 5 with OPC UA over TSN by Jonathan Mandl et al.	61
11	Multi-Modal 3D Mesh Reconstruction from Images and Text by Tessa Pulli et al.	67
12	Sensorized Adaptive Grasping: ROS2 Based Integration of UR3e and Schunk SVH with Force Sensors by Youssef Aboud et al.	73
13	Category-Level and Open-Set Object Pose Estimation for Robotics by Peter Hönig et al.	79
14	Simulation-Driven Optimization of Stanley Controller Gains for Enhanced Tracking in Autonomous Navigation Robots by Hector Villeda et al.	85
15	Investigating 2.5D path-planning methods for autonomous mobile robots in complex unstructured off-road scenarios by Andre Koczka et al.	91
16	LiDAR-Based Ground Segmentation with Structured Point Clouds for Multi-Sensor AMRs by Hamid Didari et al.	97
17	Multi-Waypoint Path Planning and Motion Control for Non-holonomic Mobile Robots in Agricultural Applications by Mahmoud Ghorab et al.	103
18	Comparison of neural networks road detection in off-road environments by Jakob Oberpertinger et al.	109
19	Multi Robot Route Planning for ROS2 by Matthias Reicher et al.	115

20 ROS with LEGO Spike by Daniel Marth etal.	117
21 Elastic Structure Preserving Control for a Structurally Elastic Robot by Alexander Kitzinger etal.	119
22 A Modular and Configurable Architecture for ROS2 Hardware Integration with micro-ROS by Jakob Friedl etal.	121
23 Automating 3D printing for mass production by Felix Daunert etal.	123
24 A Cost-Effective Testbed for Measuring the Performance of Reference Switches by Tobias Hofer etal.	125
25 A Trajectory Consistency Metric for GNSS Anomaly Detection with LiDAR Odometry by Hans-Peter Wipfler etal.	127

Update of the SCARAB robot to sort valuable items in containers of residual waste

Daniel Reischl¹, Johannes Wenninger, Simon Zwirtmayr and Johannes Schröck

Abstract—In this paper the features of the autonomous mobile robot SCARAB are extended. SCARAB is now not only exchanging full waste containers with empty ones but also sorting out the valuable objects of the waste. For this task, a gripper was added to the robot's end-of-arm tool. The fingers of the gripper have a Fin Ray design to robustly grasp the objects. Adaptions of the waste container allow to empty the waste onto a sorting table without additional actuators. Object detection is done with a YOLOv8 model which was initially trained with an open data set and improved with additional training data. In order to label this training data a standalone tool based on the Segment Anything Model (SAM) was developed. The paper shows the mechanical design of the gripper fingers, the adaption of the waste container as well as the design of a suitable sorting table. It is demonstrated that the waste sorting task is carried out robustly without the need of any additional expensive equipment.

Index Terms—object detection, segmentation, waste sorting

I. INTRODUCTION

Automated image-based recognition and sorting of waste using robots is already being used commercially worldwide. Companies such as ZenRobotics, WasteRobotics, AMP Robotics, Recycleye, Machinex, Bollegraaf, Green Machine and many others offer solutions for efficient sorting on a conveyor belt. However, efficient object recognition is also still a topic of research [7].

This paper, however, is not about a highly efficient implementation of a waste sorting system with expensive cameras and fast delta robots. Our focus is on the subsequent and cost-effective retrofitting of an existing robot, which is used already to autonomously exchange full waste containers for empty ones.

The development platform SCARAB [10] was able to collect full waste containers on demand autonomously and bring them back to a garage. With this setup however it was not possible to sort the waste and all the waste was treated as "residual waste". In January 2025 a deposit on non-returnable containers was put into force in Austria [1], which changed the requirements for the SCARAB platform. As minimum requirement, at least the containers (bottles, cans, etc.) which are subject of the deposit have to be identified and separated of the waste automatically. In this paper the challenges of adapting an existing mobile robot to this new task are described as well as the technical solutions applied for a successful implementation.

¹All authors are with Linz Center of Mechatronics GmbH, Altenberger Straße 69, 4040 Linz, Austria daniel.reischl@lcm.at



Fig. 1. SCARAB during operation while changing the container.

II. SCARAB DEVELOPMENT PLATFORM

The mobile robot SCARAB shown in Fig. 1 was designed to drive autonomously in a semi-public area and exchange the full waste containers. As presented in [10], a sensor in the waste container reports the filling height and a mission to exchange the container is initiated, if the boundary conditions (e.g. weather) are fulfilled. The entire process is not time-critical and the main focus is on personal safety. The new task of sorting waste is therefore carried out in a locked garage to which no passers-by have access. The garage door is controlled automatically via the higher-level mission control system.

After returning back to the garage, SCARAB is now driving to a sorting table. The full waste container is emptied onto the sorting table with the robot arm. No additional actuators or sensors are necessary for the robotic arm or the waste container as shown in section III in more detail. The pile of waste on the sorting table is slightly distributed by a statically programmed movement of the robot arm to facilitate object recognition. A picture of the waste is taken with the wrist camera of the robotic arm. Based on this picture, the valuable items in the waste are detected, as shown in section IV. The recognized objects are sorted out of the waste one by one and separated in the appropriate containers. The waste remaining after the sorting process can then be fed into an appropriate residual waste container by tilting the sorting table. Once the sorting process is complete, SCARAB picks up the empty waste bin and moves to the charging

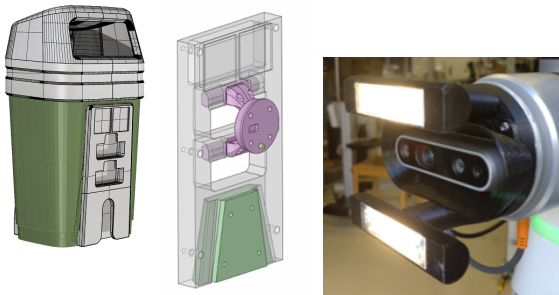


Fig. 2. Already existing passive end of arm tool with Realsense camera and LED lights to manipulate the containers.

station to wait for its next mission.

III. MECHANICAL ADAPPTIONS

The design of SCARAB should not be changed, but additional features are necessary to perform the sorting process. In order to solve this challenge, mechanical adaptations of the waste containers were necessary as well as to add a sorting table and a gripper.

A. Container

The waste bin has a lid with an integrated fill level sensor. This configuration with lid and the robot end effector, which picks up the waste bin via a form-fit connection shown in Fig. 2, do not allow the bin to be emptied by turning it over.

A mechanism has therefore been developed that allows the base of the container to be opened. This mechanism opens the base when the container is pressed against the rear wall of the sorting table with the robot arm, shown in Fig. 3 and Fig. 4. After the contents of the container have fallen out, the bottom of the container is closed again with a suitable trajectory. Both processes, opening and closing, are carried out without additional actuators but solely by pressing the container against the sorting table. The empty waste container is put on a fixture and the robotic arm with the gripper is now free for the sorting task.

B. Sorting table

A suitable sorting table was set up, which allows SCARAB to attach the waste container to the table and then move partially under the table itself. In this way, it is possible to optimize the working space of the robot arm. The sorting table has 2 storage bins, to the left and right of the sorting surface, into which the cans and bottles are deposited. Once the sorting process is complete, the sorting area can be tilted with the robot arm and the remaining waste falls into a residual waste container, as shown in Fig. 5. The sorting table is not equipped with any actuators or electronics.

C. Gripper

A passive gripper system was originally developed for manipulating the waste containers to ensure the most robust

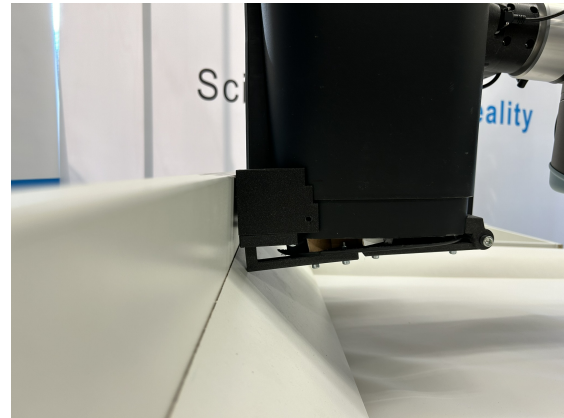


Fig. 3. Mechanism to open the container at the bottom without actuators by pressing the container to the rear wall of the sorting table.

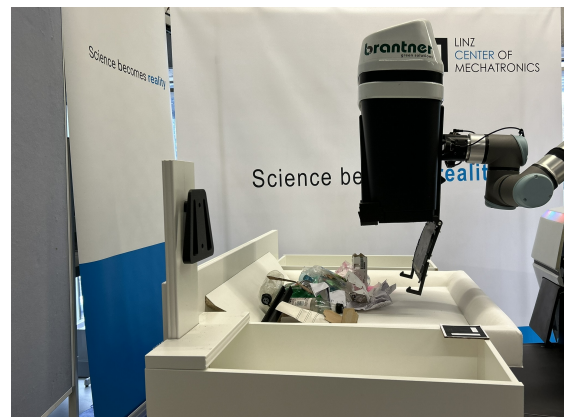


Fig. 4. Container with opened bottom. The container will be placed on the fixture after closing the bottom.



Fig. 5. The sorting table is also operated by the robotic arm without additional actuators.



Fig. 6. Different designs of the gripper: Left with rigid fingers, right with fin ray design. In the right figure, the Gimatic gripper is shown in red and blue and the capacitor box is under the lid with the LCM logo.

and safe handling possible. However, a gripper is now required for sorting the waste.

A self-centring electric angle gripper from Gimatic was used as the gripper for sorting the waste. This fulfils the special requirements in terms of available installation space, closing force and compatibility with the robot arm. The exact type of gripper is 'MPBM3240'. The gripper requires additional external control electronics (Capacitor Box CAPBOX3200-03), which must be used to provide the power for the Gimatic gripper. Without these electronics, the power requirement at the pins on the wrist of the UR10e robot arm could not be covered. Furthermore, different fingers can be used thanks to the modular design. The gripper also allows rapid adaptation to other problems, as the fingers can be created and customised using rapid prototyping.

As the available installation space is very limited in the folded state, the angular gripper was integrated into the existing robot end effector to save as much space as possible. An adapter plate was designed for this purpose, which must be fitted to the robot's wrist in the first assembly step. The Gimatic angular gripper can then be screwed onto this adapter and fixed in place. The electronics of the Capacitor Box are located directly in front of the gripper on the robot end effector. The original robot end effector has been adapted accordingly so that it can be mounted on the adapter with the Intel Realsense camera fitted.

The first version of the fingers was 3D-printed from TPU (thermoplastic material) and is shown in Fig. 6. As the narrow design of the fingers led to twisting when gripping and different objects were not always gripped correctly, a new finger design was tested.

The new design of the fingers was based on the so-called Fin Ray design, which has already been successfully used in the literature to grasp variable shapes, [8], [3], [11]. This design is originally biologically inspired by the tail fins of fish and patented by the company Evologics GmbH. The company Festo offers commercial products of gripper fingers based on this concept. The soft gripper used in our studies is lightweight (entirely 3D-printed from TPU), has a simple structure, high compliance and adaptability, and is capable of grasping objects of any geometry. Fig. 7 shows how the principle of the Fin Ray design works: In the unloaded state, the design retains its original shape. If any object is gripped (for example an already deformed aluminium can), the gripper automatically adapts to the shape of the object. This enables various objects to be gripped safely. As the



Fig. 7. Elastic fingers with fin ray design in opened and closed configuration. (The LED lights still need to be installed next to the Realsense camera.)

inner gripper surfaces are aligned parallel to each other in the open state, the object is automatically pressed towards the gripper when gripping.

It is also important to mention that the material of the functional model (except the fingers) is PLA (polylactide). PLA is not resistant to ultraviolet radiation (UV) and should be replaced with a UV-resistant material if necessary. If the first tests are successful, a change to the commercial product of FESTO will be considered.

IV. MANIPULATING THE OBJECTS

In order to sort out the valuable objects, it is necessary to identify them within the residual waste, grab them robustly and place them in separate containers.

A. Segmentation with YOLOv8 model

An instance segmentation model was selected in order to not only obtain a bounding box of the objects, but also to detect the exact contour of the waste object. This property is important in the later calculation of the gripping point in order to be able to analyse the shape of the object. Therefore, a YOLOv8 model [5] was used to detect the valuable parts of the waste.

The model was first trained with the TACO dataset [9], an open image dataset of waste in the wild. This dataset has 63 classes of objects but only "clear plastic bottle", "drink can" and "food can" are used in our work.

To create additional training images, the waste container was filled and opened several times from a defined height in the center of the table. In total 200 photos were taken of different waste distributions on the table. 160 photos were used for training and 40 for validation. A semi-automatic labeling tool was developed as all contours of the objects must first be labeled for each photo in order to be able to train the network later. This would be very time-consuming with manual labeling. The Segment Anything Model (SAM) [6] implemented by Meta was used for this purpose, which saves a great deal of time when labeling the waste objects. The online version of SAM can not be used for generating the training data as no labels are available. Meta provides the code as open source and it was possible to use this code for developing a standalone offline tool for labeling the images taken in our lab. The workflow is the following:

- 1) Click on a single object in the image and SAM will highlight automatically (at least a part) of the object.



Fig. 8. Validation of the segmentation: The training data labeled with the offline tool based on SAM.



Fig. 9. Validation of the segmentation: Result of the YOLOv8 model with the same picture as in the training.

- 2) Add or subtract parts of the object by continue clicking with the mouse.
- 3) When the entire object is highlighted, enter the appropriate label for the object.
- 4) Continue with the next object in the image.
- 5) When all objects in the image are labeled, continue to the next image.

Comparing Fig. 8 and Fig. 9 shows the good results of the segmentation algorithm, including concealed and deformed objects. For the manual labeling only the classes "food can" and "clear plastic bottle" have been used, which will be called "bottle" and "can" in the following.

B. Gripping pose

The Realsense camera is used to find the gripping positions of the objects. The first step is to take a 2D photo and a depth image with the camera mounted on the robotic arm from a well defined position right above the sorting table.

With the YOLOv8 model, the objects are segmented in the 2D photo and processed one after the other. As output of the YOLOv8 model the contour of each object is provided in 2D together with a label and a numerical value for the confidence, as shown in Fig. 10 for a bottle which is obstructed by a sheet of paper. With the function `minAreaRect` of OpenCV library [2] the center point, orientation and main axis of the object contour are computed. The distance between the camera and the gripping point is determined with an ArUco marker [4]. The 3D position of this gripping point can then

be calculated using the usual camera calibration algorithms. The following assumptions are made in order to calculate the 6D pose of the gripping point from the position: The gripper is parallel to the image plane and rotated around the global vertical axis corresponding to the rotation of the 2D object contour, as shown in Fig. 11. A safe gripping of the objects was observed, even in the cases when only small parts of the object are visible, as shown in Fig. 12.

V. TEST RESULTS

The robustness of the waste sorting process described above was tested extensively. The objects to be sorted out of the residual waste were not part of the training data and can be seen in Fig. 13. The test data consists of 4 bottles and 5 cans. In addition to these desired objects, the test waste contains 15 disturbing objects, which were also not part of the training data: Plastic packaging films, cardboard and paper.

The tests were done in the following way:

- 1) fill the waste (desired and disturbing objects) into the bin and mix thoroughly
- 2) empty the waste on the sorting table
- 3) distribute the waste with the robotic arm
- 4) take a picture of the waste
- 5) grasp a desired object and put it into the bins next to the sorting table
- 6) repeat steps (4) and (5) until no more desired objects are detected

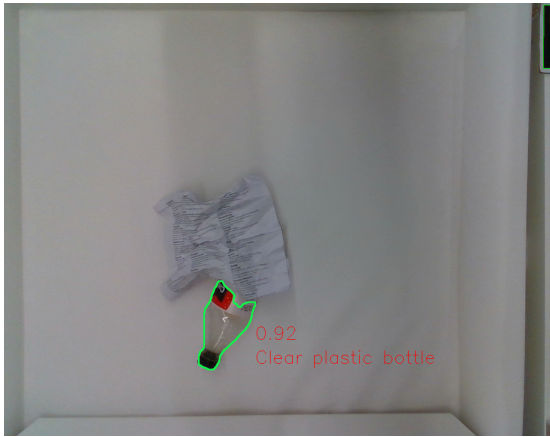


Fig. 10. Output of the YoloV8 model: visible contour (shown in green) and label (with confidence) of the object.



Fig. 11. Computation of the gripping point: bounding box (light blue rectangular) with center point and its rotation in degree.

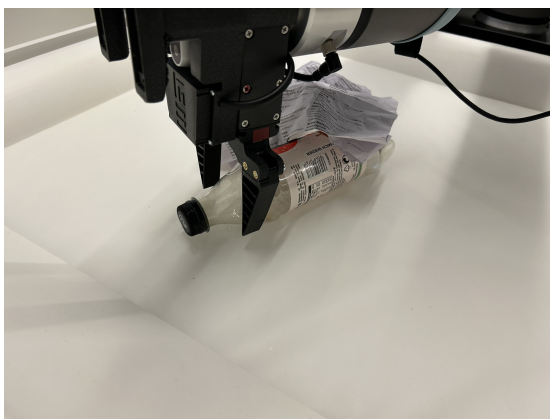


Fig. 12. Gripping in the center of the bounding box of the (visible) contour of the bottle.

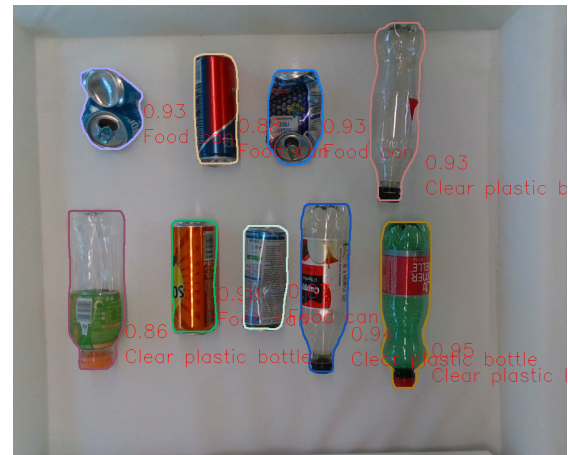


Fig. 13. Test objects with their classes according to the YoloV8 model.

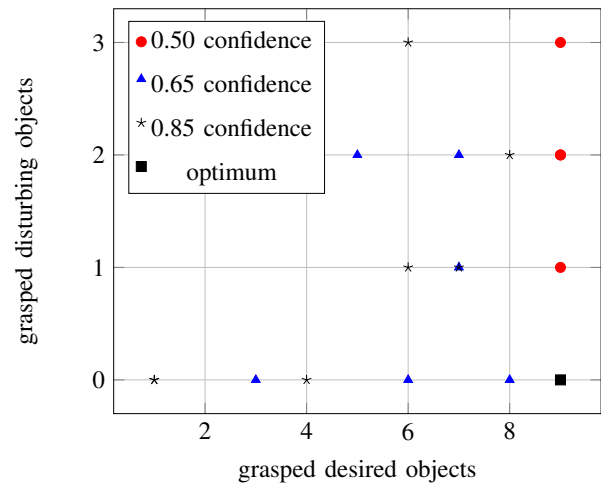


Fig. 14. Results of the tests with different settings of the confidence parameter

These tests were repeated multiple times with different settings. The most significant parameter regarding the performance of the sorting process was the confidence of the segmentation step. A low confidence value leads to a high number of successfully picked objects. However, you have to accept that a few unwanted objects will also be picked up. In Fig. 14 the results of the tests are shown. The optimal result would be to pick 9 out of 9 desired objects and 0 out of 15 disturbing objects. If the same result was observed multiple times with the same setting, the result is still just shown as a single point in the graph. The graph shows, that it was not possible to reach the optimal result with any setting and that it was not possible to strictly avoid grasping disturbing objects. However with setting the confidence to 50% it was possible to reach a robust result of picking all desired objects while accepting to pick 1 to 3 of the 15 disturbing objects.

A more detailed evaluation of the detection process was done to study the influence of the confidence value. For each of the 150 photos taken during the tests, it was analyzed

TABLE I

INFLUENCE OF THE CONFIDENCE SETTINGS ON THE DETECTION RATE.
AVERAGE VALUES BASED ON 150 IMAGES.

confidence setting	correct detected objects	wrong objects per image
0.50	83%	0.57
0.65	40%	0.19
0.85	34%	0.17

TABLE II

RECORDED TIME FOR THE OBJECT DETECTION IN SECONDS. AVERAGE
VALUES BASED ON THE RECORDING OF 16 OBJECTS.

take picture	0.046
preprocess picture	0.056
segment object (incl. saving the picture)	2.182
compute bounding box	0.158
compute gripping pose (incl. 2 coordinate transformations)	0.010

how many of the desired objects depicted were correctly recognized and how many of the undesired objects were erroneously marked. The average values for the 3 different settings can be seen in Table I. The value for the "wrong objects per image" is an absolute value and is between 0.17 and 0.57 objects per image. The correctly detected objects are given as percentage of the desired objects in the image and differs between 34% for a high confidence value and 83% for a low confidence value.

The time required for object recognition depends heavily on the hardware used. In the tests shown here, the photo was taken using a Realsense camera, the data was read out via the RTDE interface of the Universal Robot and then analyzed with a Python script. The evaluation was carried out on a NUC (Next Unit Computing). All computations are performed locally with hardware located in the SCARAB platform. The duration of the individual steps is shown in Table II.

VI. SUMMARY AND OUTLOOK

It was demonstrated how the functionality of an existing mobile robot was extended with low cost hardware to add the feature of waste sorting. The low amount of training data in the lab still limits the quality of the overall performance but was sufficient to find the most significant parameter. A good choice of the limit for the confidence in the segmentation step has large impact on the results. During operation SCARAB will collect much more (and more realistic) training data on a daily basis which will lead to a more robust performance.

ACKNOWLEDGMENT

This work has been supported by the COMET-K2 Center of the Linz Center of Mechatronics (LCM) funded by the Austrian federal government and the federal state of Upper Austria.

REFERENCES

- [1] P. Beigl and A. Allesch, "Einwegpfand in österreich: Gestaltung und herausforderungen im internationalen vergleich," *Österreichische Wasser-und Abfallwirtschaft*, pp. 1–11, 2024.
- [2] G. Bradski, A. Kaehler, et al., "OpenCV," *Dr. Dobb's journal of software tools*, vol. 3, no. 2, 2000.
- [3] J. M. Gandarias, J. M. Gómez-de Gabriel, and A. J. García-Cerezo, "Enhancing perception with tactile object recognition in adaptive grippers for human–robot interaction," *Sensors*, vol. 18, no. 3, p. 692, 2018.
- [4] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.
- [5] G. Jocher, J. Qiu, and A. Chaurasia, "Ultralytics yolo (version 8.0.0) [computer software]." 2023. [Online]. Available: <https://github.com/ultralytics/ultralytics>
- [6] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al., "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.
- [7] W. Lu and J. Chen, "Computer vision for solid waste sorting: A critical review of academic research," *Waste Management*, vol. 142, pp. 29–43, 2022.
- [8] O. Pfaff, S. Simeonov, I. Cirovic, P. Stano, et al., "Application of fin ray effect approach for production process automation," *Annals of DAAAM & Proceedings*, vol. 22, no. 1, pp. 1247–1249, 2011.
- [9] P. F. Proença and P. Simões, "Taco: Trash annotations in context for litter detection," *arXiv preprint arXiv:2003.06975*, 2020.
- [10] D. Reischl, J. Wenninger, J. Schröck, A. Reininger, M. Trinkl, J. Parz, and M. Weywoda, "Scarab - autonomous demand-oriented waste disposal with mobile robot," in *Proceedings of The First Austrian Symposium on AI, Robotics, and Vision (AIROV24)*, 2024.
- [11] Y. Yang, K. Jin, H. Zhu, G. Song, H. Lu, and L. Kang, "A 3d-printed fin ray effect inspired soft robotic gripper with force feedback," *Micromachines*, vol. 12, no. 10, p. 1141, 2021.

Mechanical Design Optimization of a Pneumatically Actuated Parallel Kinematic Machine

Klemens Springer¹, Hubert Gatringer² and Andreas Müller²

Abstract—The application field of motion simulation needs robotic platforms with a high level of dexterity payload. Therefore increasingly parallel manipulators/platforms are used as 3 to 6 degree-of-freedom constructions. The contradictory aim for high applicable forces and large workspace volumes necessitates an optimization of the mechanical construction. In contrast to common configurations the robot utilized here is a hexapod equipped with antagonistic type of pneumatic actuation, imitating the flexor-extensor principle of human muscles. A counter force is applied passively through a spring in the center point of the hexapod. This structure offers advantages for application as motion simulator such as little maintenance requirements and low cost assembly. Due to the direct correlation between actuator length and dynamics, the use of classical techniques for workspace evaluation in the area of design optimization is not applicable. The paper illustrates the optimal design of this parallel kinematic machine concerning maximum workspace taking into account the dynamical system. The presented method ensures stability in the upper maximum possible position through an additional optimization of the maximum disturbance force. The resulting multi-objective optimization problem is solved by using an evolutionary algorithm with a Pareto approach. The introduced method for evaluating an adequate measure of the maximum workspace volume for parallel platforms is well suited in the application field of motion simulators. The optimal solutions of the Pareto front are evaluated and compared to the parameters used in the existing configuration of the platform at the Institute of Robotics.

Keywords: multi-objective optimization, parallel robots, design optimization, motion simulator, pneumatic actuation

I. INTRODUCTION

Parallel kinematic machines have received growing attention in the fields of vibration damping, medical surgery and industrial applications like toolheads in the last few years, see [1], [2]. Originally invented for motion simulation (see [3], [4]), which is the purpose here as well (Fig. 1), hexapods have successfully asserted themselves in this area. Following the most accurate definition *Gough platform* is used for the parallel platform. The main advantages, good accuracy and dexterity, of a Gough platform accompany the disadvantage of small workspace, which is most important for the given application. Thus a main aim within the mechanical design of these platforms is the maximization of the workspace volume without losing the advantageous properties, see [5], [6], [7]. In the last years a lot of research has been done in the optimization of the dynamic behavior and compliance



Fig. 1: Motion simulator mounted on the parallel platform

by Zhang in [8], stiffness by Krefft in [9], manipulability by Wen in [10] and general workspace maximization with respect to constructive constraints by Masory in [11]. Hardly any attention has been paid to mechanical design optimization concerning antagonistic actuation systems with a passive component. This article introduces new techniques for the workspace optimization of a pneumatically actuated 6-degree of freedom Gough platform including dynamical considerations. That necessity results from the direct correlation between the kinematics (contraction) and dynamics (pressure) of the actuator. Due to the lack of the possibility to impress forces of arbitrary directions by the pneumatic actuators, see Fig. 2, a spring is mounted in the center of the construction to passively apply opposite forces and torques. In order to maximize the possible disturbance force at the topmost pose, an additional objective criteria is introduced for avoiding the loss of manipulability. Countless authors addressed single-objective optimizations of parallel mechanisms. This approach leads to a dominant problem for the present contradictory formulation. To find an appropriate solution, it is formulated as a multi-objective optimization problem. Genetic algorithms, that are predestined for non-convex and non-smooth optimization formulations, use evolutionary strategies from genetic programming to cope these types of problems, see [12], [13]. In contrast to standard gradient-based solvers, they have no need for gradient information, are nearly independent of discontinuities and are more efficient in performing a global search. To allow for multi-objective considerations, a Pareto approach in combination with genetic algorithms is used.

In accordance with the contents presented above, this paper

¹Klemens Springer, Engel Austria GmbH, 4311 Schwertberg, Austria

² Hubert Gatringer, Andreas Müller are with Institute of Robotics, Johannes Kepler University Linz, 4040 Linz, Austria {hubert.gatringer,a.mueller}@jku.at

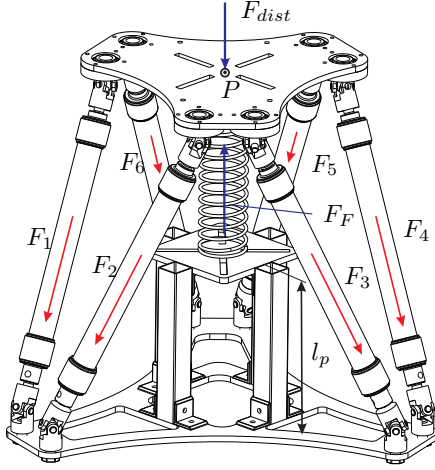


Fig. 2: Possible directions of actuator and spring forces

is arranged as follows. After a description of the modeling of the mechatronic system including kinematical and dynamical considerations (section 2), the formulation of the optimization problem is shown in section 3. Introducing new techniques, the calculation of the workspace and maximum disturbance force is illustrated. Section 4 focuses on the explanation and implementation of the problem formulation through a genetic programming based solver with a Pareto approach for multi-objective considerations. Furthermore the results of the optimization are presented. At the end of the paper in section 5 a conclusion for the used technique is drawn.

II. MODELING OF THE MECHATRONIC SYSTEM

The considered mechatronic system is split into a kinematical and a dynamical section, containing the pneumatic subsystem as well.

A. Kinematic description

The considered robot consists of two rigid platforms - the fixed base and the movable, coaxially arranged, upper one. They are connected by six flexible pneumatically driven fluidic actuators and a spring in the center of the robot, see again Fig. 2. This concept is based on the principle of the human muscle system, whereby here the opponent to the muscles is a passive one. The inertial coordinate system is chosen in the center of the base platform. In order to calculate the maximum workspace, the inverse kinematics, that describes the actuator lengths in dependence of the Cartesian coordinates of the tool center point P , is needed. For this, the solution of ${}_I\mathbf{l}_i = {}_I\mathbf{r}_P + \mathbf{R}_{I4} {}_4\mathbf{r}_{bi} - {}_I\mathbf{r}_{ai}$ has to be found (see Fig. 3), where the endpoint vector ${}_I\mathbf{r}_P$ is equivalent to the first three entries of the minimal coordinates $\mathbf{q} = [x \ y \ z \ \alpha \ \beta \ \gamma]^T$.

There, the angles α , β , and γ represent the rotation of the upper platform in Cardan description and x , y , z the position relative to the inertial coordinate system. The rotation matrix $\mathbf{R}_{I4}$ relates the body-fixed coordinate system ${}_4K$ in the

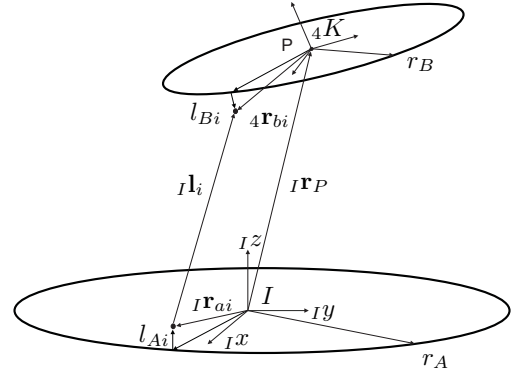


Fig. 3: Coordinate systems and kinematics for one arm

center of the upper platform to the inertial frame. The vectors ${}_I\mathbf{r}_{ai}$ and ${}_4\mathbf{r}_{bi}$ to the actuator contact points are calculated as functions of the optimization variables r_A , r_B (radii of the mounting mounts of the actuators), α_{off} and β_{off} (offset angles), shown in Fig. 4 and Fig. 3.

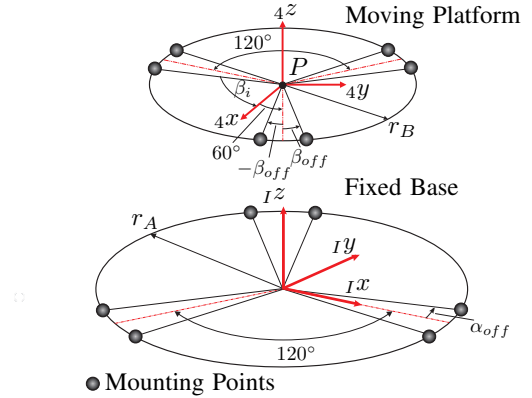


Fig. 4: Angular offsets

Constraints: For respecting constructive constraints, the maximum actuator lengths $l_{max} = l_0 (1 + 0.05)$, $l_{min} = l_0 (1 - 0.25)$, given by the manufacturer's specifications, and passive joint angles

$$\begin{aligned} \theta_{A,iMin} &\leq \theta_{A,i} = \cos^{-1} ({}_I\mathbf{e}_3^T {}_I\mathbf{u}_i) \leq \theta_{A,iMax} \quad , \quad i = 1 \dots 6 \\ \theta_{B,iMin} &\leq \theta_{B,i} = \cos^{-1} ({}_4\mathbf{e}_3^T {}_4\mathbf{u}_i) \leq \theta_{B,iMax} \quad , \quad i = 1 \dots 6 \end{aligned} \quad (1)$$

with the actuator direction vectors and the unit vectors in the respective coordinate systems

$$\begin{aligned} {}_I\mathbf{u}_i &= \frac{{}_I\mathbf{l}_i}{\|{}_I\mathbf{l}_i\|} = \frac{{}_I\mathbf{r}_P + \mathbf{R}_{I4} {}_4\mathbf{r}_{bi} - {}_I\mathbf{r}_{ai}}{\|{}_I\mathbf{r}_P + \mathbf{R}_{I4} {}_4\mathbf{r}_{bi} - {}_I\mathbf{r}_{ai}\|} \quad , \quad i = 1 \dots 6 \\ {}_I\mathbf{e}_3^T &= [0, 0, 1] \quad , \quad {}_4\mathbf{e}_3^T = [0, 0, 1] \end{aligned} \quad (2)$$

have to be formulated (Fig. 5). The pneumatic actuators have a nominal length of l_0 and are fixed with universal joints, mounted in axial bearings, in the upper platform. As a consequence of this additional degree of freedom, the upper universal joints do not constrain the maximum angles

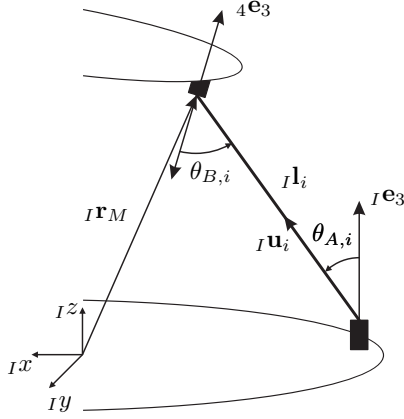


Fig. 5: Kinematic constraints for joint angles

of inclination $\theta_{B,i} = 90^\circ$. In contrast to this, only 60° are allowed for the universal joints' inclination angles $\theta_{A,i}$ at the base platform. Actuator collisions can be neglected because other constraints become active before they would occur.

B. Dynamical description

The equations of motion in minimal description are calculated with the projection equation, see [14] and results in

$$\begin{aligned} \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{g}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{K}\mathbf{q} &= \mathbf{Q}_m = \mathbf{B}(\mathbf{q})\mathbf{F}_m \\ \mathbf{F}_m &= [F_1, F_2, \dots, F_6]^T \end{aligned} \quad (3)$$

see [15] for details. $\mathbf{M}(\mathbf{q})$ is the mass matrix, and $\mathbf{g}(\mathbf{q}, \dot{\mathbf{q}})$ contains the remaining nonlinear terms (gravity, centrifugal, Coriolis). $\mathbf{K}$ represents the stiffness matrix due to the spring forces. The generalized driving forces $\mathbf{Q}_m$ can be separated into the input matrix $\mathbf{B}(\mathbf{q})$ and the actuator forces $\mathbf{F}_m$. These actuator forces are projected into the minimal space by

$$\mathbf{Q}_m = \sum_{i=1}^6 \mathbf{J}_{m,i}^T \mathbf{I}\mathbf{u}_i F_i. \quad (4)$$

The partial derivatives of the vectors to the actuator mount base $\mathbf{I}\mathbf{r}_{Mi}$, see Fig. 5, yield the Jacobian

$$\mathbf{J}_{m,i} = \frac{\partial \mathbf{I}\mathbf{r}_{Mi}}{\partial \mathbf{q}} = \frac{\partial (\mathbf{I}\mathbf{r}_P + \mathbf{R}_{I4} \mathbf{4}\mathbf{r}_{bi})}{\partial \mathbf{q}}, \quad i = 1 \dots 6. \quad (5)$$

Eq. (4) can be combined to

$$\mathbf{Q}_m = \mathbf{B}(\mathbf{q})\mathbf{F}_m. \quad (6)$$

1) Pneumatic subsystem: The 6 pneumatic subsystems consist of a fluidic actuator by FESTO, called fluidic muscle, an analog proportional valve, a pressure sensor and a linear potentiometer to measure the actuator lengths $\|\mathbf{I}\mathbf{l}_i\|_2$. The muscles are made of a fiber-reinforced rubber tube with mounting flanges at the ends. The actuator operates as follows: Air flows into the tube and leads to increasing pressure p_i , $i = 1 \dots 6$ and thus to a broadening of the muscle. Because of specially arranged fibers this results in a contraction h of the muscle

$$h_i = \frac{l_{0,i} - \|\mathbf{I}\mathbf{l}_i\|_2}{l_{0,i}} 100\%, \quad i = 1 \dots 6 \quad (7)$$

in percent in longitudinal direction with the relaxed link length $l_{0,i}$ of muscle i . This fact is used to generate pulling forces

$$F_i = \left(p_i \sum_{k=1}^{n_a} a_k h_i^k + \sum_{k=1}^{n_b} b_k h_i^k \right), \quad i = 1 \dots 6 \quad (8)$$

that have nonlinear characteristics and depend on the pressures p_i and the contractions h_i . The polynomial coefficients a_k , b_k are derived from a mathematical approximation of the actuator's characteristics given by the manufacturer, see Fig. 6.

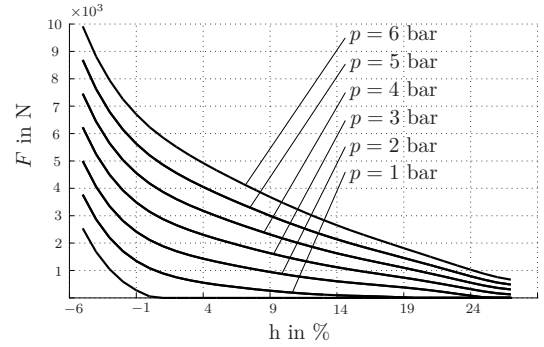


Fig. 6: Characteristics of the fluidic muscle DMSP40 by FESTO

2) Identification of the spring parameters: In order to describe the dynamical model from section II-B as exact as possible, which is needed for the optimization formulation, the spring parameters have to be known. Therefore an identification of the stiffness matrix $\mathbf{K}$ is done based on the Least Squares method, see [16] for details. This identification was verified through calculating a feedforward control based on the identified spring stiffness matrix and evaluating the position error. The error is around 0.5mm in the middle of the workspace and not much higher in the topmost and the lowest pose.

Since the identification is only done relative to the one set of muscles used for the measurement and the specified repeatability is $\leq 1\%$, the inaccuracy of the actuators does not influence the spring identification. More sophisticated models of the spring using neuronal networks can be found in [17].

III. OPTIMIZATION PROBLEM

As discussed extensively in numerous publications, the most important optimization criterion in the process of mechanical design of parallel kinematic machines is the maximization of the workspace. Commonly the workspace is iteratively evaluated with a method, based on the inverse kinematic of the Gough platform, see e.g. [11]. Different from this approach an application adequate measure is used here as a substitution for the workspace volume. This simplification balances out the increased calculation effort caused by the necessary consideration of the actuator dynamics while resulting in a quality of the solution that is sufficient for motion

simulation. Furthermore the orientation is included in the used application adequate measure.

A. Workspace

Due to the high complexity of the analytical calculation of the workspace, a numerical approach similar to that in [11], has been chosen. Based on the assumption, that the volume of the cubage can be approached with $V_{AR} = \frac{1}{3}(x_{max} - x_{min})|_{z=z_{mid}}(y_{max} - y_{min})|_{z=z_{mid}}(z_{max} - z_{min})$ with the workspace center $z_{mid} = (z_{min} + z_{max})/2$, the maximum translational displacements x_{min} , x_{max} , y_{min} , y_{max} , z_{min} and z_{max} have to be calculated. Therefore the actuator lengths of 6 reference poses

$$\begin{aligned} \mathbf{q}_{P_1} &= [0, 0, z_{min}, 0, 0, 0] \\ \mathbf{q}_{P_2} &= [0, 0, z_{max}, 0, 0, 0] \\ \mathbf{q}_{P_3} &= [x_{min}, 0, z_{mid}, 0, 0, 0] \\ \mathbf{q}_{P_4} &= [x_{max}, 0, z_{mid}, 0, 0, 0] \\ \mathbf{q}_{P_5} &= [0, y_{min}, z_{mid}, 0, 0, 0] \\ \mathbf{q}_{P_6} &= [0, y_{max}, z_{mid}, 0, 0, 0] \end{aligned} \quad (9)$$

are evaluated with respect to the length and joint angle constraints. Only constraints concerning the lower joint angles have to be considered. The procedure is introduced briefly:

- 1) Starting at an infeasible point, e.g. $\mathbf{q}_P = [0]$, the actuator lengths and joint angles for an increasing z coordinate are iteratively calculated. The first point that does not violate the constraints represents the minimum displacement z_{min} .
- 2) Starting at $\mathbf{q}_{P_1}$ the z coordinate is increased again and the actuator lengths and joint angles are iteratively calculated. If the constraints are violated, the last feasible position reveals the maximum displacement z_{max} . Hence the relative displacement $\Delta z = (z_{max} - z_{min})$ is calculated.
- 3) Next the actuator lengths and joint angles are calculated for positions with increasing displacements in the x and y coordinates one axis after the other, starting at the workspace center $(\mathbf{q}_{P_1} + \mathbf{q}_{P_2})/2$. If the constraints are violated, the last feasible position reveals the maximum displacement x_{max} or y_{max} .
- 4) The same way x_{min} and y_{min} are computed with decreasing displacements starting at the workspace center, which from $\Delta x = (x_{max} - x_{min})$ and $\Delta y = (y_{max} - y_{min})$ ensues.

On the basis of this algorithm, an objective function for evaluating an application adequate measure as approximation for the workspace volume of this hexapod, treated as a rigid mechanism, can be suggested. The calculation of the maximum positions starting at $(\mathbf{q}_{P_1} + \mathbf{q}_{P_2})/2$ is completely admissible in the specific application field of a motion simulator, whose default position is at $\mathbf{q}_0 = (\mathbf{q}_{P_1} + \mathbf{q}_{P_2})/2$. The optimization variables to manipulate this measure are the platform radii r_A , r_B and the offset angles α_{off} , β_{off} , see Fig. 4. Reconsidering the direct correlation between

impressed force and contraction of the pneumatic actuators, it comes clear that not only kinematics have to be kept in mind, but also the dynamic modeling and a dynamic mass calculation due to variable platform radius r_B . Consequently, the workspace calculation has to be extended and the maximum positions are calculated with respect to dynamics. Regarding this problem statically with $\dot{\mathbf{q}}, \ddot{\mathbf{q}} = \mathbf{0}$, the actuator forces

$$\begin{aligned} \mathbf{F}_m(\mathbf{q}, \dot{\mathbf{q}} = \mathbf{0}, \ddot{\mathbf{q}} = \mathbf{0}) &= \\ \mathbf{B}(\mathbf{q})^{-1} \left(\underbrace{\mathbf{M}(\mathbf{q})}_{\mathbf{0}} \ddot{\mathbf{q}} + \mathbf{g}(\mathbf{q}, \mathbf{0}) + \mathbf{K}\mathbf{q} \right) & \quad (10) \\ \mathbf{F}_m(\mathbf{q}) &= \mathbf{B}(\mathbf{q})^{-1} (\mathbf{K}\mathbf{q} + \mathbf{g}(\mathbf{q}, \mathbf{0})) \end{aligned}$$

are calculated via the inverse dynamics. Hence and in combination with the muscle contractions at the current position gained via inverse kinematics and Eqn. (7), the required muscle pressures

$$p_i = \frac{F_i - \sum_{k=1}^{n_b} b_k h_i^k}{\sum_{k=1}^{n_a} a_k h_i^k}, \quad i = 1 \dots 6 \quad (11)$$

are determined. If the pressure constraints $0 \leq p_i \leq p_{max} = 6\text{bar}$, given by manufacturer specifications, are violated, then the displacement $z_{min,act} = z_{min,act} + \Delta z_{red}$, exemplary shown for the minimum displacement in z -direction, is reduced consecutively by a minimal value Δz_{red} till a feasible position is found.

In order to include the orientation in this measure, the maximum rotatory displacements ϕ , analogously to the workspace volume evaluation, are calculated in 6 reference poses

$$\begin{aligned} \mathbf{q}_{P_7} &= [0, 0, z_{mid}, \alpha_{min}, 0, 0] \\ \mathbf{q}_{P_8} &= [0, 0, z_{mid}, \alpha_{max}, 0, 0] \\ \mathbf{q}_{P_9} &= [0, 0, z_{mid}, 0, \beta_{min}, 0] \\ \mathbf{q}_{P_{10}} &= [0, 0, z_{mid}, 0, \beta_{max}, 0] \\ \mathbf{q}_{P_{11}} &= [0, 0, z_{mid}, 0, 0, \gamma_{min}] \\ \mathbf{q}_{P_{12}} &= [0, 0, z_{mid}, 0, 0, \gamma_{max}] \end{aligned} \quad (12)$$

Hence $\Delta\alpha = (\alpha_{max} - \alpha_{min})$, $\Delta\beta = (\beta_{max} - \beta_{min})$ and $\Delta\gamma = (\gamma_{max} - \gamma_{min})$ result. Now the workspace evaluation function can be stated as

$$\begin{aligned} \Psi_{AR} &= \frac{1}{2}W_{11}\Delta x^2 + \frac{1}{2}W_{22}\Delta y^2 + \frac{1}{2}W_{33}\Delta z^2 + \\ &\quad \frac{1}{2}W_{44}\Delta\alpha^2 + \frac{1}{2}W_{55}\Delta\beta^2 + \frac{1}{2}W_{66}\Delta\gamma^2 \\ &= \frac{1}{2} \begin{bmatrix} \Delta\mathbf{r}^T & \Delta\phi^T \end{bmatrix} \mathbf{W} \begin{bmatrix} \Delta\mathbf{r} \\ \Delta\phi \end{bmatrix} = \frac{1}{2} \Delta\mathbf{q}^T \mathbf{W} \Delta\mathbf{q} \end{aligned} \quad (13)$$

$$\mathbf{W} \geq 0$$

with the positive definite diagonal weighting matrix $\mathbf{W} = \text{diag}(0.5, 0.5, 5, 0.5, 0.5, 0.05)$ and the diagonal entries W_{ii} . The maximum displacements are represented with $\Delta\mathbf{r}^T = [x_{max} - x_{min}, y_{max} - y_{min}, z_{max} - z_{min}]$ and $\Delta\phi^T = [\alpha_{max} - \alpha_{min}, \beta_{max} - \beta_{min}, \gamma_{max} - \gamma_{min}]$.

In the application of a motion simulator, gravity is used for simulating sustaining accelerations through tilting the pilot's seat. Therefore defined minimum required rotations $[\alpha_{lb}, \alpha_{ub}] = [-10^\circ, 10^\circ]$ and $[\beta_{lb}, \beta_{ub}] = [-10^\circ, 10^\circ]$ are

postulated for the roll angle α and the pitch angle β , that are considered through an inequality constraint

$$\Psi_{AR,c} = \begin{cases} \Psi_{AR} & \alpha_{max} > \alpha_{ub} \wedge \alpha_{min} < \alpha_{lb} \wedge \\ & \beta_{max} > \beta_{ub} \wedge \beta_{min} < \beta_{lb} \\ 0 & \text{else} \end{cases} \quad (14)$$

implemented in a constrained workspace evaluation function.

B. Disturbance force

Numerous publications concerning research in stiffness, compliance and dynamical optimization can be found, see [8], [9], [10], as mentioned in the introductory section. These considerations are featuring minor optimization potential for the used Gough platform, because of only one variable dynamic parameter (upper platform mass) and the predominant structural compliance due to the fluidic muscles and the spring. Another challenging peculiarity of this construction is the one-way force direction of the actuators. As a consequence a force impression in positive z direction in the upper reference pose $\mathbf{q}_{P_2}$ is not possible if an adequate pretension of the spring through the parameter l_p , see Fig. 2, is missing. Therefore the maximum possible disturbance force in negative z direction $F_{dist,max}$ has to be optimized with l_p as optimization variable. The force

$$F_{dist,max} = \mathbf{e}_3^T \sum_{i=1}^6 \mathbf{J}_{m,i}^T \mathbf{I} \mathbf{u}_i \Delta F_i \quad (15)$$

$$\mathbf{e}_3^T = [0, 0, 1, 0, 0, 0]$$

results out of the maximum applicable muscle forces ΔF_i with the Jacobian $\mathbf{J}_{m,i}$ for the transmission of the generalized forces, see Eqn. (5), and the unit vectors $\mathbf{I} \mathbf{u}_i$, see Eqn. (2) and Fig. 5. The required actuator forces

$$\Delta F_i = F_i(p_{min,i}) - F_i(p_{0,i}), \quad i = 1 \dots 6 \quad (16)$$

are gained out of the drive forces, see Eqn. (8), in the upper reference pose $\mathbf{q}_{P_2}$ with the unknown pressure

$$p_{min,i} = p(F_i(\mathbf{q} = \mathbf{q}_{P_2}, F_{dist} = F_{dist,max}), h_i(\mathbf{q}_{P_2})) \quad (17)$$

occurring at the impression of the unknown maximum disturbance force and the pressure

$$p_{0,i} = p(F_i(\mathbf{q} = \mathbf{q}_{P_2}, F_{dist} = 0), h_i(\mathbf{q}_{P_2})) \quad (18)$$

occurring in the absence of the disturbance force, see Eqn. (11). The needed forces in joint space $\mathbf{F}_m \in \mathbb{R}^6$ are a result of an adapted inverse dynamic, formulated out of Eqn. (10) with an additional disturbance term F_{dist}

$$\mathbf{F}_m(\mathbf{q}, F_{dist}) = \mathbf{B}(\mathbf{q})^{-1} (\mathbf{K}\mathbf{q} + \mathbf{g}(\mathbf{q}, 0) - F_{dist}\mathbf{e}_3). \quad (19)$$

In order to calculate $p_{min,i}$ we need to know that in the top-most position at least one actuator holds a relative pressure of $p_{min} = 0$ bar when the maximum controllable disturbance is impressed. Hence the maximum pressure reserve

$$\Delta p = \min_i \{0 - p_{0,i}\} \quad (20)$$

due to the muscle pressure constraints, mentioned in section III-A is determined. Furthermore, the muscle forces can be expressed with $p_{min,i} = p_{0,i} + \Delta p$ and the evaluation of Eqn. (16) and Eqn. (8) to

$$\Delta F_i = \Delta p \sum_{k=1}^{n_a} a_k h_d^k, \quad i = 1 \dots 6. \quad (21)$$

If force impressions of arbitrary directions are desired, then the maximum pressure reserve is computed to

$$\Delta p = \min_i \left\{ \frac{(0 - p_{0,i})}{\frac{\partial p_i}{\partial F_{dist}}} \right\} \frac{\partial p_i}{\partial F_{dist}}, \quad i = 1 \dots 6 \quad (22)$$

with the pressure rate gained through a difference approximation

$$\frac{\partial p_i}{\partial F_{dist}} = \frac{p_{\Delta F_{dist,i}} - p_{0,i}}{\Delta F_{dist}}, \quad i = 1 \dots 6$$

$$p_{\Delta F_{dist,i}} = p(F_i(\mathbf{q} = \mathbf{q}_{P_2}, F_{dist} = \Delta F_{dist}), h_i(\mathbf{q}_{P_2})), \quad (23)$$

and a small disturbance force ΔF_{dist} . With the evaluation of Eqn. (15) the second objective function is defined as well.

C. Optimization Problem

The overall optimization problem can now specified as

$$\begin{aligned} \max_{\mathbf{x}} \quad & J_1 = \Psi_{AR,c} \\ \max_{\mathbf{x}} \quad & J_2 = F_{dist,max} \\ \text{s.t.} \quad & r_B \leq r_A \\ & \underline{\mathbf{x}} \leq \mathbf{x} \leq \bar{\mathbf{x}} \end{aligned} \quad (24)$$

with the optimization variables $\mathbf{x} = [r_A, r_B, \alpha_{off}, \beta_{off}, l_p]$ that are bounded to lower bounds $\underline{\mathbf{x}}$ and upper bounds $\bar{\mathbf{x}}$.

IV. OPTIMIZATION PROCEDURE

The formulation of Eqn. (24) describes a multi-objective optimization problem. The objectives J_1 and J_2 behave contradictory, whereby a multicriterial approach is needed for finding an appropriate solution. Therefore a Pareto approach is applied, because using one objective function as a result of a *direct weight assignment* method for example does not describe a physical representation. The solver used here is the existent and versatile solver *gamultiobj* in *Matlab*. The resulting Pareto front in Fig. 7, that represents the number of non-dominated solutions, shows the dominance of the maximized disturbance force, evaluated in the objective function J_2 . Despite this fact, an applicable set of solutions has been found through the Pareto approach. The Pareto front also reveals solutions that allow very high disturbance forces in the upper maximum pose. Based on a maximum load of 150 kg, the parameters with the biggest workspace volume measure $\Psi_{AR,c}$ and sufficient $F_{dist,max}$ are gained out. Important for the construction in the application field of motion simulation are mainly the parameters $\Delta z, \Delta \alpha$ and $\Delta \beta$, as it emerges from the weighting values in section

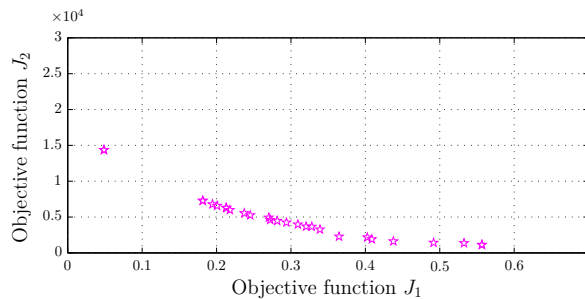


Fig. 7: Pareto front

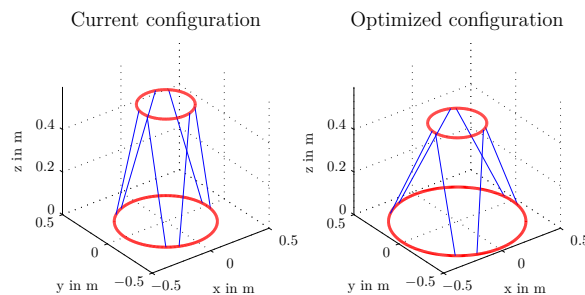


Fig. 8: Comparison between the current and one optimized configuration for the Gough platform

III-A, in order to simulate high-frequency up and down movements and sustaining accelerations through utilizing the gravitational vector. In comparison to the current configuration of the Gough platform at the Institute of Robotics, a huge improvement in the important degrees of freedom is achieved. This comparison of the resulting constructions is illustrated in Fig. 8, that shows the expected behavior. The narrower the construction is, the higher is the maximum applicable disturbance force and the more expanded the platform is, the bigger the workspace measure will get.

V. CONCLUSION

Optimizing the kinematic design parameters of the construction is a tradeoff between maximizing the admissible disturbance forces at the upper extremal position and maximizing the workspace. In the decision making process for the choice of a solution from the Pareto front, a compromise was made. Furthermore the combination of kinematics and actuator dynamics involves challenging difficulties. It was shown that, despite the pneumatic actuator dynamics, a workspace optimization with a maximization of the admissible disturbance force was possible using the methods of genetic algorithms in combination with a Pareto approach concerning multi-objective optimization formulations.

ACKNOWLEDGMENT

This work has been supported by the “LCM – K2 Center for Symbiotic Mechatronics” within the framework of the Austrian COMET-K2 program.

REFERENCES

- [1] J. P. Merlet, *Parallel Robots (Solid Mechanics and Its Applications)*, 2nd ed. Springer, 2001. [Online]. Available: <http://www.worldcat.org/isbn/1402003854>
- [2] A. Müller, “Parallel robots,” in *Robotics Goes MOOC*, B. Siciliano, Ed. Springer, 2025, pp. 89–165.
- [3] K. L. Cappel, “Motion simulator,” US Patent RE27051, 1964.
- [4] D. Stewart, “A platform with six degrees of freedom,” in *Proceedings of the Institution of Mechanical Engineers*, vol. 180, 1965, pp. 371–386.
- [5] J. P. Merlet, “Determination of 6d workspaces of gough-type parallel manipulator and comparison between different geometries,” *The International Journal of Robotics Research*, vol. 18, no. 9, pp. 902–916, 1999. [Online]. Available: <http://ijr.sagepub.com/content/18/9/902.abstract>
- [6] D. Kim, W. Chung, and Y. Youm, “Geometrical approach for the workspace of 6-dof parallel manipulators,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 4, 1997, pp. 2986–2991.
- [7] M. Badescu and C. Mavroidis, “Workspace optimization of 3-legged upu and ups parallel platforms with joint constraints,” *ASME Journal of Mechanical Design*, vol. 126, pp. 291–300, 2004.
- [8] D. Zhang, Z. Xu, C. M. Mechefske, and F. Xi, “Optimum design of parallel kinematic toolheads with genetic algorithms,” *Robotica*, vol. 22, pp. 77–84, 2004. [Online]. Available: <http://portal.acm.org/citation.cfm?id=976500.976508>
- [9] M. Krefft and J. Hesselbach, “Elastodynamic optimization of parallel kinematics,” in *Proceedings of the IEEE International Conference on Automation Science and Engineering*, 2005, pp. 357–362.
- [10] J. Wen and L. Wilfinger, “Kinematic manipulability of general constrained rigid multibody systems,” *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 1020–1025, 1999.
- [11] O. Masory and J. Wang, “Workspace evaluation of stewart platforms,” *Advanced Robotics*, vol. 9, pp. 443–461, 1995.
- [12] M. Gen and R. Cheng, *Genetic Algorithms and Engineering Optimization (Engineering Design and Automation)*. Wiley-Interscience, 1999. [Online]. Available: <http://www.worldcat.org/isbn/0471315311>
- [13] S. Stan, M. Manic, R. Balan, and V. Maties, “Genetic algorithms for workspace optimization of planar medical parallel robot,” in *Proceedings of the IEEE International Conference on Emerging Trends in Computing*, 2009.
- [14] H. Bremer, *Elastic Multibody Dynamics: A Direct Ritz Approach*, G. Tzafestas, Ed. Springer-Verlag GmbH, 2008.
- [15] H. Gatringer, R. Naderer, and H. Bremer, “Modeling and control of a pneumatically driven stewart platform,” in *Motion and Vibration Control*, 2009, pp. 93–102.
- [16] H. Trogmann, *Neue Regelverfahren für eine pneumatische Stewart Plattform*. Masterarbeit, Johannes Kepler Universität Linz, 2009.
- [17] S. Gadringer, C. Mayr, H. Gatringer, and A. Mueller, “Neural network feedforward control for pneumatic hexapod excavator simulator,” in *Proceedings of Austrian Robotics Workshop 2023*, 2023, pp. 1–6.

Analysis and tuning of PID controller gains for DC servo drives using Garpinger's trade-off plots*

Simon Hoher¹ and Jakob Rehr²

Abstract—Although PID tuning for DC drives is widely studied, a structured, practical guide addressing robustness and setpoint tracking/disturbance rejection trade-offs is still lacking. This paper condenses established methods into a clear, step-by-step approach for optimal PID tuning using Garpinger's trade-off plots, aiming at practical use in industrial applications.

Index Terms—PID control, Garpinger's trade-off plots, AMIGO and Garpinger method, servo drives

I. INTRODUCTION

Precise PID tuning is essential for electric drive control in industrial settings, requiring fast disturbance rejection, setpoint tracking, and robustness. This paper presents the Approximate M-constrained Integral Gain Optimization (AMIGO) [7] and the Garpinger method [4] as structured tuning approaches addressing these needs. AMIGO ensures fast, robust control without overshoot; Garpinger adds flexibility by allowing gain adjustment without compromising performance. Though focused on DC motors due to their modeling simplicity, results apply to synchronous motors via PQ-transformation [6], which are standard in industry. Using Garpinger trade-off plots, we show that optimal gains can be selected directly, offering an efficient and practical tuning method suitable for broader adoption.

II. RELATED WORK

A. PID Control: Basics and Challenges

PID control is widely used due to its simplicity and robustness [3]. The controller output is defined as

$$u(t) = K_P \cdot e(t) + K_I \cdot \int_0^t e(\tau) d\tau + K_D \cdot \frac{de(t)}{dt},$$

where $u(t)$ is the controller's output, $e(t)$ the control error, K_P the proportional gain, K_I the integral gain and K_D the derivative gain.

Tuning the gains K_P , K_I , K_D is nontrivial, as it must ensure:

- Stability of the closed loop system,
- Fast response to setpoint changes and disturbances,
- Minimal overshoot,
- Robustness to model uncertainties.

In practice, cascaded control structures are often used to enhance performance (see Figure 1).

B. Cascaded Control

Cascaded control is widely used in motor control, especially for DC and synchronous motors [2]. It consists of nested loops (see Figure 1):

1. **Inner Control Loop:** The inner control loop regulates the motor's speed and quickly responds to load changes.

2. **Outer Control Loop:** The outer control loop handles position control. The primary goal of the outer loop is to maintain accurate position control and ensure closed-loop stability.

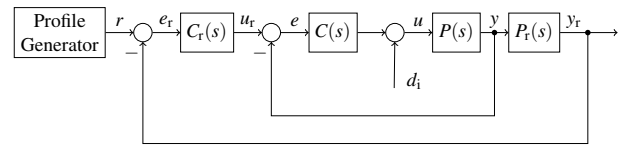


Fig. 1. Cascaded control with upstream profile generator

Typically, a trapezoidal velocity profile is used to provide the motor movement in three phases [3] (see Figure 2):

1. **Acceleration Phase:** The motor accelerates with a constant maximum acceleration $a_{\max}$ to the maximum speed $v_{\max}$.

2. **Constant Speed Phase:** After reaching the maximum speed, the motor continues to move at constant speed $v_{\max}$.

3. **Deceleration Phase:** The motor decelerates with the same maximum acceleration $-a_{\max}$ to precisely reach the final position.

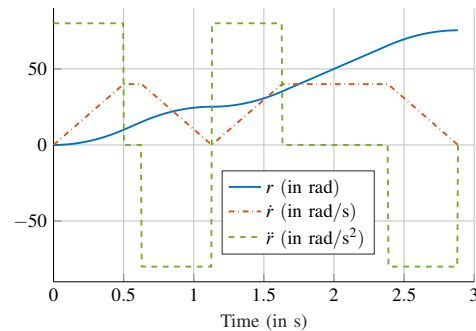


Fig. 2. Calculated trajectory with a trapezoidal velocity profile for given waypoints 4 and 12 revolutions ($v_{\max} = 40$ rad/s, $a_{\max} = 80$ rad/s²)

C. Garpinger's trade off diagrams

The design of PID controllers often involves balancing competing objectives, such as minimizing control error and

* The financial support of the Christian Doppler Research Society and the associated company partners of JRC ISIA is gratefully acknowledged

¹Salzburg University of Applied Sciences, JR Center for Intelligent and Secure Industrial Automation (JRC ISIA) simon.hoher@fh-salzburg.ac.at

²Salzburg University of Applied Sciences, JR Center for Intelligent and Secure Industrial Automation (JRC ISIA) jakob.rehr@fh-salzburg.ac.at

ensuring robustness against disturbances and model uncertainties. The Garpinger trade-off plots provide a powerful visualization of these trade-offs by representing performance and robustness criteria explicitly [4]. These plots help to identify optimal PID parameters by highlighting the compromise between error minimization and robustness. Three key criteria are typically used in these analyses:

Performance Criteria: IE and IAE

The IE criterion measures the integral of the control error e , i.e., $IE = \int_0^\infty e(t) dt$. This metric captures the overall magnitude of the error but does not emphasize short-term or large deviations.

The IAE criterion improves upon the IE by emphasizing absolute deviations, which are often more relevant in practical systems,

$$IAE = \int_0^\infty |e(t)| dt. \quad (1)$$

The IAE is widely used as a performance metric because it penalizes persistent deviations more effectively than the IE . A lower IAE indicates better performance in terms of setpoint tracking and disturbance rejection. If $|IE|$ and IAE yield identical values, no overshoot occurs in the system.

The computation of the $I(A)E$ values is typically done for two experiments: i) a step response from d_i to y (disturbance rejection), and ii) a step response from u_r to y (setpoint tracking) in Figure 1.

Robustness Criterion: Maximum Sensitivity M_{st}

The robustness of a control system is commonly evaluated using the maximum sensitivity criterion, defined as:

$$M_{st} = \max_{\omega} (|S(j\omega)|, |T(j\omega)|) \quad (2)$$

$S(j\omega)$ is the sensitivity function, representing the system's response to disturbances and model uncertainties at different frequencies. $T(j\omega)$ is the complementary sensitivity function and represents the closed-loop frequency response for setpoint tracking, describing how the output y responds to changes in the setpoint r . A lower M_{st} corresponds to a more robust system that tolerates model variations better. A higher M_{st} suggests the system is less robust, as uncertainties are amplified more significantly.

The Trade-Offs in Garpinger Plots

The Garpinger trade-off plots visualize the interplay between performance (IAE) and robustness (M_{st}) (see Figure 3).

Unstable K_P - K_I parameterizations are colored grey (by the term unstable, a closed-loop system that is not internally stable [9] is meant). The M_{st} is plotted as red line, and the IAE as blue line. The IE value can be calculated by reading the controller gain on the ordinate axis: $IE = -1/K_I$. Where the $|IE|$ value coincides with the horizontal line of the IAE value, $|IE|$ and IAE have the same value. The IAE value does not change along the blue lines. Each point corresponds to a specific set of PI controller gains.

The optimal line (green in Figure 3), or Pareto front, is

the set of points where no further improvement in one criterion can be achieved without degrading the other and is plotted as green line. Designers can choose parameters along this (green) line depending on the specific application requirements.

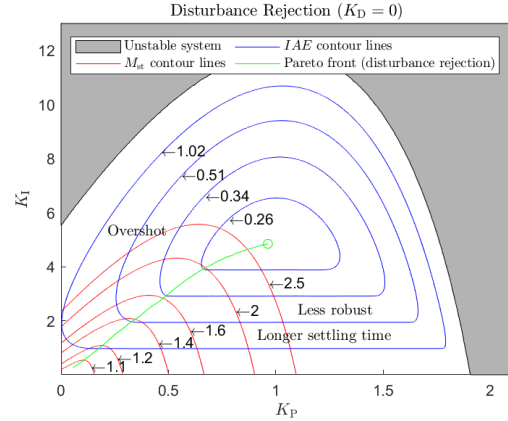


Fig. 3. Garpinger's trade off plot

Control gains to the left of the Pareto front lead to an overshoot (since $|IE| \neq IAE$), to the right of the Pareto front to a less robust system (since IAE is constant but M_{st} increases) and longer settling time (since IAE is constant but proportional controller K_P gain is increasing). By using these plots, it is possible to systematically select PID parameters that balance performance and robustness in a way that aligns with the specific needs of the control system. This approach not only improves system reliability but also provides a clear methodology for achieving optimal tuning.

The plots show how reducing the $|IE|$ or IAE (better performance) often comes at the expense of increased M_{st} (reduced robustness). Lowering the IAE typically requires higher PID gains, which may improve disturbance rejection or setpoint tracking but also makes the system more sensitive to noise and model uncertainties. Reducing M_{st} enhances robustness but may result in slower responses and larger errors. In the Garpinger's trade-off plots, an optimal line emerges, representing the best compromise between performance and robustness (see green line in Figure 3). *Remark:* This optimal line yields different sets of controller parameters when considering either disturbance rejection or setpoint tracking trade-off plots. In the remainder of the paper, the type of plot that is used is always mentioned. In some of the Garpinger plots, two lines describing the optimal controller parameters are shown. The green one is related to the disturbance rejection, whereas the magenta one is related to setpoint tracking.

D. AMIGO and Garpinger Method for PID Controller Tuning

The AMIGO method is a modern approach for tuning PID controllers, designed to overcome the limitations of traditional tuning methods such as Ziegler-Nichols method [7].

The method incorporates advanced design criteria to optimize disturbance rejection and minimize overshoot, while ensuring high robustness against model uncertainties and process variations.

The AMIGO method calculates the PID controller gains (K_P , K_I , K_D) based on the step response of the system.

1. **Process Gain (K):** The steady-state gain of the system, calculated as: $K = \Delta y / \Delta u$ where Δy is the change in the output and Δu is the change in the input of the plant.

2. **Time Delay (L):** The time it takes for the output to begin responding significantly after the input step.

3. **Time Constant (T):** The time required for the system to reach approximately 63 % of its steady-state response, minus the time delay.

Based on the above parameters, the AMIGO method calculates the PI gains as follows:

$$K_P = \frac{0.15}{K} + \left(0.35 - \frac{L \cdot T}{(L+T)^2}\right) \cdot \frac{T}{K \cdot L}, \quad (3)$$

$$T_I = 0.35L + \frac{13L \cdot T^2}{T^2 + 12L \cdot T + 7L^2}, \quad (4)$$

$$K_I = \frac{K_P}{T_I}. \quad (5)$$

If the D term is also to be considered, then there are also analogous formulas that interpret the controller gains somewhat more conservatively [1].

In certain cases, process constraints require adjustments to the controller gain. The Garpinger method addresses this by calculating the optimal integral gain (K_I) as a function of the proportional gain (K_P) based on parameter fitting derived from Garpinger's trade-off plots [4]

$$K_I = \frac{K_P + 0.1 K \cdot K_P^2}{0.3L + 0.7T}, \quad (6)$$

and is valid for $M_{st} < 1.6$.

III. RESEARCH QUESTION AND APPROACH

This research explores and compares the performance and robustness of two modern tuning methods for PID controllers in motor control: the AMIGO method and the Garpinger method. Both methods are evaluated in cascaded control systems for speed and position control, using Garpinger's Trade-Off Plots to balance performance metrics (e.g., fast disturbance rejection, precise position tracking) with robustness criteria (e.g., maximum sensitivity).

The evaluation was conducted on an Arduino-based DC motor system, replicating realistic operating conditions and noise to test controller robustness. The controller parameters were validated by observation of the response times of the speed and position.

IV. CASE STUDY

This section presents the design of a velocity control loop based on feedback control (IV-A to IV-C) and the implementation of a position control loop (IV-D).

A. System Identification and Model Extraction

To capture the behavior of the real system, an open-loop step response was performed using a DC motor from the Makeblock mBot Ranger kit. The motor is driven by a PWM signal ranging from -255 to +255 (12 V max). The measured outputs are angular velocity y (in rpm) and angular position y_r (in radians), obtained via the onboard encoder. The controller operates at a cycle time of 5 ms.

The step response (Figure 4) reveals significant noise in the speed signal, whereas the position response is relatively smooth.

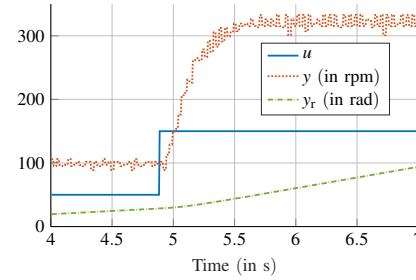


Fig. 4. Step response of DC motor

To address this, a first-order low-pass filter with a time constant $T = 0.05$ s was applied, chosen to be about four times faster than the system's natural cutoff. This reduces noise while preserving essential dynamics (see Figure 5).

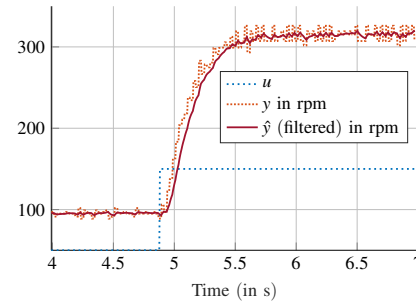


Fig. 5. Step response of DC motor with a filtered angular velocity with filter time constant $T = 0.05$ s

The filtered system is modeled as

$$P(s) = \frac{\hat{Y}(s)}{U(s)} = \frac{K}{T \cdot s + 1} \cdot e^{-L \cdot s} = \frac{2.222}{0.198 \cdot s + 1} \cdot e^{-0.087 \cdot s}. \quad (7)$$

This first-order lag plus time delay (FOLPD) model sufficiently captures the motor dynamics (see Figure 6) and serves as the basis for controller design.

The maximum velocity v_{max} and acceleration a_{max} , needed for trajectory generation, are estimated directly from steady-state K and maximum input u as

$$v_{max} \approx K \cdot u_{max} \cdot \frac{2\pi}{60} = 2.222 \cdot 255 \cdot \frac{2\pi}{60} \approx 50 \text{ rad/s}, \quad (8)$$

$$a_{max} \approx 2 \cdot u_{max} \cdot \frac{K}{T} \cdot \frac{2\pi}{60} \approx 600 \text{ rad/s}^2. \quad (9)$$

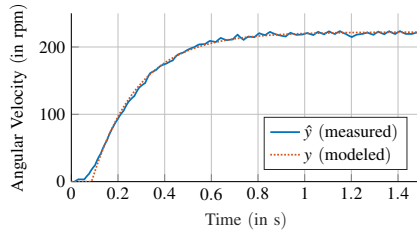


Fig. 6. Comparison of the real measurement data with the identified FTOD model

B. Garpinger's Trade-Off Plots for Performance and Robustness

To evaluate the controller design, trade-off plots based on the Garpinger method were created in MATLAB for both disturbance rejection (Figure 7) and setpoint tracking (Figure 8). These plots visualize the trade-off between performance (measured by IAE) and robustness (measured by maximum sensitivity M_{st}).

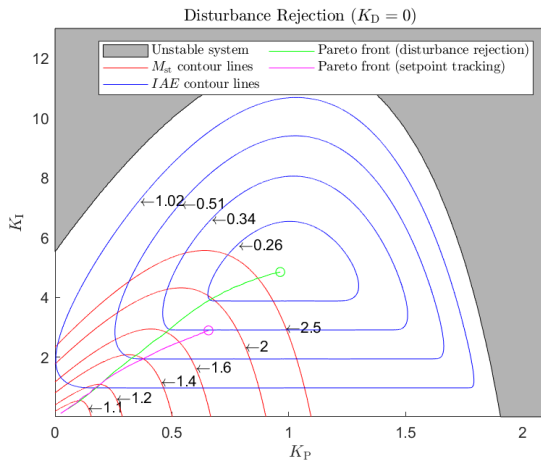


Fig. 7. Garpinger's trade-off plot for disturbance rejection

1. Pareto Front Analysis:

The green and magenta lines represent Pareto fronts for disturbance rejection and setpoint tracking, respectively. Each point on the front offers the best achievable performance for a given robustness level. For $M_{st} < 1.2$, the fronts diverge, revealing that both objectives cannot be optimized simultaneously.

2. Choice of Robustness Level:

A moderate robustness level of $M_{st} = 1.4$ is selected, as this value provides a reasonable balance between sensitivity to disturbances and robustness against uncertainties. For this M_{st} value, the controller gains can then be read off the Pareto Front for disturbance rejection (green line) at $K_P \approx 0.34$ and $K_I \approx 2.07$ (see Figure 7) and $K_P \approx 0.38$ and $K_I \approx 1.95$ for setpoint tracking (see Figure 8).

3. Impact of Optimization Choice:

Tuning for disturbance rejection leads to better rejection performance but results in overshoot during setpoint changes.

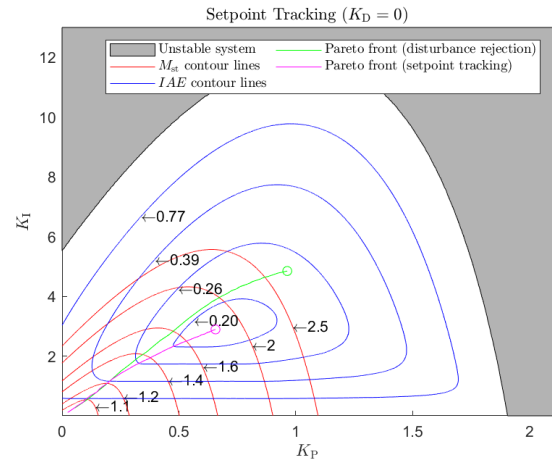


Fig. 8. Garpinger's trade-off plot for setpoint tracking

Conversely, tuning for setpoint tracking sacrifices disturbance suppression. This reflects the inherent conflict between these objectives in PID control.

4. Significance of the Integral Term (K_I):

The results consistently demonstrate the importance of including an integral term ($K_I > 0$) in the controller design. While the literature often suggests that a P-P cascaded control may be sufficient for many applications [8], the trade-off plots show that an integral term significantly enhances both disturbance rejection and setpoint tracking. By implementing an I -term, the controller achieves superior overall performance compared to purely proportional control strategies. In addition, the disturbance error would not be eliminated with a pure P control, as the plant does not have a pure I component.

By analyzing these plots, designers can select the most suitable controller gains for their specific application, balancing the trade-offs between disturbance rejection, setpoint tracking, and robustness. Furthermore, the findings clearly underscore the practical benefits of including an integral term in the control design.

C. AMIGO and Garpinger's tuning rules

The controller gains were initially analyzed using the Garpinger's trade-off plot for disturbance rejection. We now calculate the gains using the AMIGO rule-of-thumb method (see equations (3), (4) and (5)) and the parameters L , T and K from our model (see equation (7)), which yielded specific values for $K_P \approx 0.21$, $T_I \approx 0.18$, and $K_I \approx 1.17$.

These values were then compared to the trade-off plot for disturbance rejection (see Figure 9). The results showed that the gains obtained from the AMIGO method lie closely on the Pareto front for a robustness level of $M_{st} < 1.4$. This demonstrates that the AMIGO method provides optimal controller gains for a fixed robustness criterion of $M_{st} < 1.4$, ensuring a balance between disturbance rejection and robustness.

However, the AMIGO method has a notable limitation: it does not allow for independent adjustment of the pro-

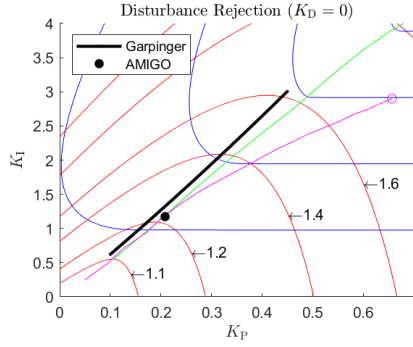


Fig. 9. AMIGO and Garpinger's rule-of-thumb highlighted in Garpinger's trade-off plot for disturbance rejection

portional gain K_P . To address this issue, the Garpinger method was applied. Different values below an M_{st} value of 1.6 were chosen manually for the proportional gain and the corresponding integral gain K_I is calculated using the Garpinger formula (6):

K_P	0.1	0.2	0.3	0.4	0.5
K_I	0.62	1.27	1.94	2.64	3.37

The new K_P - K_I pairs lie once again near the Pareto front of the Garpinger trade-off plot for disturbance rejection (see Figure 9). This result confirms that the Garpinger method not only accommodates adjustments to K_P but also ensures that the recalculated K_I maintains an optimal balance between performance and robustness for disturbance rejection. However, these pairs are located significantly to the left of the Pareto front for setpoint tracking (magenta line in Figure 9), particularly for higher K_P values. This indicates that controllers optimized for disturbance rejection are expected to exhibit considerable overshoot in response to setpoint changes. The results further emphasize the fundamental trade-off between disturbance rejection and setpoint tracking: the system can be optimized for one objective or the other, but not for both simultaneously. Consequently, the choice of controller parameters must carefully consider the specific performance priorities of the application, as optimizing for one criterion will inevitably compromise the other.

To validate the calculated controller parameters, the AMIGO and Garpinger methods (with $K_P = 0.4$ and $K_I \approx 2.64$) were tested on the real motor system. The controllers were implemented on the Arduino-based setup, and their performance was evaluated under practical conditions, focusing on setpoint tracking scenarios (see Figure 10).

The results revealed that the motor followed the desired velocity setpoint accurately, with a significant velocity overshoot for the Garpinger method. This behavior is consistent with the predictions from the trade-off plot for setpoint tracking, which shows that controller parameters optimized for disturbance rejection (with a robustness level of $M_{st} \leq 1.4$) tend to exhibit reduced performance in setpoint tracking. Specifically, the controller gains derived from the AMIGO and Garpinger methods prioritize disturbance re-

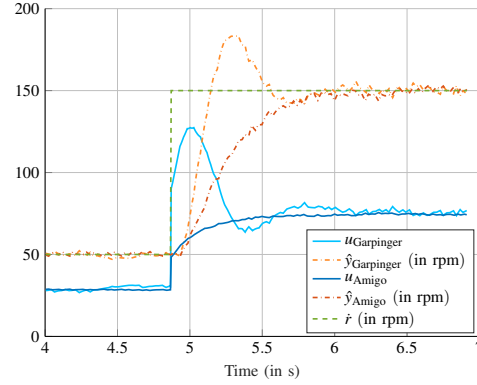


Fig. 10. Step response of motor with AMIGO and Garpinger's rule-of-thumb

jection, which can lead to overshoot during rapid setpoint changes.

D. Cascaded control

Experimental results highlight how the cascaded control structure performs in response to a trapezoidal velocity profile (see Figure 11). A significant deviation was observed between the reference and actual response of the system and the target values could not be accurately reached. Instead, load disturbances are optimally compensated for.

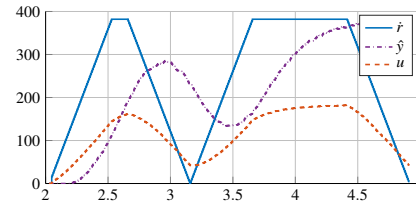


Fig. 11. Trapezoidal velocity profile response

In the final stage, a PID controller is implemented in the outer position loop. Unlike the inner loop, whose gains were determined via tuning rules, the outer loop gains are directly derived from the Garpinger trade-off plots to optimize tracking robustness.

The plant for the outer loop is defined as:

$$P_{out} = \frac{Y_r}{U_r} = \frac{PC}{1+PC} \cdot P_r, \quad (10)$$

where $P_r = 1/s$ represents the integrator that translates the velocity to a position signal.

A Garpinger trade-off plot is generated for setpoint tracking. Since the system has an integrating behavior, the plot visualizes the trade-off between the proportional gain (K_P) and the derivative gain (K_D), rather than the integral gain (K_I) used in the previous trade-off analyses. The disturbance still acts at the input of P and the noise filter was included in the calculation (compare equation (7)). The resulting Pareto front distinctly demonstrates the benefit of incorporating a

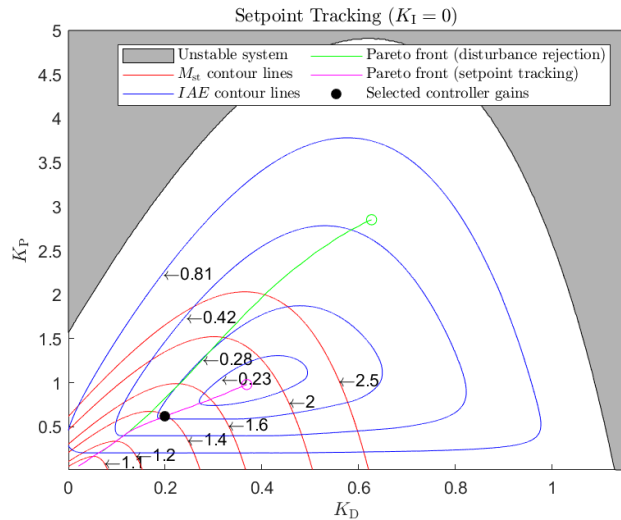


Fig. 12. Garpinger trade-off plot for setpoint tracking of the outer loop

derivative action to improve system performance (see Figure 12).

We again select an M_{st} value of 1.4 and now determine the controller gains from the trade-off plot with $K_p \approx 0.62$ and $K_D \approx 0.2$.

The complete control system is now validated using trapezoidal velocity trajectory tracking (see Figure 13). The system successfully follows the trapezoidal velocity profile, demonstrating that the controllers are properly tuned.

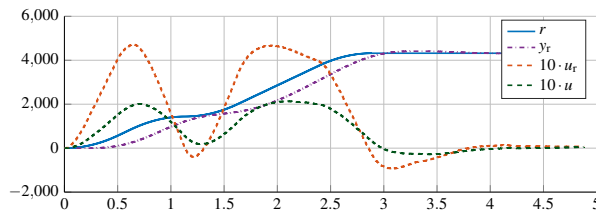


Fig. 13. Trapezoidal velocity profile response of the complete controlled system

The results confirm the importance of tuning the inner and outer loops differently to achieve optimal system performance:

- The **inner loop** should be primarily tuned for disturbance rejection, ensuring that speed fluctuations and external disturbances are suppressed effectively. The controller should have an I component for two reasons. Firstly, the trade-off plot shows that only then the smallest possible IAE value at a certain robustness requirement is met. Secondly, the I component in the inner loop is required to obtain zero steady-state control error.
- The **outer loop** should prioritize robustness and, if necessary, setpoint tracking, allowing the overall system to achieve smooth, accurate position control. However, since the setpoint tracking can also be achieved by

feedforward control [5], particular attention should be paid to robustness.

By synthesizing controller gains directly from the Garpinger Trade-Off Plots, the system achieves a well-balanced trade-off between disturbance rejection, robustness, and setpoint tracking. The combination of optimized PID tuning, and low-pass filtering for noise reduction ensures that the cascaded control system performs with high precision in real-world applications.

V. CONCLUSION

The tuning of PID controllers for DC drives is well established, yet a structured step-by-step approach that systematically considers robustness, setpoint tracking, and disturbance rejection is still lacking. This paper addresses this gap by consolidating existing methods, particularly the AMIGO and Garpinger approaches, and systematically applying them to DC motor control. Garpinger's trade-off plots are utilized to facilitate the targeted selection of optimal controller parameters. The performance and robustness of the controllers are experimentally validated on an Arduino-based motor system, demonstrating enhanced setpoint tracking. Unlike the inner loop, which is tuned using rule-of-thumb methods, the outer loop of the cascaded control system is directly synthesized using the trade-off plots, ensuring an optimal balance between robustness and tracking performance. The results highlight the practical relevance of this methodology for industrial applications requiring high precision and reliability.

REFERENCES

- [1] K. J. Åström and T. Hägglund, *Advanced PID control*. ISA-The Instrumentation, Systems and Automation Society, 2006.
- [2] N. Bacac, V. Slukic, M. Puškarić, B. Stih, E. Kamenar, and S. Zelenika, "Comparison of different dc motor positioning control algorithms," in *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2014, pp. 1654–1659.
- [3] L. Desborough and R. Miller, "Increasing customer value of industrial control performance monitoring-honeywell's experience," in *AICHE symposium series*, no. 326. New York: American Institute of Chemical Engineers; 1998, 2002, pp. 169–189.
- [4] O. Garpinger, T. Hägglund, and K. Åström, "Performance and robustness trade-offs in PID control," *Journal of Process Control*, vol. 24, 05 2014.
- [5] P. Lino, J. Königsmarkova, and G. Maione, "Feedback-feedforward position and speed control of dc motors by fractional-order PI^λ controllers," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 2584–2589.
- [6] N. Mohamed, F. Aymen, B. H. Mouna, and S. Lassaad, "Modeling and simulation of vector control for a permanent magnet synchronous motor in electric vehicle," in *2021 4th International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, 2021, pp. 1–5.
- [7] H. Panagopoulos, K. Åström, and T. Hägglund, "Design of PID controllers based on constrained optimisation," *Control Theory and Applications, IEE Proceedings* -, vol. 149, pp. 32 – 40, 02 2002.
- [8] D.-L. Sabău, "Advanced control techniques for cnc machines," Ph.D. dissertation, Technical University of Cluj-Napoca, 2022.
- [9] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control - Analysis and design*. John Wiley & Sons, 2001.

Towards Automated Handling and Sorting of Garments combining Visual Language Models and Convolutional Neural Networks

Serkan Ergun^{1†}, Tobias Mitterer^{1†} and Hubert Zangl^{1,2}

Abstract—Ambitious goals set by the European Union are aiming towards full recycle-ability of garments by 2030. According to the EU, 12 kg of garments are discarded by each citizen per year. In order to process such vast amounts of garments, automation of garment handling and recycling is unavoidable. Automated handling and sorting of such garments is a major challenge in the field of robotics. Current approaches specialize in one part of this challenging task. For sorting, current approaches use cameras and pre-trained networks with a dataset with a pre-defined set of classes. This paper presents an approach of using artificial intelligence (Convolutional Neural Network and Visual Language Models) to locate and separate garments from a pile and identifying and sorting them into dedicated containers. This combines the advantages of both neural network types, where convolutional neural networks are used for grasping (segmentation and corner detection) and visual language models are used for classification of garment types and to help the grasp prediction network in narrowing in on better grasp positions.

Index Terms—Garments, Sorting, Visual Language Models

I. INTRODUCTION

In recent years, the European Union has set out to combat the huge amount of textiles being discarded every year. To this purpose, a directive has been released stating to achieve full recycle-ability of garments by the year 2030 [1]. To be able to recycle garments, facilities need to sort disposed garment according to their type, material composition and color and detect their state of health regarding faults, unremovable stains or tears. Such facilities currently rely heavily on manual labor to accomplish those tasks. Given that each European Union citizen disposes of approximately 12 kg of clothing annually [1], this results in substantial quantities that require sorting. To be able to better handle this workload, robots in combination with artificial intelligence are a viable alternative. Such a sorting workflow can be split into multiple tasks, starting with retrieving a garment from a pile, inspecting it and sorting it according to pre-defined categories. The robot needs to be able to differentiate between the different garments in the pile to at least be able to pick one textile out of the pile for individual inspection, detect which type the garment belongs, perform an inspection of the garment and sort it into given containers. To be able to retrieve a garment from a pile, a first visual inspection is needed for

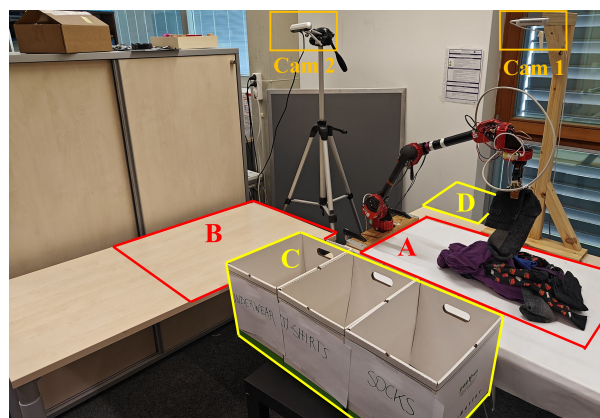


Fig. 1: Illustration of the garment handling and sorting scenario: Garments are picked from a pile (zone A) and manipulated to the inspection desk (zone B). Each garment piece is then sorted in the corresponding container (zone C) or discarded, if it does not meet sorting criteria (zone D). Two RGB-D cameras (Cam 1 and Cam 2) are used for grasp prediction and garment classification, respectively.

a preliminary distinction of different textiles and to be able to pick out a single object from the pile. To this purpose, different techniques in artificial intelligence can be used. We propose to use image segmentation and corner detection in conjunction, in a pre-trained Convolutional Neural Network (CNN) to be able to distinguish between different garments and to be able to detect first optimal grasp positions. After the textile has been moved from the pile for individual inspection, a class needs to be assigned to the item. For this purpose, different options are available. One option is to use a pre-trained neural network to try to sort the garment into a given set of classes [2]. This limits the operation, as in such a facility you cannot be sure what types and subtypes of garments are inside the piles that need to be sorted. We propose to use Visual Language Models (VLMs) to match a class to the inspected textile instead. Utilizing a VLM offers benefits, including eliminating the need for further training and providing autonomy from specific subsets of classes. An additional step is needed to semantically match all detected labels given by the VLM to the final list of classes needed for the specific use-case. VLMs can also be used for a first semantic state of health detection of the garment and to find best semantic grasp positions for each given class of garment. This helps the grasp prediction network in the next step to be able to find better grasp positions to place the object into the corresponding container. An overview on the complete

¹Serkan Ergun, Tobias Mitterer and Hubert Zangl are affiliated with the Department of Smart Systems Technologies, Sensors, Actuators and Modular Robotics Group, University of Klagenfurt, 9020 Klagenfurt am Wörthersee, Austria, serkan.ergun@aau.at, tobias.mitterer@aau.at, hubert.zangl@aau.at

²Hubert Zangl is also affiliated with the Ubiquitous Sensing Lab, University of Klagenfurt, 9020 Klagenfurt am Wörthersee, Austria

[†]These authors contributed equally.

handling and sorting set-up is shown in Fig. 1.

The main contributions of this paper are an evaluation of using VLMs for classification of textiles, finding optimal grasping positions, comparing between the performance of using a dedicated CNN and the more general VLM and demonstrating a use-case for this evaluation in a robotic gripper textile sorting context.

II. RELATED WORK

Recent advances in object detection and classification show a shift from pre-trained networks like CNNs, where a specific list of classes is given in the training and detected to a more general, semantic-based approach like in VLMs. One example of a CNN is Suchi et al. [3], where in the recorded dataset, Object Clutter Indoor Dataset (OCID), a given set of objects and their classes are defined and can be detected by the network. While such networks have good performance and require relatively less hardware, there are advantages to use VLMs for object detection and classification. These VLMs use Zero shot object detection to be able to detect classes by using semantics and free-text queries as input. First advances have been made by the Vision Transformer for Open-World Localization (OwlVit) network [4], in which image-text models are transferred to open-vocabulary object detection. Current advances in the field combine advantages of both methods. Examples are GroundVLP [5], which harnesses the visual grounding abilities from pre-trained image-text pair models and open-vocabulary object detection data to better detect and localize objects without dedicated training on those classes or DINO [6], which achieves good performance on the Common Objects in Context (COCO) dataset by incorporating improved de-noising techniques and anchor boxes. For robotics, the next step after detecting objects with a sensor, e.g. a camera, is to identify suitable positions to grasp the detected object. One example of finding such grasping candidates is an extension to the previously mentioned Suchi et al. [3], Ainetter et al. [7], where the OCID dataset has been used as base to annotate grasping positions on labeled data. The CNN combines grasp detection with dense, pixel-wise semantic segmentation and was tested with a parallel-plate gripper in [8] in combination with capacitive sensing in the gripper. A special case of robotic grasping is, if the objects to grasp have a special shape or material such that they can only be grasped properly at dedicated areas, e.g. a cup. One type of object like this are textiles and garments, as due to their size or given constraints by the task (such as robot assisted dressing) they have the need to be grasped at dedicated positions. One such use-case is if a garment needs to be visually inspected for faults or current state and therefore all parts need to be fully visible. Yamazaki [9] presents a CNN trained to detect optimal grasp points for cloth based on shape classification to be able to properly unfold a given textile lying on a table. Another example of textile grasping is Fu et al. [10], where a network is used to detect the state of the textile based on visible corners and decides on grasping points to unfold it. A detailed semantic description of each class of textiles

is given by the European Commission in [11]. As already shown for object detection earlier, VLMs can also be used to semantically detect the best location to grasp a specific type of object, thus increasing the number of successful grasps. LanGrasp [12] uses Large Language Models (LLMs) and VLMs to enable semantic one shot object grasping, where the LLM gives the part of the object which should be grasped and the VLM grounds that information in an image. Finally, a grasp planner plans and executes the grasp. Another example on VLMs being used for robot grasping is Huang et al. [13], with a focus on handover tasks of household objects between a human and a robot. The robot uses a combination of VLM and LLM to detect objects of given classes and semantically detect appropriate grasping parts of the objects. As a last step, a grounded VLM is used to segment and detect the grasping parts of the object. Our approach introduces the innovative use of VLMs in combination with a CNN in the environment of sorting textiles from a highly cluttered heap depending on given semantic classes.

III. EXPERIMENTAL SETUP

The proposed lab scale experimental setup is shown in Fig. 1. It consists of a single modular series elastic 6-DoF arm with a two-fingered cable gripper (type A-2085-06G) by HEBI Robotics [14]. The finger tips are custom made and flexible. Random garments are placed in a convoluted pile in the initial inspection workspace. The robot is being used to grasp and manipulate a single piece of garment from the pile (zone A in Fig. 1) and place it on a second table for garment type classification: underwear, shirts (including t-shirts and polo shirts) and socks (zone B). As the performance of the VLM is evaluated the garments are just slightly dragged over the table edge to perform an initial unfolding and no further flattening or optimal positioning of the garment is done. The robot then places the garment in the corresponding box in front of the table (zone C). Garments, which do not fall in one of these categories, or which are not identifiable are discarded at the back of the table for manual inspection (zone D).

Two Intel RealSense cameras (models D415i and D455f) are being used for capturing depth images for the grasp prediction from the pile (Cam 1) and the inspection table (Cam 2), respectively. The RGB stream of the D455f was also used for capturing the input for the garment classification.

The grasp prediction algorithm uses a CNN, which is based on the works of [7] and [15]. The CNN has been trained with a modified training set based on [3]. It has furthermore been used effectively in previous works, such as [8].

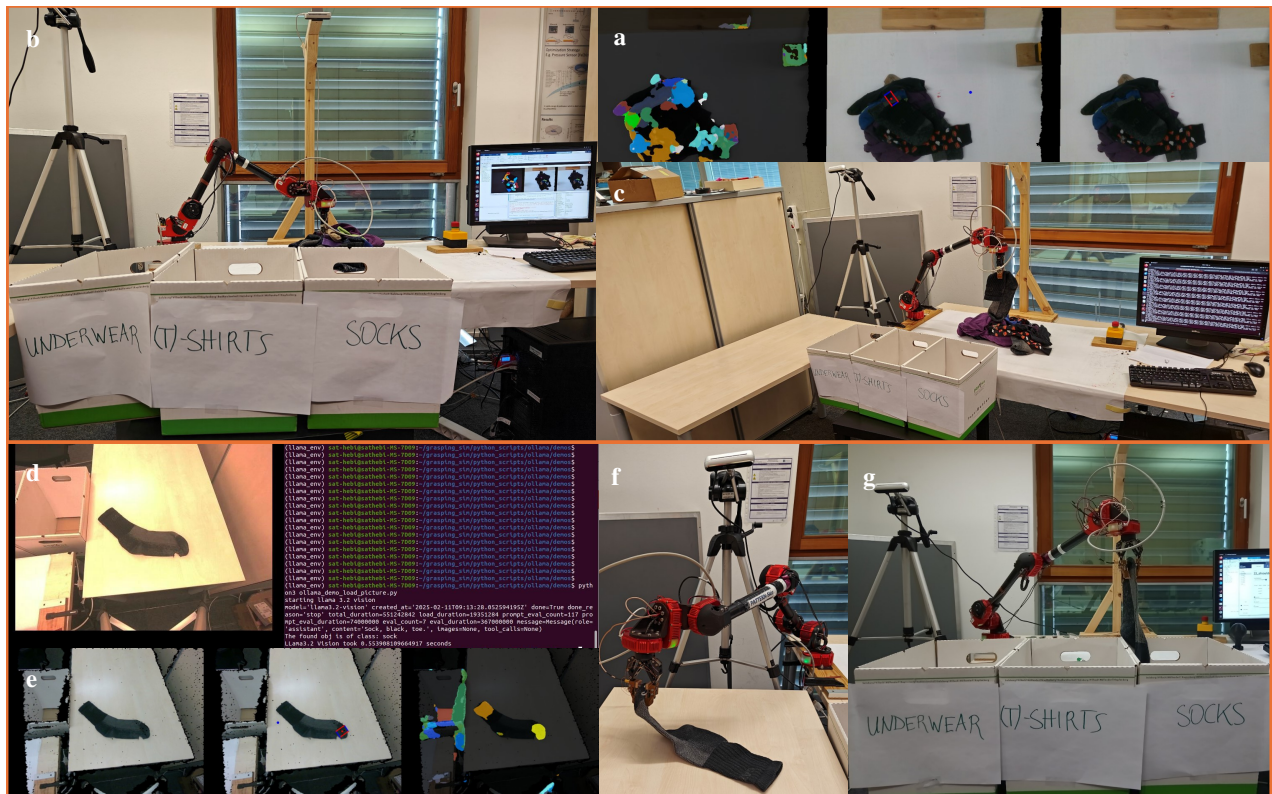


Fig. 2: Procedure of the experiment. A single piece of garment is identified by the grasp prediction algorithm (a), the robot then picks up the garment (b) and places it on the inspection table (c). The VLM identifies the type of garment and its color (d). The grasp prediction algorithm is then run again to identify the optimal grasp position (e). The garment is then picked up (f) and placed in the dedicated container (g). Unrecognized or not categorized are discarded at the back of the table for further manual inspection.

The garment classification is achieved by using VLMs, as they are able to match a semantic text input to a specific object in the picture. Dedicated VLMs like OWL-FIT [4] usually work by giving the network a semantic list of objects to find, which can lead to either a reduced number of objects detected or a long list of possible classes in the prompt. To this purpose a VLM with a broader language part in the model, namely Llama 3.2-Vision 11b from Meta Inc. [16] is used via the Ollama distribution [17]. This enables a more open prompt definition due to its bigger training data-set, where the types of garment to be found are not specified but the network has to match a garment type to the textile presented in the picture. For the proposed use-case, only the output classes of 'sock', 'underwear', 'shirt' and 'unknown' are used. A semantic assignment of the detected sub-types to these four classes, where each garment type which cannot be assigned to the three known classes is assigned to the 'unknown' class is done. This semantic matching process uses the textile classification specification given by the European Commission in [11].

Listing 1: Minimal Python code example for running llama3.2-vision with Ollama

```
1 import ollama
2
3 response = ollama.chat(
4     model='llama3.2-vision',
5     messages=[{"role": "system",
6                 "content": "You are an intelligent robotic
7                 arm."}, {"role": "user", 'content': 'What
8                 clothing item can you see? If your confidence
9                 is below 85 percent, classify the type as
10                unknown. Classify them in the classes: Shirt,
11                Sock, Underwear or unknown. Just specify type
12                of clothing and color. which part of the
13                garment makes the most sense to grasp? Name one
14                part. Make the answer very short and concise.
15                In three words. Your response is garment type,
16                color, grasp location. Omit any line breaks or
17                newlines.'}, {"images": [fullPathToImages]}
18     ]
19 )
```

The minimal code, necessary to run Llama3.2-vision with Ollama in Python3 is shown in Listing 1, where `fullPathToImages` is the location of the image on the hard drive. The grasp prediction(s) and the VLM are run sequentially on the processing unit, a lab PC with an 11th Gen Intel® Core™ i7-11700KF @ 3.60GHz × 16 CPU with 64 GB of DDR4 RAM. The Graphics Processing Unit (GPU)



Fig. 3: Sample pieces for garment classes: shirt, underwear and socks

used is the Nvidia GeForce RTX 3060 with 12GB V-RAM. The entire scenario is shown in Fig. 2. First, an ideal grasp pose is selected by the CNN (a). Then the robot moves to the selected pose and grasps the garment (b) and manipulates it to the inspection table (c). On the inspection table, the VLM is then run to identify the garment type, color and to return a semantic description of the best grasping position on this type of garment. The terminal output is shown alongside the input image in (d). The garment type output of the VLMs is then further refined and categorized semantically into the four given classes. Consequently, the CNN is then run again to identify the ideal grasp pose (e). The robot then picks up the garment once more (f) and places it in the correct bin (g). If the garment does not fall into one of the specified categories, it will be discarded at the back of the table for potential manual inspection.

IV. RESULTS

Sample pieces for each considered garment class (underwear, shirt, sock) are shown in Fig. 3. Further garments are shown in Table I. A subset of the garment samples (five to ten pieces) were placed untidily in zone "A" (see Figure 2c). The experiment described in section III was repeated until zone "A" was cleared. In total, more than 100 individual grasps were taken.

A. Validation of the approach

The grasp prediction algorithm was previously tested and validated in [8]. In some cases, the grasp prediction algorithm does not return an ideal grasp pose on every attempt. In such cases, another image is taken automatically and the grasp prediction is repeated until a valid grasp pose is found. During our experiments, no loss of garments during manipulation occurred. However, due to the small size ("arm reach") of the robot, only smaller garment pieces were considered in our experiments. Large and heavier pieces were only used for the validation of the VLM.

B. Validation of the VLM

To validate the accuracy of the VLM a larger set of garments were analyzed (zone "B"). In total, 122 images from trousers, jeans, jackets, sweatshirts were investigated alongside our pre-determined classes shirts, underwear and socks. Additionally, at random instances, non-garment type objects were presented to the camera (e.g. bottom entry in Table I).

Ten exemplary results of the garment classification algorithm are shown in Table I. Out of these 122 images, the garment type was correctly classified in 118 cases. The color of the garments were correctly identified in 112 cases, caused by inconsistent lighting conditions during the experiments. The color information was not used in our experiments but recorded alongside for future use, if the need arises to sort the garments not only by type but also color. An additional analysis was done on identifying semantic grasp positions for each type of garment, to optionally help the grasp prediction in narrowing in on better grasp positions for specific type of garments. In the testing data-set, non-textiles were included to observe the output if the prompt tells the network to identify a given textile, when no garment is presented. As can be seen in Table II for most cases the precision is greater than 80%. For the non-garment objects, they were classified correctly as 'unknown', but a non-ideal grasping position was returned.

V. SUMMARY AND OUTLOOK

This paper presented a method to utilize a combination of pre-trained CNN and VLM for automated handling and sorting of garments. Overall, the VLM is able to identify the garment type with a precision of 96.72%. The color of the garments is correctly identified in 91.80% of our experiments. The proposed sorting setup has the capability to scale in size and thus, productivity.

By using a serial arm manipulator with higher reach, larger garments can be handled. Furthermore, a larger arm reach allows more containers for sorting to be placed in the scene. The usage of a second arm manipulator for picking up inspected garments while the other one is grasping garments from the pile increases productivity. Alternatively, a conveyor belt type setup may also be considered using pushers to slide garments in the corresponding containers. A second processing unit with a dedicated GPU allows to run both grasp prediction algorithms in parallel together with the VLM to further increase productivity. The current work can be used to include multi-point grasp for improving the handling of bigger textiles like shirts or trousers and allowing all-round inspection of garments for visual defects.

ACKNOWLEDGMENT

This work has received funding from the "Austrian Research Promotion Agency" (FFG) within the AdapTex project under grant number 899044.

TABLE I: Exemplary results of the garment classification algorithm. The image on left is the input of the VLM. Type, Color, Suggested Grasp Pose and Class are returned by the VLM.

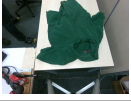
Image	Type	Color	Suggested Grasp Pose	Class
	Sock	Black	Heel	Sock
	Shirt	Green	Sleeve	Shirt
	Shirt	Blue	Collar	Shirt
	Boxers	Blue	Waistband	Underwear
	Shirt	Purple	Collar	Shirt
	Jeans	Blue	Waistband	Unknown
	Shirt	Blue	Cuff	Shirt
	Boxers	Blue	Waistband	Underwear
	Trousers	Grey	Waistband	Unknown
	Unknown	Blue	Heel	Unknown

TABLE II: Overall results of the garment classification. In total, 122 images of garments were captured. The VLM output was then manually examined. Accuracies for determining the correct class, color and potential grasp position are given for each garment type were recorded.

Type	Count	Class	Color	Grasp Position
Underwear	14	100%	85.71%	100%
Shirt	46	97.83%	95.65%	97.83%
Sock	36	97.22%	91.67%	97.22%
Unknown	26	92.31%	88.46%	53.85%
Total	122	96.72%	91.80%	88.52%

REFERENCES

- [1] Directorate-General for Environment, “COMMUNICATION FROM THE COMMISSION TO THE EUROPEAN PARLIAMENT, THE COUNCIL, THE EUROPEAN ECONOMIC AND SOCIAL COMMITTEE AND THE COMMITTEE OF THE REGIONS - EU Strategy for Sustainable and Circular Textiles,” 2022. [Online]. Available: https://environment.ec.europa.eu/publications/textiles-strategy_en
- [2] R. Tian, Z. Lv, Y. Fan, T. Wang, M. Sun, and Z. Xu, “Qualitative classification of waste garments for textile recycling based on machine vision and attention mechanisms,” *Waste Management*, vol. 183, pp. 74–86, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0956053X24002629>
- [3] M. Suchi, T. Patten, D. Fischinger, and M. Vincze, “Easylab: a semi-automatic pixel-wise object annotation tool for creating robotic rgb-d datasets,” in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6678–6684.
- [4] Minderer, Matthias and Gritsenko, Alexey and Stone, Austin and Neumann, Maxim and Weissenborn, Dirk and Dosovitskiy, Alexey and Mahendran, Aravindh and Arnab, Anurag and Deghani, Mostafa and Shen, Zhuoran and Wang, Xiao and Zhai, Xiaohua and Kipf, Thomas and Houlsby, Neil, “Simple Open-Vocabulary Object Detection,” in *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part X*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 728–755. [Online]. Available: https://doi.org/10.1007/978-3-031-20080-9_42
- [5] H. Shen, T. Zhao, M. Zhu, and J. Yin, “Groundvlp: harnessing zero-shot visual grounding from vision-language pre-training and open-vocabulary object detection,” in *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, ser. AAAI’24/IAAI’24/EAAI’24. AAAI Press, 2024. [Online]. Available: <https://doi.org/10.1609/aaai.v38i5.28278>
- [6] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. Ni, and H.-Y. Shum, “DINO: DETR with improved denoising anchor boxes for end-to-end object detection,” in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: <https://openreview.net/forum?id=3mRwyG5one>
- [7] S. Ainetter and F. Fraundorfer, “End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from rgb,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 452–13 458.
- [8] S. Ergun, T. Mitterer, S. Khan, N. Anandan, R. B. Mishra, J. Kosel, and H. Zangl, “Wireless capacitive tactile sensor arrays for sensitive/delicate robot grasping,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 777–10 784.
- [9] K. Yamazaki, “Selection of grasp points of cloth product on a table based on shape classification feature,” in *2017 IEEE International*

- Conference on Information and Automation (ICIA)*, 2017, pp. 136–141.
- [10] T. Fu, C. Li, J. Liu, F. Li, C. Wang, and R. Song, “FlingFlow: LLM-Driven Dynamic Strategies for Efficient Cloth Flattening,” *IEEE Robotics and Automation Letters*, vol. 9, no. 10, pp. 8714–8721, 2024.
 - [11] European Commission. (2025) Classifying textiles. [Online]. Available: <https://trade.ec.europa.eu/access-to-markets/en/content/classifying-textiles>
 - [12] R. Mirjalili, M. Krawez, S. Silenzi, Y. Blei, and W. Burgard, “LAN-grasp: An effective approach to semantic object grasping using large language models,” in *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024. [Online]. Available: <https://openreview.net/forum?id=SfHjWbfW02>
 - [13] J. Huang, C. Limberg, S. M. N. Arshad, Q. Zhang, and Q. Li, “Combining vlm and llm for enhanced semantic object perception in robotic handover tasks,” in *2024 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, 2024, pp. 135–140.
 - [14] Hebi Robotics Inc., “Hebi A20856G Data Sheet,” February 2025. [Online]. Available: <http://docs.hebi.us/resources/kits/datasheets/x-series/A-2085-06G.Datasheet.pdf>
 - [15] L. Porzi, S. Rota Bulò, A. Colovic, and P. Kotschieder, “Seamless scene segmentation,” in *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
 - [16] Meta Inc. (2025) Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. [Online]. Available: <https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/>
 - [17] Ollama. (2025) Ollama Website. [Online]. Available: <https://ollama.com/>

Towards Inclusive and Accessible Industrial Workstations by Shaping Safe and Adaptive Human-Robot Collaboration*

Mara Vukadinovic¹, Clara Fischer¹, Thomas Haspl¹, Bernhard Reiterer¹, and Michael Rathmair¹

Abstract—Human-robot collaboration combines the strengths of human workers with the capabilities of robots, creating opportunities to improve inclusion and accessibility in manufacturing environments. This study investigates the integration of adaptive workstations within human-robot systems to close gaps in safety and diversity in industrial settings. The presented design and safety framework incorporates workstations with adjustable heights, flexible tool positioning, and multimodal communication interfaces to accommodate workers with varying physical and cognitive abilities. Through a collaborative assembly use case, the study demonstrates how robots can handle repetitive and physically demanding tasks while human workers focus on skill-dependent activities. This approach improves task efficiency and fosters workforce inclusivity, providing a pathway for integrating individuals with disabilities into the primary labor market. The findings emphasize the need to shift towards adaptive, human-centered design to ensure equitable participation in industrial workplaces.

Index Terms—human-robot collaboration, robotic assistance, people with disabilities, inclusive workstations

I. INTRODUCTION

In Austria, approximately one in four individuals aged 15 to 89 living in private households — equivalent to around 1.9 million people — suffer from health-related limitations in managing daily activities [29]. The employment rate among people with disabilities is 52.8%, slightly above the EU average of 50.8%. However, only 14.9% of these individuals are employed in the regular labor market, as the majority of them work in specialized environments designed to support their needs [30]. To address this disparity, Austrian legislation mandates that companies with at least 25 employees hire one registered disabled person for every 25 employees. An individual is considered registered disabled if he or she has a degree of disability of at least 50%, as defined in § 2 of the Disability Employment Act [28]. According to § 3 of the BEinstG, disability is defined as a lasting physical, mental, psychological, or sensory impairment that is likely to hinder participation in the labor market for a period of more than six months. Despite these legislative measures, many companies do not meet these requirements. In 2023, only 23.9% of companies nationwide fulfilled this employment obligation. Consequently, 76.1% of enterprises are subject to compensation tax due to not meeting the mandatory employment quote [31].

*This research was conducted as part of the SAFEIVERSE project, funded by the KWF – Kärntner Wirtschaftsförderungs Fonds.

¹All authors are with JOANNEUM RESEARCH Forschungsgesellschaft mbH, Institute for Robotics and Flexible Production, Klagenfurt, Austria {First Name}.{Surname}@joanneum.at

Concurrently, demographic change leads to an aging population, decreasing birth rates, and shifts in the population structure. This development has led to an increase in retirement age and a growing shortage of skilled workers [7]. While companies in various sectors, such as industry and tourism, face a severe shortage of skilled workers, there is an untapped potential for individuals who have not yet been integrated into the primary labor market. The current situation underscores the need to find new ways to include people with disabilities in the workforce [15]. One approach to overcome this challenge are so-called sheltered workshops (SWs), which are integrative work organizations designed to meet the specific needs of people with disabilities. In Austria, SWs are structured as non-profit organizations that operate separately from the regular labor market and mainly offer simple repetitive tasks [21]. As a result, they establish a separate employment sector instead of fostering true inclusion. SWs tend to reinforce segregation, limiting opportunities for equal participation in the broader workforce [11].

A. Motivation and Problem Statement

Addressing the interconnected challenges of an aging population, a skilled labor shortage, and the underemployment of individuals with disabilities requires a comprehensive strategy. Promoting the inclusion of individuals with disabilities can help to mitigate the shortage of skilled labor by tapping into an underutilized talent pool [6]. Human-robot collaboration (HRC) not only leads to increased productivity, but also enhances participation. Robot systems can be designed to assist individuals with disabilities, enabling them to perform tasks that might otherwise be challenging. Moreover, robot applications can support older or disabled employees by handling physically demanding tasks, thus improving ergonomics and reducing monotony [11], [8].

In the perspective of robot safety, ISO 10218, as published in 2025, Part 1 describes the safety requirements for industrial robots [16], while Part 2 of this standard defines industrial robot applications and robot cells, including modes for safe HRC [17]. In a collaborative application, the state interaction can be reduced by avoiding collisions between humans and robots or restricting the transferred energy in intended or unintended contact. The second option can be implemented by limiting the application's power and force so that biomechanical limits by ISO/TS 15066:2016 are observed in contact. These thresholds and the associated assessment procedures have now been integrated into ISO 10218-2:2025 [17]. Biomechanical limits are the tolerated pressure and force transmitted to different parts of the human

body during contact with the robotic system. The values are based on studies that analyze the occurrence of pain onset (pressure) and minor injuries, such as bruises (force) in different anatomical regions [18]. However, these thresholds are predominantly based on average percentile values, which do not fully account for individual variations such as age, physical impairments, or gender-related anatomical and physiological differences. For example, differences in BMI, bone density, and pain tolerance between men and women can affect their sense of force and pressure loads. Similarly, older adults or people with disabilities may require adjusted safety parameters to mitigate risks of injury and ensure safe interaction [4], [3], [20].

Furthermore, the European Regulation (EU) 2023/1230 on machinery requires manufacturers to carry out comprehensive risk assessments, e.g. according to ISO 12100:2010 [1]. A main aspect of this process is the definition of machine limits, including operational and user restrictions based on specific physical requirements [9]. As a result, certain machines may not be approved for being operated by people of different sex, ages, or with physical disabilities.

B. Contribution

This paper investigates the integration of diversity and inclusion in the design of safe HRC. The contribution is provided by a systematic literature review, from which a framework for the design of adaptive and inclusive HRC is presented. The objective is to explore how safety standards and interaction protocols can be improved to accommodate differences such as age, sex, and physical impairments.

The structure of the paper is as follows: First, the state-of-the-art in inclusive HRC is analyzed. Then, our vision of the future of inclusive manufacturing is outlined, covering the concept and the description of an industrial application scenario. Subsequently, the evaluation assesses the inclusivity of the concept, identifies limitations, and suggests next steps. The paper concludes with a summary and directions for future work.

II. STATE-OF-THE-ART

A systematic literature review on inclusive and accessible HRC within industrial settings revealed three key thematic areas, which are summarized in the subsequent subsections.

A. Assistive Robotics

Assistive robotics has made considerable progress in recent years, particularly in health care and home care, where robots have demonstrated the potential to improve the autonomy and quality of life of people with disabilities. There are various ways in which physically assistive robots can help people with disabilities. The key research areas include assistance in navigation, feeding, and pick-and-place tasks. Although most studies include participants with disabilities, a significant proportion of summative evaluations involve only able-bodied individuals, highlighting the need for more inclusive and representative research methodologies. Additionally,

there is a lack of comprehensive studies exploring the real-world deployment of physically assistive robots, emphasizing the need for more in-context evaluations. Future research shall focus on tailoring these systems to individual user preferences and considering the broader social and regulatory factors that influence their adoption [27].

B. Robot Assistance on the Shopfloor

The literature shows that robot-assisted workplaces can support marginalized individuals in production settings by compensating cognitive and physical deficits. In the case of cognitive disabilities, the research by Kildal et al. demonstrates how collaborative robots can empower assembly workers with cognitive impairments by assisting them with complex tasks, reducing workload, and providing task-specific support [19]. Similarly, for physical limitations, Arboleda et al. highlight how HRC can support people with mobility impairments in the workplace, facilitating tasks that require physical strength or mobility, thus enhancing productivity and inclusion [2].

The AQUIAS project exemplifies how robots can support people with disabilities by helping them participate in modern manufacturing. This is achieved by assigning physically demanding tasks to robots while allowing individuals with disabilities to focus on other aspects, such as quality control. The project focuses on creating scenarios at the intersection of economic efficiency and participation in meaningful work. In the first pilot area, the production assistant "APAS" is implemented in an integration company where employees with severe disabilities perform assembly tasks. The second pilot area explores different models of HRC within an advanced manufacturing setting. The findings show that although close HRC can improve efficiency, it also presents challenges such as safety concerns, ergonomic load, and limited robot processing speed. The prototype developed addressed these concerns by incorporating height-adjustable tables for accessibility, a laser-based safety system to protect workers, and an integrated learning system to support employees with disabilities [22].

Another related project, IIDEA, focuses on promoting the inclusion and integration of people with severe disabilities into the primary labor market through collaborative robotics. Unlike traditional models that often relegate disabled workers to isolated tasks or sheltered workshops, IIDEA emphasizes human-centered, adaptive work environments at the core of Industry 4.0. The project aims to bridge the gap by offering training, modular robotic workstations, and mobile demonstration units to promote awareness and adoption. By customizing robot assistance to individual capabilities and fostering a broad network of stakeholders, including industry, advocacy groups, and training institutions, IIDEA seeks to establish inclusive employment opportunities [24].

C. Design Aspects of Inclusive Human-Robot Collaboration

Key aspects in designing industrial HRC include safety, efficiency, ergonomics, interaction, and acceptance [14], [23].

Studies indicate positive acceptance of collaborative industrial robots among people with disabilities, further emphasizing their role in fostering inclusive work environments [11].

The literature highlights capability-based approaches that form the basis for inclusive and collaborative work environments. Several tools and methodologies have emerged to optimize the allocation of tasks between humans and robots in inclusive environments and ensure that people with disabilities receive the necessary support. One such tool is IMBA (Integration of People with Disabilities into the Working Life), developed by the German Ministry of Health and Social Security. It serves as a method for comparing the requirements of workplace tasks with human capabilities and documenting both. However, IMBA has its limitations in modeling dynamic workflows. Specifically, it does not track changes in workload within workflows, which is essential for adaptive task allocation in HRC environments [13]. To address these limitations, RAMB (Robotic Assistance for Manufacturing Including People with Disabilities) was introduced, which analyzes specific process steps where individuals with disabilities may require personalized assistance. This is achieved by combining the decomposition of the process based on MTM (Method-Time Measurement) and IMBA, allowing a uniform evaluation of the process requirements that can be compared with the capability profile [32]. Despite the usefulness of IMBA and RAMB, these tools struggle to adapt to dynamic workflows. Mandischer et al. proposed a two-stage reasoning approach for adaptive task allocation in HRC to overcome these limitations. This system assesses the capabilities of a worker using an ontology-based methodology that distinguishes between factors that change quickly (e.g. fatigue) and others that change slower and have more gradual effects (e.g. worsening of a disease) [26].

Moreover, to ensure appropriate support for people with disabilities, the selection of input and output devices is essential. Weidemann et al. present an approach for selecting suitable devices based on a person's specific disabilities and the demands of the work process. For example, a person experiencing tremors after a stroke, with limited mobility in one hand, may benefit from hand or foot buttons as input devices that require minimal fine motor control [33].

III. THE FUTURE OF INCLUSIVE MANUFACTURING

The literature indicates that, while initial approaches have been proposed to address the challenges discussed in previous sections, their potential can be significantly improved by integrating adaptive workplace design concepts [25]. However, a considerable gap persists in the development of diversity-oriented safety strategies within production environments. Research highlights that diversity in robotics is still underdeveloped in workplaces and its implementation is often overlooked or not considered sufficiently [12].

The achievement of inclusive manufacturing requires a concerted effort to develop and integrate diversity-oriented safety strategies. This requires human-centered, safe, and technology-supported environments designed to be adaptive and autonomous. In particular, workspaces should be tailored

to the capabilities of the individual workers to enable them to perform tasks efficiently and safely while considering physical and cognitive differences. Our vision is a labor market that maximizes the usage of human potential by moving from a user-driven interaction paradigm to one in which systems and work environments dynamically adapt to human capabilities and enable seamless HRC. This includes designing workstations with appropriate reach and movement areas, performing ergonomic evaluations specific to the individual, and ensuring universal accessibility to safety features, such as the emergency stop button. In addition, integrating auditory, visual, and mechanical warning signals will improve accessibility and provide further support for people with different sensory requirements. Furthermore, a comprehensive risk assessment that accounts for individual variations in risk perception will be crucial to minimizing potential hazards and ensuring a safe and inclusive work environment.

A. Concept

A comprehensive design and safety framework is fundamental to creating inclusive manufacturing environments. Our strategy leverages advanced technologies and robot-assisted systems, focusing on HRC to establish adaptable workspaces. The core principle of this concept prioritizes incorporating diverse user groups to ensure that individual abilities are accommodated rather than relying on generic solutions. This is accomplished through a preliminary assessment of workers' skill profiles to identify competencies rather than limitations. For evaluating the working conditions in human-robot workplaces, the previously described RAMB tool is applied [32]. The evaluation considers factors such as body posture, body movement, sensory capabilities, and complex characteristics to assess the worker's skills. This process also involves analyzing job demands, workflow specifications, and task-specific constraints. Once the comparison is complete, tasks are assigned by distinguishing between those more suitable for human workers and those that robots can perform to assist. This allocation ensures that tasks are assigned in a way that optimizes efficiency and inclusivity as much as possible.

Following task allocation, process planning incorporates our design and safety framework, which extends traditional process optimization by embedding aspects of workplace and process design, risk assessment, and human inclusion.

First, we incorporate flexible workplace configurations that can be adjusted to different physical and cognitive needs, ensuring that workstations are ergonomically optimized for all users. This includes adaptable work surfaces, customizable tool positions, and universally accessible emergency controls.

Second, we aim to extend the risk assessment approach, as scripted in ISO 12100:2010, to have a more human-centered focus. By no means, we intend to replace the normative approach, but we rather add parameters to be able to consider the diversity of potential users already during the process of risk assessment. This is supposed to result in a more sensitive process with respect to the diversity of human

workers in an industrial workplace. In particular, we want to adapt two steps within the process chain of risk assessment, the identification of hazardous situations and the estimation of related risks. We expect that an additional parameter that considers the skill profiles of various workers leads to a more granular and more expressive risk assessment.

Third, we improve human-robot interaction by implementing intuitive interfaces that support multimodal communication, including voice, gesture, and touch-based input. This ensures that users with varying abilities can interact with robot systems in a way that suits their needs.

In addition, tablet-based guidance can be incorporated to provide accessible work instructions. These instructions could include audio guides and visual aids to ensure that individuals with varying cognitive abilities can easily understand and follow tasks.

Finally, our framework prioritizes barrier-free access and inclusive design principles by universal safety measures such as multi-sensory warning signals and robots with force and speed limitations tailored to individual risk profiles.

B. Use Case

To illustrate our approach, we present an assembly use case from series production that showcases the seamless collaboration between humans and robots. The process begins with a Universal Robots UR5 manipulator, which autonomously retrieves part A from its designated holder and positions it for the worker. The worker's task is to tighten a screw in the opening on the right-hand side of part A. After the worker completes this step, the robot sets part A down and retrieves a new unassembled part A, placing it in front of the worker for the same screwing task. Once the worker has assembled both part A pieces, the robot picks up part B and positions it in front of the worker. The worker's final task is to attach the two finished parts A to the left and right sides of part B, completing the assembly. This workflow illustrates a balanced division of labor, where the robot handles repetitive, precise tasks while the human worker performs the assembly steps that require more dexterity.

Figure 1 shows the robot and the collaborative assembly workstation. The most important adaptive components of the workstation are highlighted in green. Firstly, the autonomous height-adjustable table enables ergonomic adaptation to the physical requirements of the operator. In addition, the positions of the robot within the workstation can be adjusted to suit the operator's reach and ensure optimal interaction. This customization also includes the positioning of the box of screws, which is designed to be accessible to all operators, including left- and right-handed and one-armed operators, ensuring ease of use and involvement. Furthermore, the proposed system incorporates adaptive human-robot interaction by detecting when the operator is fully positioned at the workstation and ready to begin the task. The robot remains in standby mode until the operator arrives and confirms readiness to proceed. Depending on the operator's information processing needs, the system uses various signaling mechanisms, including visual, audible, and haptic

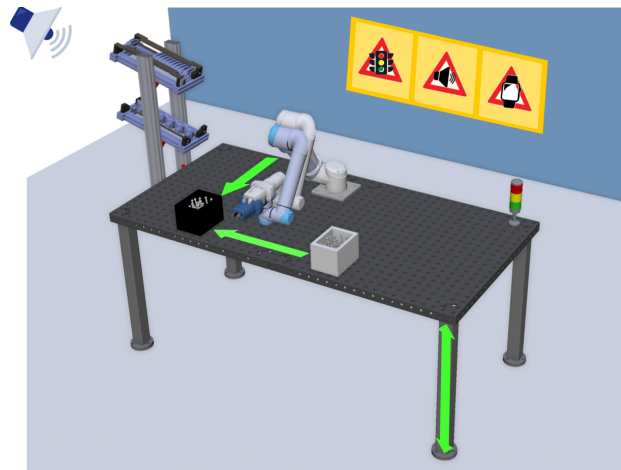


Fig. 1. Human-robot workstation with adaptive components, including ergonomic adjustments, flexible robot positioning, and multimodal signaling

signals. Visual indicators are provided by a light tower on the worktable, which uses color-coded signals to display system status and warn of potential hazards. In contrast, audible alerts provide immediate warnings via a loudspeaker in the work area. Haptic signals are transmitted through a smartwatch worn by the operator. This watch warns the user in dangerous situations through vibrations, for instance when a mobile robot is approaching.

The comparison between a standing human and an individual seated in a wheelchair is presented in Figure 2. In the right image, the table is lowered to accommodate the seated operator, ensuring ergonomic accessibility. In addition, the robot's end effector is positioned closer to the wheelchair user, optimizing reach and interaction. These adjustments demonstrate the adaptability of the workstation in supporting both standing and seated operators.

IV. DISCUSSION

The use case presented illustrates how adaptive workstations can create inclusive environments through HRC. It shows an assembly process in which robots assist in completing repetitive and physically demanding tasks while human workers focus on skill-dependent assembly steps. Several adaptive modifications have been designed, such as adjustable heights, flexible tool positioning, and multimodal communication interfaces. These adjustments ensure better ergonomics, accessibility, and usability for workers with varying physical capabilities. In addition, tablet-based instructions can be incorporated to support physical and cognitive accessibility. These instructions provide clear and tailored guidance to workers, enhancing their ability to perform tasks independently, regardless of mental challenges.

It has to be mentioned that these are relatively small-scale modifications. The illustrated use case does not allow for fundamental changes, such as modifying tools, significantly altering workflows, or introducing fully customized task assignments. These limitations highlight that, although

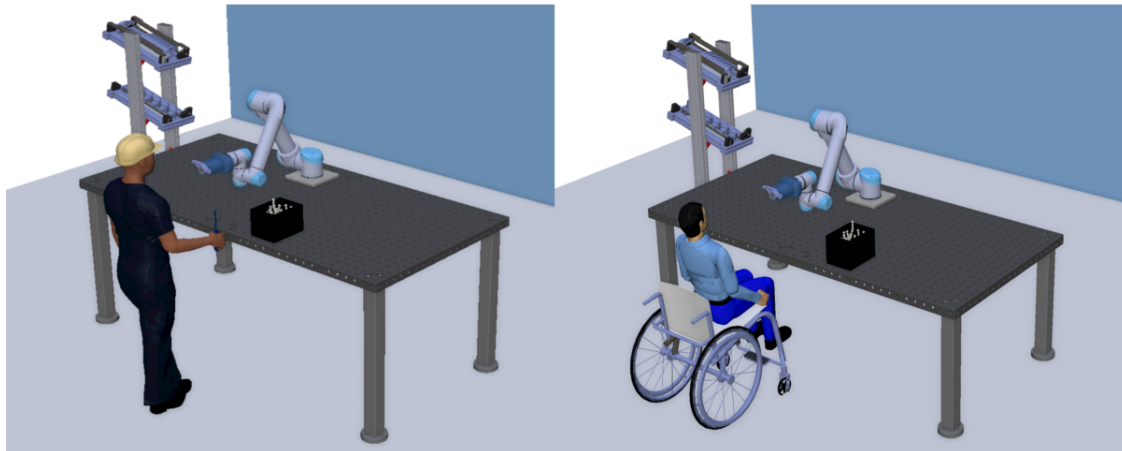


Fig. 2. Comparison of workstation adaptations for standing and wheelchair operators

the approach improves adaptability, it does not yet support comprehensive transformation for highly diverse work environments. Despite these constraints, small adaptations can still have a significant impact. Even minor modifications, such as adjusting the height of the workstation, optimizing component placement, and providing multiple modes of interaction, contribute to greater inclusivity and worker well-being. For instance, an adaptive arrangement of workstation components could improve accessibility for left-handed workers, thereby increasing comfort and overall job satisfaction. These adjustments facilitate a more accessible and efficient workplace without requiring a complete overhaul of existing systems. In manufacturing, where workers are often exposed to cognitive and physical overload, awkward postures, and repetitive tasks, such small adjustments can effectively reduce strain and improve productivity [5], [10].

These considerations underscore the role of adaptive workplaces in fostering inclusion while maintaining economic efficiency. As global labor market competition intensifies, companies are forced to implement flexible, efficient, and sustainable workstations. Organizations must decide whether to implement highly personalized workstations for each employee or develop universally adaptive environments that can accommodate a wide range of needs. A flexible and adaptive system can ensure that workers' abilities align with the demands of their tasks without requiring extensive modifications, thus increasing efficiency and reducing costs.

Another challenge is the broader implementation of adaptive workstations that promote social awareness. Many industries still lack a clear understanding of the benefits of inclusive and adaptable work environments. Public and corporate awareness must be raised through education and advocacy to encourage the adoption and investment in such workplaces. Highlighting long-term advantages, such as improved employee well-being, increased productivity, and the cultivation of a more inclusive culture, can help organizations recognize the value of creating environments that accommodate diverse needs. Thus, a key impact of our presented concept is its

ability to drive greater societal awareness and recognition of the importance of inclusive work environments.

To ensure the effectiveness of adaptive work environments, systematic evaluation is necessary. This includes evaluating safety features, worker satisfaction, productivity levels, and economic impacts. In addition, it is crucial to overcome challenges such as organizational resistance, technical limitations, and financial constraints. The current framework would benefit from extensive user testing with a diverse range of participants, particularly individuals with different disabilities. Such an approach would provide deeper insights into usability challenges and enable data-driven improvements to enhance accessibility and functionality. Integrating a wider spectrum of user experiences can optimize the framework to ensure a truly inclusive and effective design.

V. SUMMARY AND OUTLOOK

This paper addresses the gap in manufacturing environments, where safety standards and workplace designs often fail to consider diversity and inclusivity. By integrating adaptive workstations and HRC, this approach aims to create more inclusive and accessible environments. Ergonomic modifications, such as height-adjustable workstations, flexible tool positioning, and multimodal communication interfaces, enhance usability for workers with diverse needs. In addition, tablet-based instructions offer structured, tailored guidance, supporting physical and cognitive accessibility.

To validate and refine such adaptations, the next steps will include user testing on a physical workstation with a diverse group of participants, particularly people with disabilities. This process will involve direct observations and interviews to assess usability, physical and cognitive workload, and trust in the system. The results are used for further improvements, ensuring that future workstations are more inclusive, functional, and adaptable to the diverse needs of employees. Furthermore, by developing a physical demonstrator, this research aims to raise awareness of the potential of adaptive workstations for industrial companies and showcase their

benefits for inclusive labor market integration.

ACKNOWLEDGMENT

We would like to thank Stefanie Jeroutschitsch for proof-reading.

REFERENCES

- [1] “Regulation (EU) 2023/1230 of the european parliament and of the council of 14 june 2023 on machinery and repealing directive 2006/42/EC of the european parliament and of the council and council directive 73/361/EEC (text with EEA relevance),” legislative Body: CONSIL, EP. [Online]. Available: <http://data.europa.eu/eli/reg/2023/1230/oj/eng>
- [2] S. A. Arboleda, M. Pascher, Y. Lakhnati, and J. Gerken, “Understanding human-robot collaboration for people with mobility impairments at the workplace, a thematic analysis,” in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2020, pp. 561–566.
- [3] R. Behrens, G. Pliske, S. Piatek, F. Walcher, and N. Elkmann, “A statistical model to predict the occurrence of blunt impact injuries on the human hand-arm system,” *Journal of biomechanics*, vol. 151, p. 111517, 2023.
- [4] R. Behrens, G. Pliske, M. Umbreit, S. Piatek, F. Walcher, and N. Elkmann, “A statistical model to determine biomechanical limits for physically safe interactions with collaborative robots,” *Frontiers in Robotics and AI*, vol. 8, p. 667818, 2022.
- [5] F. BEUß, J. Sender, and W. Flügge, “Ergonomics simulation in aircraft manufacturing—methods and potentials,” *Procedia CIRP*, vol. 81, pp. 742–746, 2019.
- [6] J. Bickenbach, “The world report on disability,” *Disability & Society*, vol. 26, no. 5, pp. 655–658, 2011.
- [7] D. E. Bloom, D. Canning, and A. Lubet, “Global population aging: Facts, challenges, solutions & perspectives,” *Daedalus*, vol. 144, no. 2, pp. 80–92, 2015.
- [8] A. Bonello, E. Francalanza, and P. Refalo, “Smart and sustainable human-centred workstations for operators with disability in the age of industry 5.0: A systematic review,” *Sustainability*, vol. 16, no. 1, p. 281, 2024.
- [9] DIN EN ISO 12100:2010, “Safety of machinery – general principles for design – risk assessment and risk reduction,” March 2011.
- [10] X. Ding, Z. Guan, N. Liu, M. Bi, F. Ji, H. Wang, X. Zhang, B. Liu, D. Niu, T. Lan, *et al.*, “Prevalence and risk factors of work-related musculoskeletal disorders among emerging manufacturing workers in beijing, china,” *Frontiers in Medicine*, vol. 10, p. 1289046, 2023.
- [11] S. Drolshagen, M. Pfingsthorn, P. Gliesche, and A. Hein, “Acceptance of industrial collaborative robots by people with disabilities in sheltered workshops,” *Frontiers in Robotics and AI*, vol. 7, p. 541741, 2021.
- [12] C. Fischer, F. Gregshammer, M. Steiner, M. Neuhold, and S. Schlund, “Personalized safety: Considering the worker’s anthropometry in safety evaluation of human-robot collaboration,” in *European Robotics Forum*. Springer, 2024, pp. 286–291.
- [13] A. Glatz and H. Schian, “Imba-integration von menschen mit behinderungen in die arbeitswelt,” *Diagnostische Verfahren in der Rehabilitation*, pp. 368–371, 2007.
- [14] L. Gualtieri, E. Rauch, R. Vidoni, and D. T. Matt, “Safety, ergonomics and efficiency in human-robot collaborative assembly: design guidelines and requirements,” *Procedia CIRP*, vol. 91, pp. 367–372, 2020.
- [15] A. Houtenville and V. Kalargyrou, “People with disabilities: Employers’ perspectives on recruitment practices, strategies, and challenges in leisure and hospitality,” *Cornell Hospitality Quarterly*, vol. 53, no. 1, pp. 40–52, 2012.
- [16] ISO 10218-1:2025, “Robotics — safety requirements – part 1: Industrial robots.”
- [17] ISO 10218-2:2025, “Robotics — safety requirements – part 2: Industrial robot applications and robot cells.”
- [18] ISO/TS 15066:2016, “Robots and robotic devices: Collaborative robots,” 2017.
- [19] J. Kildal, M. Martín, I. Ipiña, and I. Maurtua, “Empowering assembly workers with cognitive disabilities by working with collaborative robots: a study to capture design requirements,” *Procedia CIRP*, vol. 81, pp. 797–802, 2019.
- [20] R. J. Kirschner, C. M. Micheler, Y. Zhou, S. Siegner, M. Hamad, C. Glowalla, J. Neumann, N. Rajaei, R. Burgkart, and S. Haddadin, “Towards safe robot use with edged or pointed objects: A surrogate study assembling a human hand injury protection database,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 680–12 687.
- [21] M. G. Kradischnig, M. P. Nausner, M. M. N. Quinz, and F. Wolfmayr, *Studie Integrative Betriebe 2020+: Endbericht*. Wien: Bundesministerium für Soziales, Gesundheit, Pflege und Konsumentenschutz (BMSGPK), 2020, druck: BMSGPK. [Online]. Available: <https://www.sozialministerium.at/>
- [22] D. Kremer and S. Hermann, “Robotik für menschen mit behinderung,” in *Bericht zur Online-Befragung und zum Praxis-Workshop des Projekts AQUIAS. Online verfügbar unter http://publica.fraunhofer.de/dokumente/N-603246.html*, 2020.
- [23] E. Lutin, S. A. Elprama, J. Cornelis, P. Leconte, B. Van Doninck, M. Witters, W. De Raedt, and A. Jacobs, “Pilot study on the relationship between acceptance of collaborative robots and stress,” *International Journal of Social Robotics*, vol. 16, no. 6, pp. 1475–1488, 2024.
- [24] M. Hüsing, C. Weidemann, S. Keunecke, E. Hüsing, C. Jansen and R. Youness-Sinaky, “IDEA – Inclusion and Integration through Cobots in the First Labor Market,” <https://www.iidea.rwth-aachen.de/cms/badmp/idea/>, 2025, training project funded by the Compensation Fund of the Federal Ministry of Labor and Social Affairs (BMAS), Germany.
- [25] A. A. Malik, “Future of industrial assembly: Intelligent reconfigurable & repurposable adaptive assembly (irraa),” *International Journal on Interactive Design and Manufacturing (IJIDeM)*, pp. 1–16, 2025.
- [26] N. Mandischer, M. Gürtler, C. Weidemann, E. Hüsing, S.-O. Bezrucav, D. Gossen, V. Brünjes, M. Hüsing, and B. Corves, “Toward adaptive human–robot collaboration for the inclusion of people with disabilities in manual labor tasks,” *electronics*, vol. 12, no. 5, p. 1118, 2023.
- [27] A. Nanavati, V. Ranganeni, and M. Cakmak, “Physically assistive robots: A systematic review of mobile and manipulator robots that physically assist people with disabilities,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 7, 2023.
- [28] Republik Österreich, “Behinderteneinstellungsgesetz (beinstg),” 2024, [Accessed 29-01-2025]. [Online]. Available: <https://www.ris.bka.gv.at>
- [29] V. Schuller, N. Mlinarević, and J. Klimont, *Menschen mit Behinderungen in Österreich I: Erhebungsübergreifende Datenauswertung aktueller Befragungen anhand des GALI-Indikators zu gesundheitsbedingten Einschränkungen bei Alltagsaktivitäten als Stellvertretervariable für Behinderung*. Wien: Statistik Austria im Auftrag vom Bundesministerium für Soziales, Gesundheit, Pflege und Konsumentenschutz (BMSGPK), 2024, stand: 23. Oktober 2024. [Online]. Available: <https://www.statistik.at/>
- [30] J. Schuster, F. Foissner, V. Schuller, and J. Klimont, *Menschen mit Behinderungen in Österreich III: Bildung, Erwerbstätigkeit und institutionelles Wohnen von Menschen mit registrierter Behinderung 2022*. Wien: Statistik Austria im Auftrag vom Bundesministerium für Soziales, Gesundheit, Pflege und Konsumentenschutz (BMSGPK), 2025, stand: 13. Februar 2025. [Online]. Available: <https://www.statistik.at/>
- [31] Sozialministeriumservice, “Geschäftsbericht 2023: Behinderung und arbeitswelt, gleichstellung und barrierefreiheit, pflegeunterstützungen, renten und entschädigungen, gesellschaftliche inklusion,” Wien, 2024, medieninhaber:in und Herausgeber:in: Bundesamt für Soziales und Behindertenwesen, Babenbergerstraße 5, 1010 Wien. Verlagsort: Wien. [Online]. Available: <https://www.sozialministeriumservice.at/>
- [32] C. Weidemann, E. Hüsing, Y. Freischlad, N. Mandischer, B. Corves, and M. Hüsing, “Ramb: validation of a software tool for determining robotic assistance for people with disabilities in first labor market manufacturing applications,” in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2022, pp. 2269–2274.
- [33] C. Weidemann, N. Mandischer, and B. Corves, “Matching input and output devices and physical disabilities for human-robot workstations,” in *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2024, pp. 972–979.

An Adaptable Multi-Robot Support System for Disaster Response*

Laurent Frering¹ and Gerald Steinbauer-Wagner¹

Abstract—In the recent years, many use-cases have been found for robots in disaster response operations, and many functionalities have been developed for those robots. But in order to facilitate the use of those robots in real operations, their usage have to be integrated at the mission level. In this work, we present our architecture for a multi-robot support system for disaster response operations. The proposed system's goal is to integrate agent-oriented programming for high-level decision-making with arbitrary robot platforms, refining goals into executable robot skills that are monitored and reasoned on. We focus on the software architecture and implementation details and provide details on the system capabilities and on the technologies used, and we outline the process for extending and adapting the proposed architecture to new projects. We discuss the different use-cases where the proposed system was deployed, and distribute its current open-source implementation: <https://gitlab.tugraz.at/D214D39B6CEB7ECC/mrss>

Index Terms—Software Architecture, Multi-Robot System, Disaster Response

I. INTRODUCTION

The use of heterogeneous robot teams for disaster response is gaining traction, with multiple recent experiments and deployments in different settings and with different robot types [1], including the use of Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs) in disaster response scenarios.

We previously proposed a generic architecture for a multi-robot support system aimed at providing first responders with a centralized situational picture obtained by a human-robot team comprising interactive goal-driven autonomous robots [2]. We deployed and tested this system during a field experiment simulating a firefighting operation in mountainous terrain, and gathered results and feedback from participating firefighters. In the chosen use case, a UAV equipped with color and thermal cameras was deployed in selected areas, highlighting detected hotspots. This was followed by sending a UGV equipped with a water tank to those hotspots, providing a water supply to firefighters in the field.

This field experiment was successful in tackling the use-case and rated positively by the firefighters, but also showed some limits in its autonomy and reliability. Those encouraging results lead us to continue upgrading the system and deploy it in two additional field tests, refining the software

architecture for easier use and adaptability to new use-cases. The system matured into a software stack displaying different functionalities stemming from the requirements elaborated with end-users. Mainly, it integrates robust and proven technologies (such as MQTT for inter-process communication), provides a streamlined process to be adapted to new projects with different robots and communication protocols, and makes use of a Belief-Desire-Intention (BDI)-based reasoning scheme for goal-driven reasoning [3].

Building on this process, we propose here an updated version of this system architecture named MRSS (Multi-Robot Support System). The focus is on the software engineering, detailing the different modules and communication technologies. We show that MRSS is modular, and can be integrated with arbitrary robots and communication protocols. We also highlight its ability to integrate high-level goal-driven reasoning with actionable robot skills. We go over each component, detailing the design choices and implementation details. We also provide an open-source implementation of this system to help with future field robotics deployments.

Our goal is to propose a flexible architecture leveraging agent-based reasoning and multi-agent monitoring, able to be easily adapted to varied projects. This leads us to leave some implementation to the project developer, in particular regarding the communication with external components. To facilitate such adaptations, a streamlined process for extending the system to new projects is presented, focusing on isolating the necessary changes to specific parts and outlining the required tasks and their rationale. We thus aim at striking a balance between robustness and flexibility.

Finally, we detail different deployments of the proposed architecture and how it allowed end-users to manage complex robot systems in the field.

To summarize the contributions, we propose a software architecture for a multi-robot support system designed in the context of disaster response, facilitating the integration of autonomous robots with external components. We detail the different modules and provide an open-source implementation, and detail how they can be adapted to different use-cases by showcasing past deployments. As an additional byproduct, the reasoner component showcases how to integrate the Jason BDI platform [3] with MQTT to easily integrate with external components to generate percepts and realize blocking actions.

II. RELATED WORK

Over the last few years, many efforts have been made to deploy robot teams in disaster response scenarios. In addition

*This work was partially supported by the Austrian Research Promotion Agency (FFG) with the project KI-SecAssist.

¹Laurent Frering and Gerald Steinbauer-Wagner are with the Institute of Software Engineering and Artificial Intelligence, Graz University of Technology, Graz, Austria. laurent.frering@tugraz.at, gerald.steinbauer-wagner@tugraz.at

to the ones mentioned earlier, we refer the reader to our previous paper for an overview of those [2].

We focus here on projects and related work developing multi-robot system applicable to disaster response scenarios and providing different levels of reasoning.

The SHERPA project [4] had very similar interests, exploring different interaction modalities and control levels with heterogeneous robot teams. The main difference is their focus on teams composed of one human and multiple robots, with direct physical interaction and co-presence. While here the humans and robots may act close to each other, the focus is on the mission level, with centralized decision-making and the ability to directly interact with the robots if necessary.

The NIFTi project [5] focused on designing a user-centric system for multi-human multi-robot cooperation, with realistic field deployments. They iterated over the design over the course of the project, converging towards a robust architecture that proved successful in deployments. However, their system differs to ours by focusing on small robot teams and user-centric semi-autonomous robot skills, whereas we propose a more scalable system with less interaction. In addition to this difference in scope, there is a difference in specificity, as their system is a fully mature solution with tightly integrated components. Our system is less rigid (though less robustly designed for a given task), providing a platform for future developments and focusing on adaptability to different projects.

More recently, Copilot MIKE [6] is an assistant system for multi-robot operations, deployed in the DARPA Subterranean Challenge. It provides well-defined levels of autonomy for task automation, and makes use of a modular scheduler component. While similar to us, we differentiate between higher and lower level goal management, and provide modular interfaces to facilitate reusability.

More generally, the authors of [7] realize a survey of recent multi-agent human-robot interaction systems. They classify those systems in terms of team size, team composition, interaction model, communication modalities, and robot control. They highlight current challenges that are in line with our objectives, such as understanding better the factors influencing workload and situation awareness in multi-human multi-robot teams, the impact of having heterogeneous robots with varying levels of autonomy on human factors, and the importance of scalability and transparency.

III. PROPOSED ARCHITECTURE

MRSS contains four main components interacting with each other, including the robots or agents. The next paragraphs will describe the components' functions and inter-communication, with the full diagram available in Figure 1.

The World Model centralizes and processes the data from different sources, namely the User Interface for newly created objects, the Task Management System for robot status data, and the robots themselves for high-bandwidth data such as image streams and direct control commands. In general, the data is processed and stored for future use by the different components, with the World Model acting as a single source

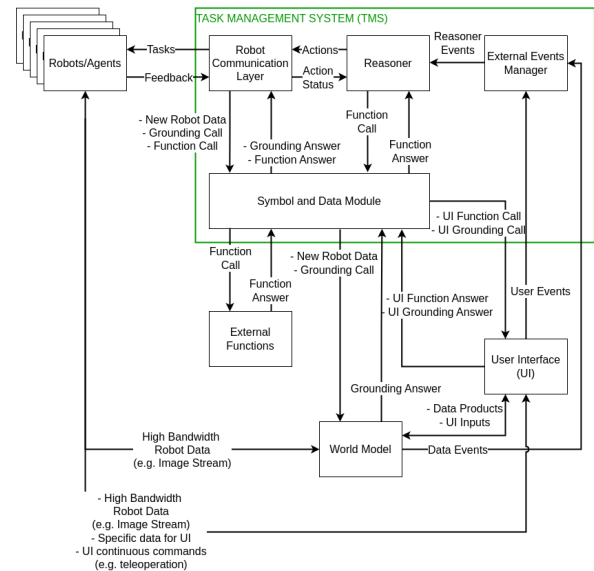


Fig. 1. Overview of the full system architecture. We focus on the Task Management System highlighted in green, and on how those components can be adapted to interact with different external modules.

of truth for the system. It is also equipped with rules dealing with current and new data, used to generate events for the Task Management System.

Those events are converted by the Reasoner submodule into actions for the robots to perform, with possible function calls to external modules during the processing. The actions refer to low-level goals, which are then assigned to one or more robots, and refined into actual robot tasks that the individual robots are equipped to perform. For example, the initial event could be triggered by a UAV detecting a hotspot; the Reasoner would refine this into an action for providing water close to the area, after checking that there is currently none in the vicinity. This action is then assigned to an available UGV, which receives the task of going to the corresponding area. In this way, the initial event is refined into more concrete operations at every level. In order to ground symbols into actual data, the World Model can be queried at any point in time. The Task Management System also monitors task execution and robot state, providing this information to the reasoner and World Model.

As shown in the figure, the User Interface is used to display data from the World Model and generate commands. In general, this information would go through the World Model before being forwarded to the Task Management System. This can also be bypassed in applications where the User Interface may be used to query non-critical information at runtime that is not planned to be used otherwise, for example to snap a picture of the current robot camera view. The same is true for the robot data. Information that is only to be used by the User Interface can be forwarded directly to it. In particular, those mechanisms are important for direct teleoperation, as it would often be the case that introducing

the World Model in the middle of each operation would introduce undesirable latency. In order to safely switch to teleoperation, an event is initially generated so that the Task Management System can suspend all the actions currently assigned to the given robot. When it is notified about the end of the teleoperation, the robot is again considered available and the relevant actions are resumed.

Finally, the robots (or agents, as this architecture could easily accommodate non-embodied agents) are equipped with the ability to autonomously realize the tasks defined for this application, and provide feedback on their progress. They should also be able to update on their current state and communicate with the World Model or the User Interface directly for high bandwidth or low latency data.

In the proposed version, MRSS is able to consistently monitor a fleet of robots and react to changes in state or external events. The communication between the modules is thought to avoid conflictual commands, by centralizing the data in the World Model and minimizing the communication points between the different modules. It also allows for direct control of arbitrary robots, in an integrated way with autonomous decision-making.

IV. IMPLEMENTATION DETAILS

We focus for the rest of the paper on detailing the Task Management System, responsible for the processing of external events and the allocation and monitoring of robot tasks. We will detail how it is implemented, and how it can easily be adapted to different robots, user interfaces, and world models by using modular adapters.

A. Architecture Setup and Communication

Every component of the Task Management System is running in a docker container, and they communicate with each other using MQTT (via an MQTT broker also running in a docker container). This allows for portability and isolation, and MQTT provides a proven and configurable framework that is reliable and easy to inspect and log. Apart from the Reasoner, the components are implemented in Python (version 3.1+). This choice was made to facilitate implementing project-specific modules, as Python3 provides many libraries implementing commonly-used communication protocols.

We will now detail the format of the messages exchanged internally between components. The External Events Manager receives the external events, which are defined as arbitrary messages with a mapping to first-order predicates using a project-specific adapter. Those first-order predicates represent the Reasoner Events, and are encoded as strings and sent over MQTT to the Reasoner. The Reasoner interprets those predicates as BDI percepts, which are refined into goals fed into the agent program. The outputs of the agent program are actions, which represent lower-level goals (either achievement or performative [8]) that the robot team can realize in the environment. They are also defined as first-order predicates and augmented with a Universally Unique Identifier (UUID) and a state variable representing the completion status. The actions are then sent

to the Robot Communication Layer, which refines them into tasks, monitors their execution, and updates their status. The tasks are project-specific parameterized robot-specific skills to be executed, represented by the state machine defined in section IV-E. Finally, the Robot Communication Layer communicates with the Symbol and Data Module to forward it newly received robot data and send it requests to ground abstract objects IDs into python objects. Similarly, the Reasoner is also able to communicate with the Symbol and Data Module as an interface to call external functions. The Symbol and Data Module communicates with external functions and components using project-specific adapters.

The Docker Compose utility is used to easily start the whole stack. This way, every component is started at once and automatically configured for a given project.

B. External Events Manager

As discussed above, the External Events Manager is a Python-based component providing project-specific adapters for converting events with arbitrary representations and communication protocols to first-order predicates encoded as strings and sent over MQTT. It takes the shape of a python package, whose main functions are first to load an MQTT publisher to communicate with the other components, and second to load the project-specific adapter responsible for forwarding external events by using the MQTT publisher. This is done by making use of Python's dynamic import capabilities, to import the right module at runtime given the project name received from the Docker Compose configuration. The project-specific module implements an *External-Adapter* class that has access to the MQTT publisher, but otherwise has the responsibility to implement the custom logic to interpret the project events.

C. Symbol and Data Module

The Symbol and Data Module is similar to the External Events Manager, as it is mostly responsible with providing an adapter from external components to the Task Management System. It is also implemented as a Python package dynamically loading a project-specific module enabling bilateral communication to the internal (via MQTT) and external (via arbitrary communication protocols) components.

It manages requests over MQTT from the Reasoner and Robot Communication Layer, and also processes robot data received from the latter. On the other hand, it has to manage answers from the World Model and external functions over project-specific protocols.

The *DataLayer* class it implements thus provides MQTT callbacks triggered when receiving requests for external function calls, for grounding data via the World Model, and when receiving robot telemetry to be forwarded to the World Model. The exact list of callback functions and internal logic is flexible, as long as the module keeps track of the requests identifiers to send back the correct answer.

To enable effective symbol grounding, it is assumed that objects in the World Model expose relevant identifiers or key attributes to be used by external queries. Those can then be used in reasoning and for retrieving the linked data.

D. Reasoner

In order to provide high-level goal-driven reasoning, the Reasoner makes use of Agent-oriented programming. More specifically, we make use of the Belief-Desire-Intention (BDI) architecture that has a proven track record in providing complex goal-driven multi-agent systems. Fully detailing those concepts and their history is out of scope for the current paper, so we redirect the reader to the following review on those topics [9].

Following our previous work, the Jason platform BDI implementation [3] is chosen for its extensibility. Jason is implemented in Java, and the agents are programmed using the AgentSpeak(L) [10] agent programming language. An example AgentSpeak(L) program is shown on Listing 1, covering a simplified version of the program used in one of the use-cases that makes use of simultaneous prioritized goals.

In order to integrate Jason into the architecture, we had to implement a few adaptations. First, we run it on a docker container as the other components. Second, we wrote a custom Jason environment that instantiates an MQTT client. This client is used to convert the events from the External Event Manager into BDI percepts (i.e. the "inputs" of the agent program). Finally, we implemented a custom logic for the BDI actions: once selected by the agent program, an action is equipped with a UUID and forwarded over to the Robot Communication Layer using MQTT. An asynchronous latching mechanism is then used, so that the action is blocking until the latch is released. Another MQTT subscriber listens to feedback from the Robot Communication Layer, and releases the latch so that the action succeeds or fails according to the received feedback. Finally, a specific action named *rcl_goal_management* bypasses this mechanism, and is used to directly inform the Robot Communication Layer of meta action commands such as cancelling or suspending.

This way, we have a straightforward integration of the BDI agent program into the overall architecture, with actions naturally blocking until the underlying tasks either succeed or fail.

E. Robot Communication Layer

The Robot Communication Layer is maybe the most complex component of the architecture. It receives new action commands from the Reasoner via MQTT, refines them into skills, and monitors their execution on the robots. It also communicates with the Symbol and Data Layer to forward robot data to the World Model, and to request the grounding of data or external function calls.

Once again, this component is implemented as a Python package, dynamically loading a project-specific module.

At the basic level, the Robot Communication Layer provides an abstract class to implement skills. Taking inspiration for existing skill models [11], skills are represented as a simple state machine progressing between the *Start*, *Run*, *Interrupt*, and *Finish* states. They can also acquire and release resources, though this mechanism was not yet tested in the deployments. A *SkillManager* class is available to interface

```

/* Initial beliefs */
isuav("r1").
available("r1").
/* Percepts */
+area_goal_received(G,P)[source(percept)] : true <- !
    coverarea("d1", G, P).
+goal_cancelled(G)[source(percept)] : currenttask(R,G,_) &
    isuav(R) <- -currenttask(R,G,_) ; .drop_intention(
    coverarea(R,G,_)) ; +available(R).
+prioritychange(G,P)[source(percept)] : .intend(coverarea(
    R,G,P)) <- .drop_intention(coverarea(R,G,P)) ; !!
    coverarea(R,G,P).
/* Plans */
+!coverarea(R, G, P) : isuav(R) & available(R) <- -
    available(R) ; +currenttask(R, G, P) ; cover(R, G) ; -
    currenttask(R,G,P) ; +available(R).
+!coverarea(R, G, P) : isuav(R) & not available(R) &
    currenttask(R, H, Q) & P>Q <- -currenttask(R,H,Q) ; +
    currenttask(R,G,P) ; .drop_intention(coverarea(R,H,Q))
    ; !!coverarea(R,H,Q) ; cover(R,G) ; -currenttask(R,G,P)
    ; +available(R).
+!coverarea(R, G, P) : isuav(R) & not available(R) <- .
    wait(2000) ; !!coverarea(R, G, P).

```

Listing 1. An example BDI Agent code for managing a single UAV with goal priority and cancellation. The Percepts are obtained via the External Event Manager, and respectively generate a *coverarea* goal, cancel an existing goal, or change the priority of an existing goal. The plans implement the behaviours for a *coverarea* goal: either sending the *cover* action to the Robot Communication Layer if the UAV is available, cancelling a previous lower-priority goal if necessary, or waiting and retrying if there is a current higher-priority goal.

with existing skills, running them in separate threads and managing their transitions. The individual skills and their behavior are left to the project programmer.

When an action is received, it is refined according to project-specific "recipes": a specific list of skill is instantiated, and data may be retrieved either from a previous execution (e.g. last waypoint reached for an area coverage skill), or by making a request to the Symbol and Data Layer. Each skill of the action is then started, and continuously monitored as part of the main loop of the module. The default behavior is akin to a logical AND: the action succeeds if all of its skills succeed, and fails if any of its skills fails. This can however be customized for each action by the developer. Once an action fails or succeeds, arbitrary data on its execution may be stored for future reference and the final action status is forwarded to the Reasoner over MQTT, so that the corresponding BDI agent plan can progress (or fail). Lastly, when the special *rcl_goal_management* action is received from the Reasoner, the corresponding action is directly transitioned to the corresponding state. This allows the Reasoner to bypass the normal behavior of the Robot Communication Layer if necessary in the high-level reasoning (for example, to directly interrupt an action or make it succeed).

V. ADAPTATION PROCESS

We now detail and summarize the process for adapting MRSS to a new project or use-case. We go over all the parts that may be changed, highlighting the reasoning and specifications. A summary of this process can be seen in Table I.

To facilitate this process, a default project is implemented, providing a template for every part to be changed.

The first step is to create a new Docker Compose file for the project. The file should be named *docker-compose-{project_name}.yaml* (with *{project_name}* to be replaced by the project name) in order to be conveniently started by the companion script which manages the clean starting, stopping, and rebuilding of containers. Using a separate Docker Compose file lets the components use different containers and images and allow for additional customization. The Docker Compose file can fully reuse the default one, but the user may modify it they need additional components to be started, or if they wish to use project-specific Dockerfiles instead of the default ones.

For the Event External Adapter, the user has to create a new module in the *external_adapters* folder named *{project_name}.py*. Similar to the default template, this module should implement an *ExternalAdapter* class, making use of the provided MQTT client to publish the Reasoner events. This class act as an adapter with the project-specific event representation and communication protocol.

Adapting the Reasoner consists only of adapting the Jason-related files. This includes the *{project_name}.mas2j* file, which should simply point to the agent file (and any addition that a knowledgeable Jason developer may use). The agent file, *{project_name}.asl*, is a standard Jason agent program. The only specific requirement is that, by default, it is assumed that the Reasoner performs action assignment to a specific robot. This means that actions are represented as first-order logic predicates in the shape *action(robot_id, goal_id)*. If the project requirements are different, this can be changed; however the Robot Communication Layer's action callback will need to be changed accordingly (see below).

For the Robot Communication Layer, the user has to create the project module *rcl-{project_name}.py* implementing the *RobotCommLayer* class in the *projects* folder. We suggest also creating a *rcl {project_name} skills.py* module in the same folder to separate the main code from the skills definitions. To implement the skills, it is necessary to import and inherit from the *Skill* class from the *rcl_skill_model.py*. Each skill should have a custom implementation of the *start*, *run*, *finish*, and *terminate* functions. The *RobotCommLayer* class can be templated from the default one, but the user has to adapt the *refine_action* function with the project-specific action refinement recipes (i.e. which skills are started with a given action). Optionally, the user can adapt the actions' termination conditions in the *check_action_status* function, and has to adapt the *actionCB* callback function if the action representation in the Reasoner was changed.

Finally, the Data Layer simply requires a module in the *projects* folder implementing the *DataLayer* class. Similar to the default template, this class should implement MQTT callbacks for receiving requests and robot data from the Robot Communication Layer. Those callbacks should trigger the necessary communication to forward the requests to external components and populate the world model.

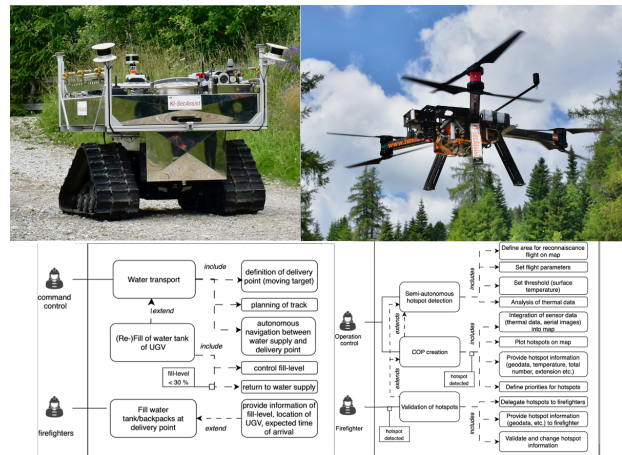


Fig. 2. Water transport UGV (top-left), hotspot recognition UAV (top-right), and use-case diagram for the first experiment, taken from [2].

VI. DEPLOYMENTS

Multiple iterations of this architecture were deployed over the last couple years in different projects. An initial prototype was used the first field tests of the KISecAssist project [2]. There, Austrian firefighters had access to a User Interface for managing one UAV and one UGV in a mountain wildfire scenario. They could define areas for the UAV to cover and map, where it may also detect hotspots. Those hotspots could then be used as targets for the UGV to navigate to. The UGV was equipped with a water tank, and firefighters could pump water out of it. Additionally, the UAV was able to interrupt its tasks to go back to the home base when its battery was low, and continue where it left off afterward. The UGV also automatically came back to the home base when the water level was low. All the goals could be sent with priorities, and the robots were able to suspend and resume goals accordingly. Pictures of the two robots and the use-case diagram for the experiment are shown on Figure 2.

This experiment used the technologies highlighted above, but this first version was not designed for customization and was therefore not containerized, and was implemented only for this use-case. The communication with the User Interface and the World Model was done via Rest API calls. The UGV was controlled via MQTT, and the UAV via MAVlink. Even though this first experiment highlighted different technical problems, it was still relatively successful and received an encouraging evaluation from the firefighters. It managed to showcase how it was possible to integrate all the components together with different communication technologies.

As a second step, MRSS was used as part of the AMADEE-24 Mars analog mission taking place in Armenia [12]. This experiment used a single mobile manipulator equipped with different sensors. For this application, the architecture was refined, making use of docker and providing insights on its adaptability by applying it to another use-case. There, the system communicated to a PostGIS database and to the robot via MQTT. The reasoner was responsible

TABLE I

SUMMARY OF THE ADAPTATION PROCESS TO NEW PROJECTS. PARTS IN ITALIC ARE OPTIONAL AND ARE RELATED TO DEEPER CHANGES TO THE SYSTEM. PROJECT_NAME IS TO BE REPLACED WITH THE PROJECT NAME.

Component	Files to create	Specific parts to adapt
Docker Compose	docker-compose-{project_name}.yaml	<i>additional components</i> <i>custom Dockerfiles</i>
Event External Adapter	{project_name}.py	ExternalAdapter class
Reasoner	{project_name}.mas2j {project_name}.asl	<i>custom action definition</i>
Robot Communication Layer	rcl_{project_name}.py rcl_{project_name}_skills.py	skills inheriting from Skill class actions refinement process in refine.action function <i>actions termination conditions in check.action_status function</i> <i>action callback in actionCB function</i>
Data Layer	dl_{project_name}.py	MQTT request and data callbacks

for checking the legality of a given action according to the operational requirements. This second deployment of the proposed system was overall simpler in the number of technologies to integrate and in the high-level reasoning, but it highlighted how the system could be adapted in another use-case, with different requirements.

Finally, the last deployment was made for the final tests of the KISecAssist project. There, similar technologies were used as in the first deployment, but the system was fully updated according to the definitions above. The use-cases were also updated following firefighters' requests, notably including the requirement to directly teleoperate the UGV. This was straightforward to implement by opening a direct HTTP connection from the User Interface to the UGV. To maintain safety, the User Interface first notifies the Task Management System of the switch to direct teleoperation. This leads the Reasoner (and then the Robot Communication Layer via the *rcl_goal_management* action) to suspend all existing goals relating to the UGV. Then, a specific skill is used to switch the UGV to direct control mode. Once the teleoperation is done, the user would move to a safe spot and manually trigger the switch back to autonomous control on the User Interface. This lead the reasoner to resume all suspended goals so that the robot could proceed naturally.

Adding such a function highlighted the flexibility of MRSS. Indeed, by relying on the goal and concurrency management of the system, this behavior could be added with only a few lines of code in the expected modules and with very minimal debugging.

VII. CONCLUSION

We showcased and detailed a system for multi-robot control in practical deployments. The proposed architecture and implementation result from an iterative design, based on requirements by end users and considerations of software engineering principles. The main benefit of the system is its capacity to blend agent-oriented programming for high-level control with task-level robot control, in a flexible way that provides clear guidelines to adapt it to different projects. The proposed system was tested during field experiments in three different occasions, allowing for iterating over the design and adapt it to different use-cases. We plan to continue using and updating the system for new projects and to accommodate

new capabilities such as resource management at the skill level. Moreover, the system would benefit for an integrated monitoring and debug tool, making use of the internal MQTT messages and providing test procedures.

REFERENCES

- [1] J. Delmerico, S. Mintchev, A. Giusti, B. Gromov, K. Melo, T. Horvat, C. Cadena, M. Hutter, A. Ijspeert, D. Floreano, *et al.*, "The current state and future outlook of rescue robotics," *Journal of Field Robotics*, vol. 36, no. 7, pp. 1171–1191, 2019.
- [2] L. Frering, A. Koefer, M. Huber, S. Pfister, R. Feischl, A. Almer, and G. Steinbauer-Wagner, "Multi-robot support system for fighting wildfires in challenging environments: System design and field test report," in *2023 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2023, pp. 32–38.
- [3] R. H. Bordini and J. F. Hübner, "Bdi agent programming in agentspeak using jason," in *International workshop on computational logic in multi-agent systems*. Springer, 2005, pp. 143–164.
- [4] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, *et al.*, "The sherpa project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments," in *2012 IEEE international symposium on safety, security, and rescue robotics (SSRR)*. IEEE, 2012, pp. 1–4.
- [5] G.-J. M. Kruijff, I. Kruijff-Korbayová, S. Keshavdas, B. Larochelle, M. Janiček, F. Colas, M. Liu, F. Pomerleau, R. Siegwart, M. A. Neerinx, *et al.*, "Designing, developing, and deploying systems to support human-robot teams in disaster response," *Advanced Robotics*, vol. 28, no. 23, pp. 1547–1570, 2014.
- [6] M. Kaufmann, T. S. Vaquero, G. J. Correa, K. Otstr, M. F. Ginting, G. Beltrame, and A.-A. Agha-Mohammadi, "Copilot mike: An autonomous assistant for multi-robot operations in cave exploration," in *2021 IEEE Aerospace Conference (50100)*. IEEE, 2021, pp. 1–9.
- [7] A. Dahiya, A. M. Aroyo, K. Dautenhahn, and S. L. Smith, "A survey of multi-agent human-robot interaction systems," *Robotics and Autonomous Systems*, vol. 161, p. 104335, 2023.
- [8] L. Braubach, A. Pokahr, D. Moldt, and W. Lamersdorf, "Goal representation for bdi agent systems," in *International workshop on programming multi-agent systems*. Springer, 2004, pp. 44–65.
- [9] R. Calegari, G. Ciatto, V. Mascardi, and A. Omicini, "Logic-based technologies for multi-agent systems: a systematic literature review," *Autonomous Agents and Multi-Agent Systems*, vol. 35, no. 1, p. 1, 2021.
- [10] A. S. Rao, "Agentspeak (I): Bdi agents speak out in a logical computable language," in *European workshop on modelling autonomous agents in a multi-agent world*. Springer, 1996, pp. 42–55.
- [11] C. Lesire, D. Doose, and C. Grand, "Formalization of robot skills with descriptive and operational models," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7227–7232.
- [12] S. Jusner, S. Moser, S. Schaffler-Glöbl, R. Halatschek, M. Eder, and G. Steinbauer-Wagner, "A mission architecture for a human-robot collaborative planetary exploration cascade," in *2024 International Conference on Space Robotics (iSpaRo)*. IEEE, 2024, pp. 321–327.

DOPS: Drone Optimized Performance Score for Evaluating Real-Time Tomato Ripeness Detection

Ylli Rexhaj¹, Roni Kasemi², Lucas Lammer³

Abstract—In recent years, deep learning (DL) has emerged as a promising tool to detect ripeness or diseases in different types of plants, which helps farmers monitor crop health and determine the optimal harvest times. However, a significant challenge is the integration of these DL models into drones (UAVs) due to low onboard computing capacity, forcing the images captured by UAV cameras to be transmitted to ground-based processors, introducing delays relying on wireless data transmission that compromise real-time identification and affect the accuracy and efficiency of real-life classification. In this study, we present a new metric called Drone Optimized Performance Score (DOPS) to optimize the performance of real-time Tomato Ripeness Detection, taking into consideration accuracy, frames per second (FPS), and latency. We use a systematic methodology where our research includes an approach in the model training phases and also in the deployment phase of two CNN models, MobileNetV2 and ResNet50, with a main focus on evaluating key performance metrics for classification from drones and integrated cameras. Initially, the lighter model MobileNetV2 proves to be more effective for real-time applications based on DOPS evaluation, but after applying a series of optimizations to ResNet50, which is a resource-intensive model, we can maintain its superior accuracy of 98%, but also outperform MobileNetV2 in DOPS evaluation with higher FPS and lower latency, proving that resource-intensive models can also be optimized for real-world deployment.

I. INTRODUCTION

Agriculture has long been a vital pillar of society, ensuring both economic sustainability and food security since the beginning of humanity. The Food and Agriculture Organization (FAO) predicts that by 2050, there will be more than 9.73 billion people on the planet, and by 2100, there may be 11.2 billion. As a result, the food sector is under pressure to provide the rising demand for food [1]. To solve these problems and boost production and efficiency, the advancement of Artificial Intelligence (AI) and Machine Vision (MV) are playing a crucial role [2].

Many agricultural practices have been significantly advanced through the integration of AI and Machine Vision in precision agriculture, and according to S. Jinya *et al* [3] one of the new developments is the integration of AI with Unmanned Aerial Vehicles (UAVs) to achieve higher productivity in special, large or untargeted spaces, minimize the cost, and automate the process. Drones or UAVs equipped

with high-resolution cameras and sensors have emerged as very valuable tools to capture detailed images and gather important crop data, which, when combined with AI, help farmers monitor crop health and determine the optimal harvest times [4],[5],[6].

Although numerous research studies have been done on the detection of ripeness or diseases of vegetables or fruits using machine vision and deep learning (DL) [7],[8], [9], the key challenge is deploying these DL models on UAVs due to limited onboard computational capacity, requiring the images captured by UAV cameras to be transmitted to ground-based processors for analysis, introducing delays relying on wireless data transmission that can compromise real-time identification and affect the accuracy and efficiency of real-time classification [10], [11], [12]. To tackle the existing problems we present a new metric called DOPS - Drone Optimized Performance Score to optimize the performance of real-time classification of tomato ripeness taking into account accuracy, frames per second (FPS), and latency. Real-time and low-latency classification are crucial in precision agriculture for timely decisions affecting crop health, yield, and resource optimization, enabling targeted interventions and minimizing damage [13]. The research conducts an analysis to compare how a lightweight model MobileNetV2 [14][15] and the resource-intensive Convolutional Neural Network (CNN) model ResNet50 [16][17] perform on two setups: a drone equipped with an onboard camera that captures aerial video and streams it to a ground-based processing unit, and a laptop using its built-in webcam in a controlled indoor environment. The study also investigates how these models should be optimized to achieve higher DOPS. The key contributions of this paper are:

- Introduction of DOPS as a Novel Evaluation Metric: The Drone-Optimized Performance Score is introduced to evaluate model performance on edge-device drones, integrating accuracy, FPS, and latency for real-time applications.
- Benchmarking CNN Architectures for UAV-Based AI: We systematically compare MobileNetV2 and ResNet50, identifying the best-performing architecture for drone-based AI applications.
- Optimization of ResNet50 for Real-Time Performance: ResNet50 undergoes targeted optimizations, including input size reduction, layer freezing, and mixed precision training, improving efficiency for real-time UAV applications without sacrificing accuracy.

The paper proceeds as follows. The related work can be

*This work was not supported by any organization

¹Ylli Rexhaj is with Faculty of Mechatronics Engineering, University for Business and Technology, 10000 Prishtina, Kosovo ylli.rexhaj@ubt-uni.net

²Roni Kasemi is with Faculty of Mechatronics Engineering, University for Business and Technology, 10000 Prishtina, Kosovo roni.kasemi@ubt-uni.net

³Lucas Lammer lucas.lammer@gmail.com

found in Section 2. Our method for the experimental setup, training phase, and DOPS evaluation measure is described in Section 3. The results of the evaluation phase, the deployment phase in a real-time application, and the model tuning for improved performance are shown in Section 5. A summary of the results and a proposal for further research are presented in Section 6.

II. RELATED WORK

Rejeb *et al.* [18] states that drones are changing the agricultural industry by improving efficiency and operational costs. Drones are used to monitor diseases, reducing pesticide usage and the need for human inspection of the crops. Image and sensor technologies in UAVs (Unmanned Aerial Vehicles) allow farmers to precisely monitor crops and detect diseases early, reducing the need for human labor. However, their study primarily offers a bibliometric overview and does not address practical aspects of deploying affordable, low-cost drones or the feasibility of running algorithms on external computing devices rather than onboard hardware. Rajagopal and Murugan [19] use AI-powered drones to detect diseases in cashew trees. MobileNetV2, a deep learning model, is used to scan photos and pinpoint diseases in their early stages to minimize damage to the trees. Egi *et al.* [20] designed a system that processes drone footage to identify and count tomato flowers and fruits. Their method uses YOLOv5 [21] for object detection and Deep-Sort for tracking. While this system works well for estimating how many fruits and flowers are present, it is focused on counting rather than analyzing ripeness. Hobart *et al.* [22] shows an example of a low-cost drone paired with a consumer-grade RGB camera to detect ripe fruits, demonstrating the potential for affordable solutions in agriculture monitoring with UAVs. While their work focuses on apples, similar approaches can be adapted for other crops, including tomatoes. For tomato ripeness classification specifically, Wang *et al.* [23] introduces a tomato ripeness detection system based on an existing detection framework (RT-DETR) [24], which they adapt to be more efficient. Khan *et al.* [25] introduce a technique that combines CNNs with transformer-based models for tomato ripeness classification. Zhang *et al.* [26] alters YOLOv8 [27] in a different investigation to manage intricate ripeness detecting settings. Although all three models are effective in classifying the ripeness of the crops, they are not developed to run on the lower-end hardware of edge devices. Hernández *et al.* [28] investigates a less compute-intensive method to deal with this by classifying tomato ripening stages using YOLOv3tiny [29]. Their approach tries to find a compromise between accuracy and computational requirements. Therefore, they only use data from a very controlled environment, which makes the model less suited outside of controlled environments.

III. OUR APPROACH: DOPS

Although ripeness detection is very critical, the main objective of this research is to tackle the existing problem of

CNN models in real-time applications. This research is structured into two significant phases: the model training phase and the real-time deployment phase. In the initial phase, we concentrate on training the models and assessing the performance of two architectures, MobileNetV2 and ResNet50. We conduct a comparative analysis of their accuracy before testing them in real-world scenarios. The subsequent phase involves deployment, during which we introduce a new metric known as DOPS to evaluate the effectiveness of real-time applications. In this phase, we compare the models using two distinct camera setups: a drone-mounted camera and an integrated laptop camera. This methodology addresses existing challenges related to the classification of wirelessly transmitted frames.

A. Experimental Setup

DJI, in collaboration with Intel, created the compact, reasonably priced DJI Tello, a fully programmable drone that records 720p HD video with its 5-megapixel camera [30]. Due to the restricted processing capability of the drone's onboard processor, direct integration of DL models is not feasible. In this instance, the Tello functions as an aerial imaging tool, gathering visual information and transmitting it in real time to a system on the ground (a laptop in this case). This enables the drone to concentrate on gathering data while the laptop's computing capacity is used to run complex AI models for tasks like classifying the ripeness of tomatoes.

The drone-captured frames are sent in real time to a ground-based laptop with powerful processing capabilities. To better compare the two setups, the laptop also has a 720p HD resolution camera, which allows for a better comparison of how well the two models work with various image sources using the same computational framework.

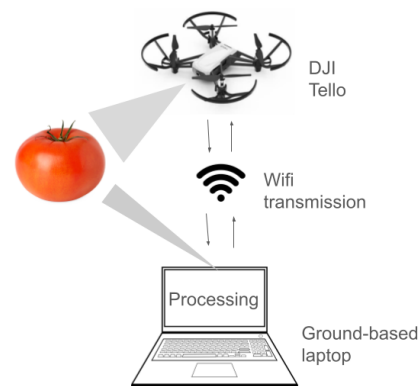


Fig. 1. Experimental Setup

B. Model Training Phase

The models MobileNetV2 and ResNet50 are trained using a small dataset of 711 images, of which 294 are of ripe tomatoes, 302 are of unripe tomatoes, and 115 are for the background to minimize false detection in the background.

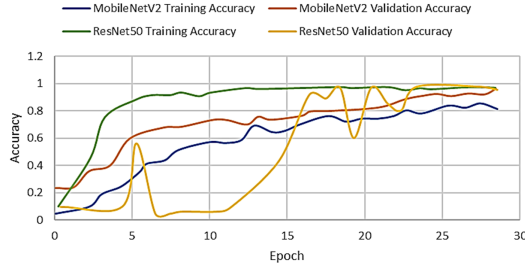


Fig. 2. Training and Validation Accuracy

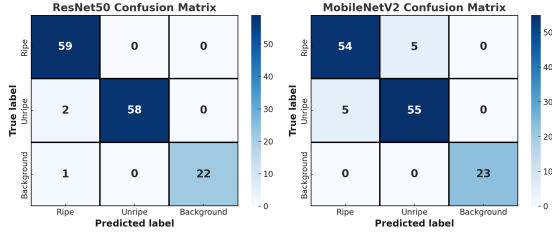


Fig. 3. Comparing Confusion Matrix

Both models are trained for 28 epochs using 569 of the dataset's images for training and 142 for validation. Images of 512x512 pixels are used to train the ResNet50 model, while 224x224 pixel images are used to train the MobileNetV2, which is the primary difference between the models. Validation accuracy and loss metrics are used to moderate the two models' training. As we will discuss later, the model's accuracy and inference latency are impacted by the disparity in using different image size inputs. The training and accuracy curves are illustrated in Figure 2. Starting with a lower initial accuracy, the MobileNetV2 gradually improves, reaching 93% validation accuracy, whereas ResNet50 demonstrates a faster rate of convergence and reaches a higher validation accuracy of 99%. The key difference is that the ResNet50 works better on larger input sizes and contributes to better feature extraction and discrimination between classes.

During training, both models showed a steady decline in training and validation loss. ResNet50 had a lower and more stable validation loss, indicating a strong fit. MobileNetV2 showed more fluctuations, which, while less stable, can help the model avoid sharp, narrow solutions in the loss landscape. Prior work [31] suggests that flatter solutions tend to generalize better to unseen data.

We look at the confusion matrix for both models in Figure

Metric	MobileNetV2	ResNet50
Accuracy	0.93	0.99
Macro Avg F1	0.94	0.99
Weighted Avg F1	0.93	0.99

TABLE I
PERFORMANCE COMPARISON

4 and a summary of important performance metrics in Table 1 to further assess how well the two models perform. Because of its deeper architecture and larger input size, ResNet50 performs better than MobileNetV2 across all evaluation metrics, according to the results of training both models. This advantage, however, comes at the expense of higher latency and computational demand, as MobileNetV2, a lightweight model, shows a competitive performance.

C. DOPS

DOPS is a metric that balances accuracy, FPS (Frames per Second), and latency to evaluate a model's effectiveness in real-time applications. In this case, we use the DOPS metric to compare models in different environments such as laptop vs drone, and also optimize the models for better deployment on real-time applications, where accuracy indicates the classification accuracy of the model, latency (ms) is the time taken for a single inference, including preprocessing and post-processing and FPS measure how many images the model processes per second.

$$\text{DOPS} = \frac{\text{Accuracy} \times \text{FPS}}{\text{Latency}} \quad (1)$$

Different deployment environments prioritize different factors for example in cloud-based applications, accuracy, and FPS are more important as computational resources are not a constraint, but on the other hand in low-power edge devices such as drones, latency, and power consumption are critical, making efficiency a key factor, adding a weight assigned to each parameter based on the real-time application. However, detailed power consumption analysis is left for future work and will be integrated in subsequent stages of development.

IV. RESULTS

The evaluation of deep learning models for real-time application requires a comprehensive analysis beyond traditional accuracy-base metrics, as introduced in the training phase section. So, using DOPS, we analyze the real-time performance of MobileNetV2 and ResNet50, ensuring the model's reliability and practical deployment constraints, such as FPS and latency, which play a critical role in real-world applications.

A. Deployment Phase

Using two different camera setups, the drone-mounted camera and an integrated laptop camera, the deployment phase concentrates on testing the two models' real-time tomato ripeness detection performance. The drone itself does not perform any onboard processing, instead, it operates as a mobile image acquisition platform, transmitting frames wirelessly in real-time to the laptop from an aerial perspective. To ensure consistency, all image processing is carried out on a laptop equipped with a dedicated GPU. In contrast, the laptop setup eliminates any wireless transmission delay by using the integrated camera to capture frames. In real-time testing, the main difference between the two configurations was the effect of wireless transmission latency.

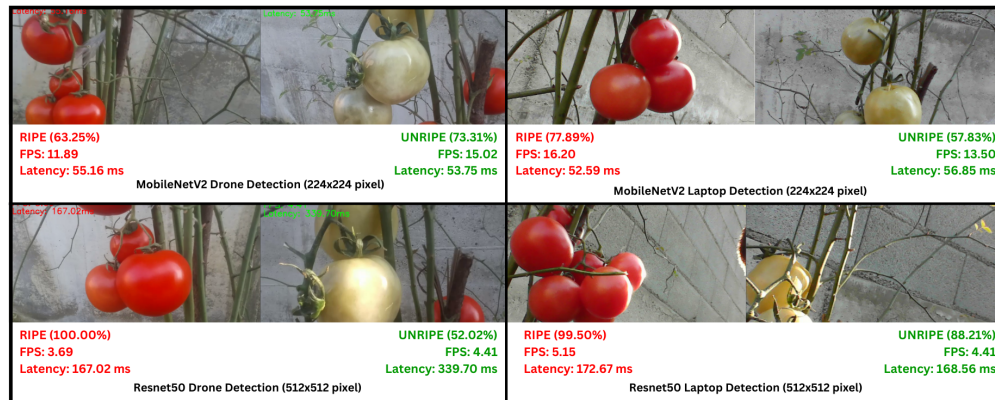


Fig. 4. Real-time FPS and latency acquisition for DOPS evaluation, comparing MobileNetV2 and ResNet50 across drone and laptop setups.

The drone camera wirelessly transmits the frames, as we previously discussed, but this leads to network-induced delays that result in irregular frame drops and slow processing time. Even though the aerial perspective covers a larger range of view, latency has a detrimental impact on real-time classification. In contrast, directly collected frames on the laptop eliminate transmission latency, enabling faster inference and greater FPS. However, because of its relatively limited field of view, the fixed camera proves less effective in monitoring large crop fields.

During the testing phase, both models MobileNetV2 and ResNet50 are tested in real-time in both setups, compared, and evaluated using the DOPS metric, the results of which are shown and compared in section 4 B, from the real-time acquisition. The live performance of both models is shown in Figure 4 with the labels of live measurement results of FPS, latency, and accuracy.

B. DOPS Evaluation

Following the real-time deployment phase, we evaluate MobileNetV2 and ResNet50 in laptop and drone configurations using the DOPS. DOPS provides a balanced assessment that reflects real-time feasibility, which is crucial for agricultural AI applications, in contrast to traditional evaluations that only consider accuracy. The primary objective of this evaluation is to assess each model's performance by combining accuracy, frames per second (FPS), and latency as critical variables. Stated differently, an AI model is considered better when it has a higher DOPS score, meaning it can process frames quickly and with minimal latency while maintaining high classification accuracy, on the other hand, lower DOPS indicates worse real-time performance. A higher DOPS score indicates that the model is well-suited for real-time applications, as speed and precision are crucial in drone-based agricultural monitoring [32].

Despite this, MobileNetV2 is an optimal model for real-time inference, maintaining a strong balance between accuracy and processing speed.

In contrast, the resource-intensive ResNet50 model, known for its superior accuracy, performs well in terms of classi-

Setup	Laptop	Drone
Accuracy	0.93	0.93
Latency	62.26	65.56
FPS	15.44	15.05
DOPS	0.23	0.21

TABLE II

MOBILENETV2 (LAPTOP VS DRONE) DOPS EVALUATION

Setup	Laptop	Drone
Accuracy	0.99	0.99
Latency	173.52	312.19
FPS	5.61	3.30
DOPS	0.032	0.010

TABLE III

RESNET50 (LAPTOP VS DRONE) DOPS EVALUATION

cation accuracy, with 99% accuracy in both sets. However, as seen in Figure 5, its computational intensity significantly impacts its real-time usability. Table 3 shows that compared to MobileNetV2, the FPS in the laptop setup is only 5.61, and in the drone setup, it is much lower at 3.30. Additionally, ResNet50 has a significantly greater latency, reaching 312.19 ms in drone setup and 173.52 ms in laptop setup. After the evaluation, the DOPS results of ResNet50 are far lower than those of MobileNetV2. Scoring 0.0320 on the laptop and only 0.0105 for the drone setup. This demonstrates that, despite its great accuracy, ResNet50 is not appropriate for real-time drone-based agricultural applications due to its slow inference speed and high latency.

The main factor causing ResNet50 to perform worse than MobileNetV2 in all performance metrics is its large input size (512x512), while MobileNetV2 has an input size of 224x224. This resolution has a significant effect on processing time and computational power, leading to higher latency and a lower frame rate since each image requires more memory and computation with each forward pass. As a result, ResNet50 is ineffective for real-time applications and takes longer to analyze each frame. This leads to lower FPS and greater latency, whereas MobileNetV2's smaller input size enables

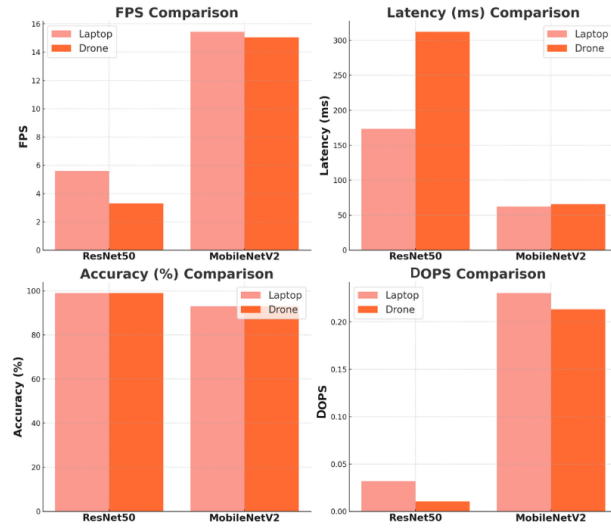


Fig. 5. DOPS Evaluation comparison of MobileNetV2 & ResNet50 (Laptop vs Drone)

faster processing with less memory usage, allowing for smoother and more stable real-time inference. To add to this, ResNet50 is a model with more parameters and convolutional layers than MobileNetV2, which uses fewer computational resources. Wireless transmission affects both models, with ResNet50's larger input size causing heavier data packets and increased network latency. MobileNetV2's smaller input size leads to faster data transfer and less lag, making it more effective in drone-based agricultural monitoring. ResNet50 performs worse in real-time deployment due to its deeper network architecture, larger input shape, and higher computational demands. However, MobileNetV2's ability to balance speed and accuracy makes it more suitable for real-time AI applications, especially in UAV-based agricultural monitoring.

C. Optimizing ResNet50 for Real-Time Use

In this section, we'll maximize ResNet50's effectiveness without sacrificing its high classification accuracy. Our main optimizations include freezing the first 50 layers of ResNet50 and lowering the size of the input image. We also use TensorFlow's automatic mixed precision feature to apply mixed precision training, using FP16 computation whenever feasible. A balance between computational efficiency and retention is achieved by reducing the input size from 512x512 to 256x256, which increases inference speed without significantly affecting classification performance.

Using mixed-precision training optimization not only reduces GPU memory usage but also accelerates training and inference speed, making the model more appropriate for real-time deployment. Freezing the layers allows the model to retain its ability to extract robust features, which results in faster model convergence and decreased processing time per frame. Following training with these adjustments, the model's training performance is displayed in Figure 6. It

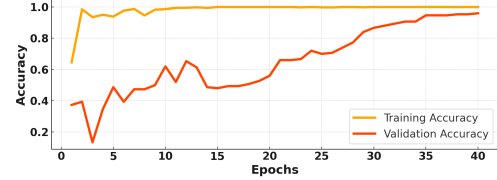


Fig. 6. Optimized ResNet50 Accuracy over Epochs

is evident that the optimized ResNet50 maintains a high training and validation accuracy of 98% throughout the training phase. This time, 30% of the data is split for validation during training, and we observe that only four images are incorrectly classified in Figure 7 in the confusion matrix. But evaluating the model in real-time testing during the deployment phase is the primary objective through the DOPS Evaluation.

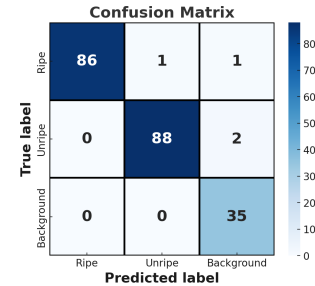


Fig. 7. Confusion Matrix of Optimized ResNet50

Setup	Laptop	Drone
Accuracy	0.98	0.98
FPS	20.93	21.69
Latency (ms)	42.89	46.74
DOPS	0.48	0.45

TABLE IV

DOPS EVALUATION FOR OPTIMIZED RESNET50

The optimized ResNet50 model outperforms MobileNetV2 in real-time inference speed and latency, despite maintaining a high classification accuracy of 98%. The laptop setup achieves an FPS of 20.93, while the drone setup slightly outperforms it at 21.69 FPS. The optimizations increase processing speed without impacting performance. Latency, a key variable of the DOPS evaluation, is substantially reduced compared to the model before the optimization. The laptop setup achieves a latency of 42.89 milliseconds, while the drone setup exhibits a slightly higher latency of 46.74 milliseconds, primarily due to wireless transmission delay. These latency improvements are significant compared to the standard ResNet50 implementation, making the optimized model more viable for real-time classification in drone-based agricultural monitoring. The overall DOPS score confirms the success of these optimizations, with the laptop setup achieving a DOPS score of 0.478, while the drone setup slightly lags at 0.454 due to network-related delays. These

results highlight that the optimized ResNet50 successfully balances accuracy and real-time performance, making it a viable solution for UAV-based agricultural classification tasks.

V. CONCLUSIONS

In this work, we introduce a new metric for evaluation the Drone Optimized Performance Score (DOPS), a benchmark for real-time deep learning inference on UAVs. DOPS is a metric that takes into account accuracy, frame rate, and latency, providing a simple yet powerful evaluation for the real-time application of AI models. As a finding initially, MobileNetV2, being a lightweight model, outperforms the heavy resource model ResNet50 in real-time inference speed and, therefore, overall in DOPS. But, after some optimizations such as input size reduction, mixed-precision training, and layer freezing made on the ResNet50, it was able to surpass MobileNetV2, raising the DOPS score from 0.010 to 0.45 and improving the real-time performance while still maintaining a high classification accuracy. In agricultural monitoring, where timely and precise identification of issues like crop stress or insect outbreaks is crucial to preventing yield loss and guaranteeing resource efficiency, this study shows that DOPS is a useful metric for assessing real-time applications. Also, it provides a framework for improving deep learning models' performance on edge devices such as UAVs while still maintaining high accuracy. Future work includes further optimizing deep learning models for UAV-based inference by integrating advanced model compression techniques such as pruning and quantization. These methods will reduce computational overhead while maintaining high classification accuracy. Additionally, deploying the models directly on the edge devices is work that will be implemented to eliminate transmission delays, improving real-time responsiveness. Also, to extend the applicability of DOPS, power consumption will be incorporated as a metric, enabling more energy-efficient AI deployments on battery-powered drones.

REFERENCES

- [1] Food and Agriculture Organization of the United Nations (FAO). *The Future of Food and Agriculture: Trends and Challenges*. FAO, Rome, Italy, 2017.
- [2] Rosana Cavalcante de Oliveira and Rogério Diagne de Souza e Silva. Artificial intelligence in agriculture: Benefits, challenges, and trends. *Applied Sciences*, 13(13), 2023.
- [3] Jinya Su, Xiaoyong Zhu, Shihua Li, and Wen-Hua Chen. Ai meets uavs: A survey on ai empowered uav perception systems for precision agriculture. *Neurocomputing*, 518:242–270, 2023.
- [4] Zhao Zhang, Hu Liu, Ce Yang, Yiannis Ampatzidis, Jianfeng Zhou, and Yu Jiang. *Unmanned Aerial Systems in Precision Agriculture*. 05 2022.
- [5] Jong-Hwa Park and Dong-Ho Lee. Development of a uav-based multi-sensor deep learning model for predicting napa cabbage fresh weight and determining optimal harvest time, 07 2024.
- [6] Juhi Agrawal and Muhammad Yeasir Arafat. Transforming farming: A review of ai-powered uav technologies in precision agriculture. *Drones*, 8(11), 2024.
- [7] Andreas Kamilaris and Francesc X. Prenafeta-Boldú. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018.
- [8] Md Habib, Md. Ariful Arif, Sumaita Binte Shorif, Mohammad Shorif Uddin, and Farruk Ahmed. *Machine Vision-Based Fruit and Vegetable Disease Recognition: A Review*, pages 143–157. 03 2021.
- [9] Mahmoud Soltani Firouz and Hamed Sardari. Defect detection in fruit and vegetables by using machine vision systems and image processing. *Food Engineering Reviews*, 14, 03 2022.
- [10] Alfonso Torres-Rua. Drones in agriculture: an overview of current capabilities and future directions. In *Utah Water Users Workshop, Saint George, UT, USA*, pages 1–9, 2017.
- [11] Chenglin Zhang, Xiaoyong Qiang, Qingzhe Lv, Hao Guan, and Fang Huang. Comparative study on real time image data transmission methods for unmanned aerial vehicle. In *IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium*, pages 6609–6612, 2024.
- [12] Xu Kang, Bin Song, Jie Guo, Zhijin Qin, and F. Richard Yu. Task-oriented image transmission for scene classification in unmanned aerial systems, 2021.
- [13] Tarek Alahmad, Miklós Neményi, and Anikó Nyéki. Applying iot sensors and big data to improve precision crop production: A review. *Agronomy*, 13(10), 2023.
- [14] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.
- [15] Xiaofei Xie, Guodong Zhao, Wei Wei, and Wei Huang. Mobilenetv2 accelerator for power and speed balanced embedded applications. In *2022 IEEE 2nd International Conference on Data Science and Computer Application (ICDSCA)*, pages 134–139, 2022.
- [16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [17] Matthew Wilkerson, Grace Vincent, Zaki Hasnain, Sambit Bhattacharya, and Emily Dunkel. Benchmarking resnet50 for image classification, Dec 2024.
- [18] Anis Rejeb, Karim Rejeb, and Harry B. Simske. Drones in agriculture: A review and bibliometric analysis. *Computers and Electronics in Agriculture*, 198:107017, 2022.
- [19] R. Rajagopal and S. Murugan. Artificial intelligence-based drone for early disease detection and precision pesticide management in cashew farming. *Agricultural Systems*, 190:103096, 2023.
- [20] R. Egi, T. Nakamura, and L. Xu. Drone-computer communication based tomato generative organ counting model using yolo v5 and deepsort. *IEEE Access*, 10:57834–57845, 2022.
- [21] Glenn Jocher. Yolo v5 by ultralytics, 2020.
- [22] Marius Hobart, Michael Pflanz, Nikos Tsoulias, Cornelia Weltzien, Mia Kopetzky, and Michael Schirrmann. Fruit detection and yield mass estimation from a uav based rgb dense cloud for an apple orchard. *Drones*, 9(1), 2025.
- [23] T. Wang, X. Liu, and Y. Zhang. Lightweight tomato ripeness detection algorithm based on the improved rt-detr. *Sensors*, 24(5):1897–1912, 2024.
- [24] Wenyu Lv, Shangliang Xu, Yian Zhao, Guanzhong Wang, Jinman Wei, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. Detsr beat yolos on real-time object detection, 2023.
- [25] M. Khan, A. Gupta, and P. Ramesh. Tomato maturity recognition with convolutional transformers. *International Journal of Computer Vision*, 132(3):412–430, 2023.
- [26] F. Zhang and L. Chen. A method for detecting tomato maturity based on deep learning. *IEEE Transactions on Automation Science and Engineering*, 21(2):334–345, 2024.
- [27] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023.
- [28] J. Hernández, P. Alvarez, and M. Gonzalez. Detection of tomato ripening stages using yolov3-tiny. *Journal of Agricultural Robotics*, 12(4):215–230, 2023.
- [29] Pranav Adarsh, Pratibha Rathi, and Manoj Kumar. Yolo v3-tiny: Object detection and recognition using one stage improved model. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 687–694, 2020.
- [30] RYZE Robotics. Tello. <https://www.ryzerobotics.com/fr/tello>.
- [31] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.
- [32] Hicham Slimani, Jamal El Mhamdi, and Abdelilah Jilbab. Deep learning structure for real-time crop monitoring based on neural architecture search and uav, Sep 2024.

Sim2Real Transfer for Vision-Based Grasp Verification

Pau Amargant^{1,2}, Peter Hönig¹, and Markus Vincze¹

Abstract—The verification of successful grasps is a crucial aspect of robot manipulation, particularly when handling deformable objects. Traditional methods relying on force and tactile sensors often struggle with deformable and non-rigid objects. In this work, we present a vision-based approach for grasp verification to determine whether the robotic gripper has successfully grasped an object. Our method employs a two-stage architecture; first YOLO-based object detection model to detect and locate the robot’s gripper and then a ResNet-based classifier determines the presence of an object. To address the limitations of real-world data capture, we introduce HSR-GraspSynth, a synthetic dataset designed to simulate diverse grasping scenarios. Furthermore, we explore the use of Visual Question Answering capabilities as a zero-shot baseline to which we compare our model. Experimental results demonstrate that our approach achieves high accuracy in real-world environments, with potential for integration into grasping pipelines. Code and datasets are publicly available at github.com/pauamargant/HSR-GraspSynth

Index Terms—Grasp verification, Robot manipulation, Deformable objects, Vision-based grasping, YOLO object detection, ResNet classification, Synthetic dataset, Visual Question Answering.

I. INTRODUCTION

Deformable object manipulation is a growing field of research in robotics due to its relevance in a wide range of tasks [26]. Deformable objects are a common occurrence in both industrial and household environments, and their manipulation poses challenges when compared to rigid objects. Their deformation and varying response to traditional force and tactile sensing methods during the grasping process introduce significant uncertainty, making it a more challenging task [25].

One critical aspect of deformable object manipulation is the verification of successful grasping. Traditional methods [1], which often rely on the object’s geometry and force and tactile sensors, struggle to account for the deformation of the object and its lack of internal structure and resistance [18]. This requires the use of more advanced sensors and control algorithms, which are often robot and situation specific.

In this context, computer vision has emerged as a promising tool to address these challenges. Various methods have been proposed to use 2D and 3D vision during the grasping process for tasks such as rope and cloth manipulation [12], [19]. These approaches use vision in combination with other input modalities such as tactile sensing to estimate the

object’s deformation during the grasping procedure. However, most proposed methods focus on the grasping control feedback and are object and task specific. These constraints and their complexity make these models unsuitable for the task of verifying a successful grasp.

This paper explores the application of computer vision for verifying whether a robot gripper has successfully grasped an object, with a focus on methods applicable to deformable objects. Our approach, which can be easily adapted to different robots and tasks, leverages object detection and machine learning to detect the grasping using the robot’s on device camera. Our main contributions are as follows:

- 1) We introduce a two-stage vision-based grasp verification model combining YOLO-based object detection and ResNet-based classification, improving generalization across different robotic platforms and object types.
- 2) We present HSR-GraspSynth, a synthetic dataset designed to simulate diverse grasping scenarios, addressing the limitations of real-world data collection and annotation.
- 3) We investigate the integration of Multimodal Large Language Models (LLMs) with Visual Question Answering (VQA) capabilities as a viable alternative for zero-shot learning in grasp verification.

II. RELATED WORK

Deformable object manipulation is an active area of research in robotics with wide practical applications [26]. Non-rigid objects are common in both industrial and domestic settings, making robots that can handle them especially useful. However, manipulating them poses additional challenges compared to rigid objects.

One of the most important aspects of the grasping pipeline is the ability to verify its success. Traditional methods use object geometry, force, tactile sensors [17], and proximity sensors [8], but often struggle to account for possible deformations and the lack of internal structure in deformable objects.

In this context, computer vision has been successfully applied to these challenges. 2D and 3D vision methods have been proposed for tasks like rope [12] and clothing manipulation [19]. These methods combine vision with other modalities to determine grasping poses and account for the object’s deformation during the procedure.

However, the majority of these methods are focused on grasping estimation and the control feedback and are robot, object and task specific. Computer vision solutions have been proposed as a simpler alternative for the task of verification [13]. In 2020, the use of low-cost machine vision

¹ All authors are with Faculty of Electrical Engineering, Technical University of Vienna, 1040 Vienna, Austria; {hoenig, vincze}@acin.ac.tuwien.at

² Pau Amargant is with Polytechnic University of Catalonia; pau.amargant@estudiantat.upc.edu

cameras installed in the robot gripper was studied [13]. They trained both YOLO [16] and MobileNet [6] models for this task, achieving high precision with different camera systems.

Inspired by this approach, we aim to develop a similar solution for robots with head-mounted cameras, such as the PAL Robotics Tiago, Toyota HSR, and Boston Dynamics' Atlas, enabling vision-based grasp verification without relying on gripper-mounted cameras.

A. Synthetic Datasets

Synthetic datasets have become increasingly popular for object grasping research [14]. With the advent of deep neural networks, the large amounts of data required make the use of synthetic data an attractive alternative to the laborious task of acquiring and annotating real world data.

Synthetic datasets have been widely used for training computer vision models for object detection and robotic tasks. Tools such as GrasPc [11] and BlenderProc [4] can be used to generate large-scale, photorealistic, and physics-aware datasets. By using commonly used object datasets such as YCB-V [24] and *ShapeNetV2* [2], synthetic data makes it possible to efficiently train models in zero-shot situations or when real data is costly to obtain. Common use cases are object detection and pose estimation [9], [22].

However, while there is a wide availability of synthetic datasets for grasp planning, there is a lack of datasets specifically designated for grasp verification. Current datasets do not capture the nuances of successful and failed grasps, such as occlusions, edge cases, and variations in sensor perspectives. This highlights a significant gap in the field and the need for dedicated synthetic datasets to support research in grasp verification.

III. HSR-GRASPSYNTH DATASET

Training a robust and generalizable model for grasp verification requires a diverse and extensive dataset. However, collecting real-world data is often expensive and time-consuming, making synthetic data an attractive alternative. Synthetic data should be diverse and similar enough to the real-world distribution in order to minimize the Sim2Real gap. [21], [23].

With these goals in mind, we created the HSR-GraspSynth dataset for grasp verification. It consists of annotated RGB images, referred to as *examples*, showing the HSR robot's gripper from the perspective of its head-mounted camera. Each example is annotated with a bounding box around the visible parts of the gripper and a binary label indicating whether an object is present in it (*object* or *no_object*).

Synthetic examples are generated from 3D simulated *scenes*, where a full environment including the robot and background distractors is randomly configured. Several examples are generated from the same scene, forming a *batch*.

The dataset consists of 12.000 examples and a separate validation dataset composed of 5.000 images.



Fig. 1: Examples from the proposed dataset. The top row shows examples with an object within the gripper while the lower row corresponds to no object. Each example corresponds to a different batch.

A. Data Generation

Synthetic data is generated using BlenderProc, a procedural pipeline that integrates Blender within Python to facilitate the rendering of large datasets.

For each batch of the dataset, a new scene is generated. A model of the robot is positioned at the centre of an enclosed room, with its arm extended in front of its head. To simulate realistic environments and obtain robust models, between 2 and 15 distractor objects are randomly scattered within the field of view of the robot using a physics-based algorithm to ensure physically plausible poses and varied object scales.

For training examples, distractors are sampled from the *ShapeNetV2* dataset, while for the validation set, objects from the YCB-V dataset are used.

Ten examples are generated per batch to improve computational efficiency and mitigate some of BlenderProc's limitations.

For each example within a batch, the robot's arm's pose is randomized by perturbing the positions of the arm joints and the camera orientation. A randomly sampled object is then placed within the robot's gripper with probability 0.5 to generate both *object* and *no_object* examples. The grasped object is sampled from the same dataset used for the distractor objects.

When an object is placed between the gripper fingers, the object is first moved away from the robot to avoid collisions, and the gripper fingers are partially closed to make contact with the object. A convex hull approximation is used to detect when the object collides with the gripper fingers. The gripper fingers are slowly closed until a collision is detected.

Fig. 1 shows six examples of rendered images of both classes. The gripper can be observed in several positions, with different distractor objects in the background.

IV. GRASPCHECKNET

The proposed approach for grasp verification is composed of a two-stage architecture that combines object detection and image classification. Object detection is used to localize the robot's gripper within the image. This makes the architecture adaptable to different robotic platforms and object

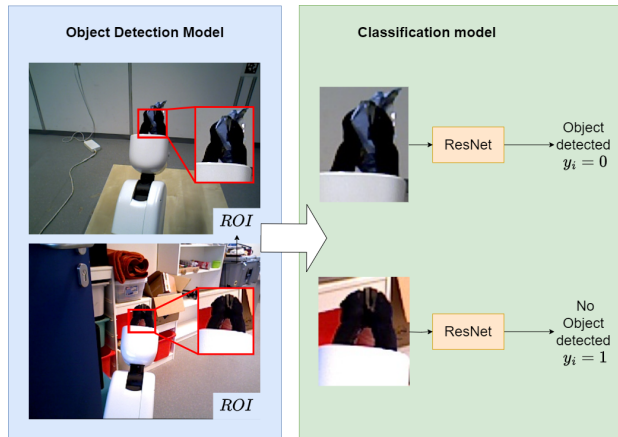


Fig. 2: Illustration of the two-stage model architecture using object detection and image classification. The YOLO object detection model localizes the robot's gripper in the image, and the ResNet classification model uses the cropped image to determine whether there is an object in the gripper.

types, while the image classification model verifies whether the grasp was successful.

A. Model Architecture

GraspCheckNet consists of two primary components; a YOLO-based object detection model and a ResNet-based image classification model. The object detection stage localizes the robot's gripper in the camera's field of view, while the classification model determines whether the gripper is holding an object.

The object detection model is a pretrained YOLO model fine-tuned on the HSR-GraspSynth dataset. The image classification model is based on a ResNet [5] architecture, and operates on the detected region of interest containing the gripper produced by the detector.

The presence of an object is formulated as a binary classification task, where a label of 0 indicates the presence of an object and a label of 1 signifies its absence.

This labeling scheme aligns with our objective of detecting unsuccessful grasp attempts.

An overview of the model's architecture is shown in Fig. 2.

The object detection step facilitates the classification task by eliminating irrelevant and unnecessary information from the image, focusing on the region of interest containing the gripper. Alternatively, this stage could be replaced by a geometric-based approach if the robot's characteristics and kinematics are well-defined and accessible or by incorporating a predefined pose following the grasping procedure, where the robot positions the gripper directly in front of the camera. However, these alternative approaches require a higher degree of integration and interaction with the grasping pipeline, as they interrupt the grasping process and require additional time.

V. EXPERIMENTS

To evaluate the performance of the proposed grasp verification model and the accompanying dataset, we conduct experiments on both synthetic and real-world data. The primary objectives are to assess the model's effectiveness in accurately detecting the gripper and determining its state, as well as to evaluate the domain gap between the synthetic and real domain. Additionally, we compare it with an LLM-based Visual Question Answering approach as a few-shot alternative.

A. Data Acquisition

To assess the model's performance in real-world conditions, a smaller evaluation dataset of real-world images is created using the robot's onboard RGB camera. Data collection is conducted in a room with furniture and domestic objects using Toyota's Human Support Robot (HSR).

The robot is placed in various environments with its arm extended, and its head-mounted camera oriented towards the gripper. Images are captured under different conditions, including scenarios where the gripper is empty and fully closed and others where it contains objects. A total of 518 real images, which we refer to as examples, are collected distributed as follows:

- 158 examples where the gripper is empty.
- 150 examples where the gripper holds 16 different rigid objects. A comprehensive set of YCB-V objects and other household items found within the label are used.
- 210 examples where the gripper holds 23 different deformable objects. Various household items such as clothes, papers, chip bags and tissues are used.

Each object is captured between 5 and 10 times. The robot is placed in various locations. The head is gradually rotated between consecutive captures of the same object to change the field of view and background.

B. Object Detection Model

We employ a fine-tuned YOLO11-l object detection model. The model is fine-tuned using Ultralytics' pretrained YOLO11-l [7] on the proposed synthetic HSR-GraspSynth dataset. To enhance the training process and mitigate the Sim-to-Real gap, various data augmentation techniques are applied during training. Used data augmentation techniques include perspective and affine transforms, and colour jitter, brightness and contrast changes and image compression.

The YOLO11-l model is fine-tuned for 100 epochs using Ultralytics' model trainer with default parameters. The best model in terms of mean Average Precision (mAP) on the validation set is kept after the training process.

Due to the Sim-to-Real gap, a low confidence threshold is required during inference on real images, leading to a large number of candidate detections distributed across different clusters in the image. To mitigate this issue, the confidence threshold is gradually reduced until detections appear.

When a low threshold value is used, a large number of detections localized around different clusters in the image can appear, leading to false positive detections. To mitigate

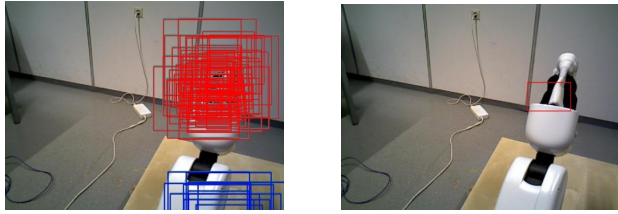


Fig. 3: Illustration of the clustering procedure applied to the detected bounding boxes. DBSCAN is used to identify clusters and assign a cluster label to each bounding box (left). Subsequently, the highest confidence bounding box from the cluster with the highest total confidence score is selected as the final detection (right).

this, we implement a post-processing refinement step after detection. Density-Based Spatial Clustering (DBSCAN) is used to identify clusters of detections within the image, as shown in Fig. 3. DBSCAN is preferred to other clustering methods such as K-means because it does not require a pre-defined number of clusters. The clusters are ranked according to the cumulative confidence scores of the bounding boxes they contain. The final detection is selected as the highest confidence bounding box within the highest ranked cluster.

C. Image Classification

The image classification model is responsible for determining whether the gripper contains an object or is empty. For this task, a pretrained ResNet-18 model is employed. Within the wider ResNet model family, the ResNet-18 was chosen for having fewer parameters than the bigger models of its family, making faster during training and inference.

The model's head is adapted for the task of binary classification. The original head is replaced by two fully connected layers with ReLU activation functions and dropout layers in-between [20].

The model is trained using ground truth cropped synthetic images containing the robot's gripper. Data augmentation techniques are used to make the model robust to the synthetic to real domain transfer. The model is trained using an Nvidia A40 GPU in different stages. First, only the head is trained using a large dropout rate of 0.7 to 0.5 to make the model more robust. Afterwards, the learning and dropout rates are decreased while also unfreezing the backbone's last layer.

D. Real-world evaluation

To validate the model's effectiveness in real-world conditions, a qualitative evaluation is conducted. First, the object detection model (stage 1) is evaluated independently, followed by the evaluation of the classification module (stage 2) using the detections as input.

1) *Object Detection Model*: We first evaluate the ability of the detection model to localize the gripper within the image. Intersection over Union thresholds are not used for the assessment. Instead, the detections are qualitatively assessed based on whether the bounding boxes sufficiently localize and encompass the robot's gripper. The fine-tuned YOLO

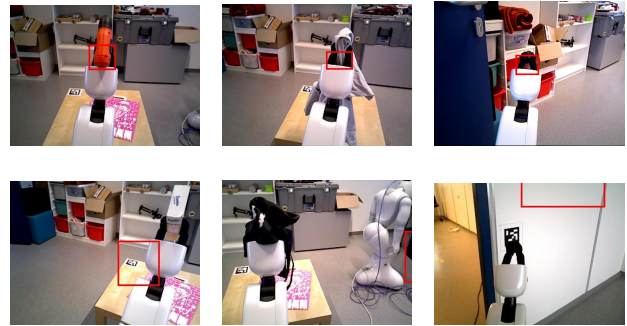


Fig. 4: Sample detections using the object detection model. The red rectangle shows the detected bounding box. Top and lower rows show correct and incorrect detections respectively.

model is used to obtain the gripper's bounding box for each of the 518 test images. Each detection is manually reviewed and considered correct if it contains, at least partially, both gripper fingers and the majority of the bounding box area corresponds to the gripper and object, with limited inclusion of background regions. Detections that mostly contain the background or fail to include both gripper fingers are labeled as incorrect.

Table I shows the results of this evaluation. We observe that the model is able to properly locate the gripper within the image in 98% of *no_object* examples, 94.67% of examples where there is a rigid object within the gripper and 96.67% of examples when there is a deformable object in the gripper.

Nevertheless, as shown in Fig. 4 the evaluation is limited by not taking into account the IoU. During the experiments it was observed that predicted bounding boxes tend to properly contain the gripper in terms of width, but often do not fully encompass it on the vertical axis. During inference we mitigate it by padding the detected bounding box.

2) *Image Classification Model*: The second stage of the model is evaluated using the cropped images obtained from the object detection outputs. Detected bounding boxes are used to crop the image region containing the gripper, with additional margins added to compensate for possible errors.

In order to account for the gap between synthetic and real data, the probability threshold for assigning to label 1 is

TABLE I: Evaluation of the object detection model on the real-world dataset. Num. detected refers to the number of examples where the predicted bounding box is qualitatively correct. Percentage of objects correct refers to individual objects that have been detected correctly in all the examples they appear in the dataset.

Category	Num. Images	Num. Detected	% Detected	% Objects Correct
No Object	158	155	98.10	N/A
Rigid	150	142	94.67	62.50
Deformable	210	203	96.67	82.61

lowered to 0.15 from 0.5 during evaluation.

Table II shows the evaluation metrics of the model across the *no_object* and *object* categories, and differentiating by rigid and deformable object. While table III contains the precision and recall values for the task of detecting *no_object* instances.

The model has an accuracy of 74.7% in examples where the gripper is empty and on 82.9% of deformable object instances, indicating that it is able to properly recognize non-rigid objects within the gripper.

In terms of detecting that the gripper is empty, as a class of binary classification, it achieves a precision value of only 0.678 albeit with a higher recall value. This lower precision value is indicative of the presence of false positives during the detection. In our case, in which we are mostly interested in detecting failed grasps, the focus is on obtaining a higher recall value.

In terms of execution time, the average inference time is of 28ms on an A40 GPU, when not taking into account the initial mode loading time. Extrapolating this result to less powerful devices indicates the suitability of the model for low-latency applications.

TABLE II: Classification accuracy per category. **Bold** indicates maximum, underline indicates minimum performance.

Category	GraspCheckNet Accuracy (%)	GPT4-o Accuracy (%)	Llama 3.2 11B Accuracy (%)
No Object	74.7	95.0	<u>48.7</u>
Rigid	86.7	95.3	<u>68.7</u>
Deformable	82.9	78.1	<u>60.0</u>

TABLE III: Precision and Recall score per model. **Bold** indicates maximum, underline indicates minimum performance.

Model	Precision	Recall
GraspCheckNet	0.678	0.749
GPT-4o	0.739	0.95
Llama 3.2	<u>0.357</u>	<u>0.513</u>

E. Visual Question Answering

In order to establish a baseline to which compare our GraspCheckNet model, we evaluate the use of state-of-the-art LLMs for Visual Question Answering as a zero-shot method for image classification. Our goal is to leverage their state-of-the-art performance in visual reasoning tasks to evaluate our model's performance.

We follow a visual-question-answering approach in which an LLM is prompted with the task that it should do and how to reply to it. The same prompt is used for all instances and no concrete information about the object in the gripper was included even though it might be available in certain grasping pipelines. We evaluate two LLMs to compare the effects of the model size and whether on-device models are able to successfully complete the task. We test GPT4-o [15] and llama 3.2 Vision 11B [10]. GPT4-o is tested using OpenIA's API while Llama 3.2 Vision is used through

UnSloth's implementation of the model [3], which reduces its memory footprint. We use both the state of the art GPT4-o, which is closed-source and has large memory requirements, and Llama 3.2 in its 11B parameters version. This latter model can be run in consumer devices with approximately 6GB of GPU or unified system memory, making it feasible to deploy in practical scenarios.

Table II shows the results of evaluating the VQA models on the real-world evaluation dataset. Llama was not able to successfully perform the VQA task, achieving a recall of only 0.513. When asked concrete questions about the images, the model often produces hallucinations or does not correctly understand the scene. This indicates that further advancements in vision LLMs or the use of larger models is required.

On the other hand, GPT4-o is able to correctly detect most instances of the gripper being empty, with a recall of 0.95. However, it shows a relatively large amount of false positives in deformable objects, recognizing the gripper as empty. This does not happen uniformly across all objects. It is not able to properly detect some clothing items such as a black glove, a kitchen drape, a t-shirt and a hat, which account for 33 out of the 46 wrong classifications of deformable objects. These objects present uniform textures, without defining features, and when grasped by the gripper they do not hold a recognizable shape. This might indicate that the vision model focus on the detection of an object and not in identifying whether there is anything in the gripper, making it susceptible to false positives when there are difficult to recognize objects. It presents a higher amount of false positives, deformable object instances classified as empty, than our model while it has a lower amount of false negatives, instances of empty gripper classified as not empty.

The use of vision language models, even when using smaller models such as Llama 3.2 11B, requires expensive compute and requires more execution time than our proposed model. Open AI's GPT4-o required on average 2.27 seconds per image, albeit with a high standard deviation of 1.53 seconds. Due to the use of a remotely hosted API, the model's latency can often not be stable and a stable internet connection is required. The higher latency and requirement for internet connection makes this method less reliable for real-world applications. In terms of cost, each instance costs approximately 0.001€ to execute. While this cost is relatively low, it can quickly scale up if a large amount of classifications is required.

When compared to our model, GraspCheckNet offers lower recall but can be run on-device with a lower inference time, making it feasible for low-latency applications and integration within grasping pipelines. Our model achieves comparable performance in detecting the presence of deformable objects but is less accurate to detect the gripper being empty.

VI. CONCLUSION AND FUTURE WORK

This paper presents GraspCheckNet, a vision-based approach for grasp verification using head-mounted cameras,

with a particular emphasis on deformable object manipulation. Our two-stage architecture uses object detection and image classification to verify successful grasps, addressing the challenges posed by non-rigid and deformable objects. We introduce HSR-GraspSynth, a synthetic dataset for training grasp verification models and help address the limitations of real-world data acquisition and reduce the Sim2Real gap. Experimental results demonstrate that the proposed approach properly detects the presence of an object within the robot's gripper, particularly for deformable objects. Our approach maintains consistence performance while offering significant advantages in terms of inference and the ability to run on-device without requiring external APIs.

Future work should focus on the integration within grasping pipelines, exploring how real-time verification can be of use and how a better integration with the pipeline can be used to increase the model's accuracy. Furthermore, domain-adaptation techniques, both supervised and unsupervised could be explored to mitigate the Sim2Real gap.

ACKNOWLEDGMENT

This research is supported by the EU program EC Horizon 2020 for Research and Innovation under grant agreement No. 101017089, project TraceBot, and the Austrian Science Fund (FWF), under project No. I 6114, iChores.

REFERENCES

- [1] A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, 2000, pp. 348–353 vol.1.
- [2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," Dec. 2015.
- [3] M. H. Daniel Han and U. team, "Unslloth," 2023. [Online]. Available: <http://github.com/unsllothai/unslloth>
- [4] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam, "BlenderProc," Oct. 2019.
- [5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.
- [6] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017.
- [7] G. Jocher, J. Qiu, and A. Chaurasia, "Ultralytics YOLO," Jan. 2023. [Online]. Available: <https://ultralytics.com>
- [8] P. Kulkarni, S. Schneider, and P. G. Ploeger, "Low-Cost Sensor Integration for Robust Grasping with Flexible Robotic Fingers," in *Advances and Trends in Artificial Intelligence. From Theory to Practice*, F. Wotawa, G. Friedrich, I. Pill, R. Koitz-Hristov, and M. Ali, Eds. Cham: Springer International Publishing, 2019, pp. 666–673.
- [9] S. Lin, K. Wang, X. Zeng, and R. Zhao, "Explore the Power of Synthetic Data on Few-shot Object Detection," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023, pp. 638–647.
- [10] Meta, "The Llama 3 Herd of Models," 2024.
- [11] A. Miller and P. Allen, "Graspit! A versatile simulator for robotic grasping," *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 110–122, Dec. 2004.
- [12] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2146–2153.
- [13] D. Nair, A. Pakdaman, and P. G. Plöger, "Performance Evaluation of Low-Cost Machine Vision Cameras for Image-Based Grasp Verification," Mar. 2020.
- [14] R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic, D. Fox, and A. Cosgun, "Deep Learning Approaches to Grasp Synthesis: A Review," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3994–4015, Oct. 2023.
- [15] OpenAI, "GPT-4o System Card," 2024, _eprint: 2410.21276. [Online]. Available: <https://arxiv.org/abs/2410.21276>
- [16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 779–788.
- [17] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, "Human-Inspired Robotic Grasp Control With Tactile Sensing," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1067–1079, Dec. 2011.
- [18] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *The International Journal of Robotics Research*, June 2018.
- [19] L. Sun, G. Aragon-Camarasa, S. Rogers, and J. P. Siebert, "Accurate garment surface analysis using an active stereo robot head with application to dual-arm flattening," May 2015, pp. 185–192.
- [20] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022.
- [21] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World," Mar. 2017.
- [22] J. Tremblay, T. To, and S. Birchfield, "Falling Things: A Synthetic Dataset for 3D Object Detection and Pose Estimation," July 2018.
- [23] J.-B. Weibel, R. Rohrböck, and M. Vincze, "Measuring the Sim2Real Gap in 3D Object Classification for Different 3D Data Representation," in *Computer Vision Systems*, M. Vincze, T. Patten, H. I. Christensen, L. Nalpantidis, and M. Liu, Eds. Springer International Publishing, 2021, pp. 107–116.
- [24] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes," in *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation, June 2018.
- [25] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Science Robotics*, vol. 6, no. 54, 2021.
- [26] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, J. Kober, X. Li, J. Pan, W. Yuan, and M. Gienger, "Challenges and Outlook in Robotic Manipulation of Deformable Objects," *IEEE Robotics & Automation Magazine*, vol. 29, no. 3, pp. 67–77, Sept. 2022.

LLM-Empowered Embodied Agent for Memory-Augmented Task Planning in Household Robotics

Marc Glocker^{1,2}, Peter Hönig¹, Matthias Hirschmanner¹, and Markus Vincze¹

Abstract—We present an embodied robotic system with an LLM-driven agent-orchestration architecture for autonomous household object management. The system integrates memory-augmented task planning, enabling robots to execute high-level user commands while tracking past actions. It employs three specialized agents: a routing agent, a task planning agent, and a knowledge base agent, each powered by task-specific LLMs. By leveraging in-context learning, our system avoids the need for explicit model training. RAG enables the system to retrieve context from past interactions, enhancing long-term object tracking. A combination of Grounded SAM and LLaMa3.2-Vision provides robust object detection, facilitating semantic scene understanding for task planning. Evaluation across three household scenarios demonstrates high task planning accuracy and an improvement in memory recall due to RAG. Specifically, Qwen2.5 yields best performance for specialized agents, while LLaMA3.1 excels in routing tasks. The source code is available at: <https://github.com/marc1198/chat-hsr>

Index Terms—Embodied AI, Task Planning, Memory Retrieval

I. INTRODUCTION

Despite recent progress in robotics and artificial intelligence, robots still struggle to adapt flexibly to the diverse, dynamic situations of real-world environments, particularly in household settings [24]. While symbolic task planning with languages like the Planning Domain Definition Language (PDDL) [11] is effective in domains with fixed rules and predictable object categories, it lacks the adaptability required for open-ended household environments. In such settings, robots must deal with ambiguous user commands, detect novel or unstructured objects, and respond to constantly changing spatial configurations [24]. These limitations motivate our hypothesis that a modular LLM-driven system can enhance flexibility by leveraging natural language understanding, contextual reasoning, and memory-based adaptation. We provide a proof-of-concept implementation and assess its performance in real-world household tasks.

In this work, we present an embodied robotic system with an LLM-driven agent-orchestration architecture, where specialized software agents collaborate to address long-horizon household tasks. Recent advances in Large Language Models (LLMs) [13], [4], [15], [23], [5] have improved systems real-world understanding, enabling common-sense reasoning in

¹ Automation and Control Institute, Faculty of Electrical Engineering, TU Wien, 1040 Vienna, Austria {hoenig, hirschmanner, vincze}@acin.ac.tuwien.at

² AIT Austrian Institute of Technology GmbH, Center for Vision, Automation and Control, 1210 Vienna, Austria marc.glocker@ait.ac.at

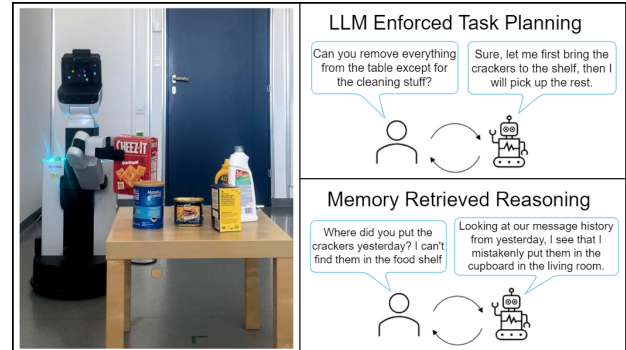


Fig. 1: Our LLM-driven robotic system autonomously plans tasks and retrieves past interactions to improve object handling, illustrated by LLM-enforced task planning and memory-retrieved reasoning in a household setting.

human language and making them accessible to researchers. These advances combined with in-context learning [26] enable flexible embodied task planning by decomposing high-level commands, such as “clear the dining table”, into actionable steps based on detected objects [2], [7], [25], [9], [21]. By integrating Grounded Segment Anything Model (Grounded SAM) [17] and LLaMa3.2-Vision [4], our system creates grounded task plans. Unlike most other works, we address long-term operations by maintaining action and environment records, utilizing Retrieval-Augmented Generation (RAG) for efficient memory retrieval. Our approach enables the robot to autonomously organize and retrieve objects, interpret complex tasks, and provide updates on object locations, all while ensuring privacy through the use of offline LLMs and avoiding explicit model training. To illustrate the systems interaction, Fig. 1 shows an example of our system in action.

In summary, we present the following key contributions:

- A long-horizon task planner for household tasks leveraging in-context learning and offline LLMs.
- Use of RAG for efficient memory retrieval and object tracking.
- A modular agent-orchestration system that improves robustness and modularity.
- Evaluation of the system’s performance in three real-world household scenarios.

This paper is structured as follows: Section II reviews related work in the areas of task planning and memory mechanisms. Section III details the proposed system architecture.

Section IV describes the experimental setup and household scenarios. Section V presents the results. Finally, Section VI concludes the paper and outlines directions for future work.

II. RELATED WORK

In this section we discuss related work for action and task planning, as well as memory and knowledge base.

A. Action and Task Planning

Recent advancements in prompt engineering have improved the problem-solving capabilities of LLMs [26], [28], enabling the generation of structured plans without fine-tuning. Consequently, modern agent architectures leverage LLMs to dynamically react to execution failures [27], [7] and expand their context by retrieval [8] or external tools [19], [18]. However, LLMs lack an inherent understanding of a robot's physical abilities and real-world constraints. *SayCan* [2] addresses this by integrating value functions of pre-trained robotic skills to ensure feasibility, whereas Huang et al. [6] leverage LLMs to match high-level plans with low-level actions through semantic mapping. Some works treat LLMs as programmers rather than direct decision-makers: *Code-as-Policies* [9] and *ProgPrompt* [21] allow LLMs to generate structured code for robotic executions, enhancing flexibility but adding an execution layer.

Pallagani et al. [14] found that LLMs perform better as translators of natural language into structured plans rather than generating plans from scratch. This ensures feasible actions based on predefined world models [20], [10]. These approaches are particularly effective in highly controlled environments, but present challenges when applied to open-ended, dynamic household settings. Our work, instead, embraces flexible, dynamic task planning with in-context learning like shown in [25]. The approaches named, while effective for short-horizon tasks, do not track object positions over time. For long-horizon tasks that involve real-world dynamic conditions, a combination of task planning and a memory mechanism is required.

B. Memory and Knowledge Base

Long-horizon tasks require robust memory mechanisms. While LLM context windows keep expanding [23], using excessively large contexts in robotics is computationally inefficient. Instead, long-term memory retrieval, accessed only when needed, is a more viable solution. RAG [8] provides an efficient mechanism for narrowing context by querying a vast dataset and retrieving only relevant information. Additionally, scene graphs, used in approaches like *SayPlan* [16] and *DELTA* [10], offer structured memory that improves action verification and contextual reasoning. However, in unstructured and constantly changing environments, maintaining these graphs becomes challenging due to the need for complex automatic mechanisms or manual curation.

Our work explores the feasibility of a lightweight, fully natural language-driven approach using RAG as a memory mechanism. Inspired by ReMemBR [1], our system incorporates temporal elements into the retrieval process, ensuring

the robot tracks long-term changes in its environment. While using language-based memory retrieval introduces potential for increased errors compared to structured models like scene graphs, we aim to evaluate how well purely language-based memory retrieval performs in practical, dynamic household scenarios. This approach offers flexibility, adaptability, and reduces the need for explicit world modelling, making it more suitable for real-world applications.

III. METHODOLOGY

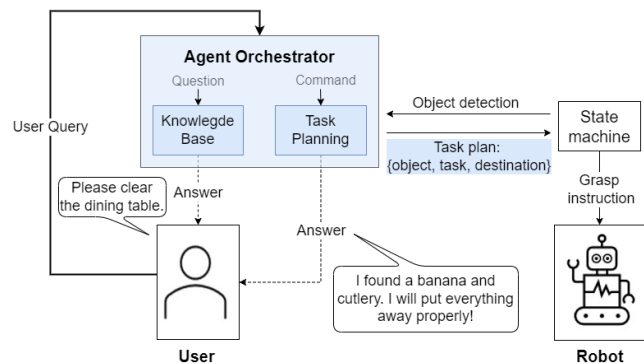


Fig. 2: The full pipeline, integrating long-horizon task planning. Newly introduced components are highlighted in blue.

Our system, coordinated by an agent-orchestration framework, combines task planning with RAG [8]. This chapter explains the individual components and their interaction.

Fig. 2 illustrates the overall pipeline. The focus of this work is the agent-orchestration system, which processes object detection and user requests to create a robot task plan. In the system, each agent uses an LLM with a specialized role. The task planning agent additionally is prompted with a chain-of-thought technique [26].

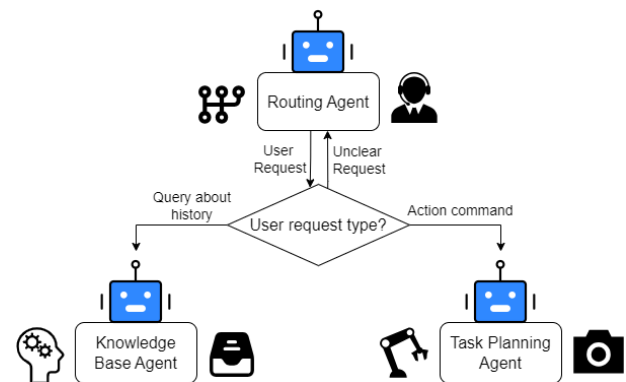


Fig. 3: The agent-orchestration architecture

The system architecture of the agent orchestrator, illustrated in Fig. 3, consists of:

- 1) A **routing agent**, responsible for analyzing incoming user requests.

- 2) A **task planning agent**, handling commands that require the robot to perform actions.
- 3) A **knowledge base agent**, processing follow-up questions about previously handled objects.

When a user request arrives, the routing agent first analyzes it to determine its nature. The request is then categorized into one of three types:

- 1) **Action command**: If the robot is asked to perform an action, it is forwarded to the task planning agent.
- 2) **Query about history**: If it concerns previously handled objects, it is directed to the knowledge base agent.
- 3) **Unclear request**: If the request doesn't fit either category, clarification is requested before proceeding.

A. Task Planning Agent

The task planning agent receives frequent environmental updates via camera perception, encoded as a list of single objects. Grounded SAM [17] enables text-driven object detection and segmentation for the pipeline, while Vision Language Models (VLMs) generate natural language descriptions of the environment. Although VLMs alone can extract the object list for the LLM, Grounded SAM is essential for precise segmentation, which is critical for grasping tasks. Using the object list, the LLM processes the user request – which can be both expressed in high-level or low-level terms – and formulates tasks that best fulfill the command. The generated answer has to include a JSON string for an action following this structure:

- 1) **Objects involved** in the task.
- 2) **The destination** for placement tasks.

After the action is determined, the grasping process is initiated. We use the segmentation from Grounded SAM and the camera intrinsics to crop the depth image and project the depth crop to a 3D pointcloud of the respective object. To estimate a grasp approach vector, we feed the cropped object point cloud to Control-GraspNet [22], a pre-trained grasp estimator.

B. Knowledge Base Agent

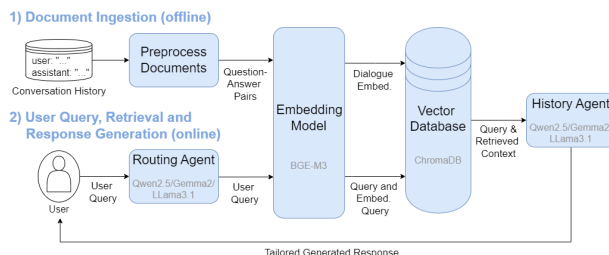


Fig. 4: RAG workflow for long-term question answering: Relevant past actions are retrieved from dialogue history, and the LLM generates responses based on the retrieved context.

The knowledge base agent is used for user inquiries regarding past robot actions, such as object locations or

task completion status. These queries require access to long-term memory, for which RAG has proven most effective, as discussed in Section II. Fig 4 illustrates the RAG workflow, comprising two key steps:

- 1) **Document Ingestion**: Input data, such as conversation history, is preprocessed, split into smaller chunks (each representing a question-answer pair), and converted into high-dimensional vectors using an embedding model. These embeddings are then stored in a vector database for efficient retrieval.
- 2) **User Query, Retrieval, and Response Generation**: User queries are embedded using the same model and are matched against the stored vectors to retrieve the most relevant context. This context is then provided to the LLM, which generates a response tailored to the user's query.

To enable chronological reasoning, essential for tracking object movements over time, we augment RAG with a time stamp for each question-answer pair.

IV. EXPERIMENTS

To evaluate our system, we conduct experiments addressing the three key challenges from Chapter I: (1) flexible task planning in dynamic household environments, (2) long-term memory usage, and (3) modular agent coordination. Specifically, we assess the system's ability to create grounded task plans, answer questions based on prior interactions, and route tasks to the appropriate agent.

A. Experimental Setup

This study evaluates an agent-orchestration system for symbolic task planning and follow-up questions via a knowledge base. To ensure a thorough evaluation, we consider three distinct phases:

- 1) **Task Planning Performance** – The symbolic task planning output is assessed independently, measuring accuracy of object assignment to their destinations.
- 2) **Knowledge Base Reliability** – The system's ability to reason about past actions (with and without RAG) is tested by asking about the system's current status, such as locations of previously moved items.
- 3) **Routing Reliability** – Measures the accuracy of the routing agent in directing queries to the appropriate agent (Task Planning, History, or itself).

To isolate the performance of the specialized agents, agent handoff is not considered in the evaluation of 1) and 2).

B. Algorithmic Framework

The frameworks and models used are shown in gray in Fig. 4. To enable efficient collaboration among agents, we use *OpenAI Swarm* [12], a lightweight framework for agent orchestration and task delegation. We evaluate the performance of *Qwen2.5-32b* [15], *Gemma2-27b* [23], and *LLaMa3.1-8b* [4], selected for their open-source availability and ability to run locally on 16GB GPU RAM. For RAG, we employ *ChromaDB* [3], a vector database optimized for fast lookups, combined with the embedding model *BGE-M3*.

C. Task Scenarios

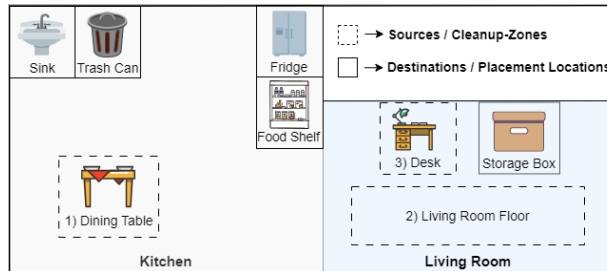


Fig. 5: The artificial household environment used in the experiment.

The experiment is conducted in an artificial household environment, where objects must be assigned to correct destinations based on high-level commands. To evaluate task planning, we define three scenarios (see Fig. 6) that share five predefined placement locations, while each uses a different cleanup zone. Fig. 5 shows a visual representation of the environment. These locations reflect common-sense knowledge typically understood by LLMs. To ensure clarity, the agent receives explicit definitions for each destination:

- **Sink** – For items that need washing.
- **Trash Can** – For disposable or inedible items.
- **Fridge** – For perishable food.
- **Food Shelf** – For non-perishable food items.
- **Storage Box** – For general storage.

Fig. 6 shows the object list extracted from a captured image of each task scenario using *LLaMa3.2-Vision* along with the user queries and the segmentation results from Grounded SAM.

After execution of all scenarios, the knowledge base agent is asked four distinct follow-up questions targeting different aspects of retrieval and reasoning:

- **Error Detection:** "Where is the jacket that was in the living room? I thought you put it in the storage box, but I can't find it there."
- **Hallucination:** "Where did you put the laptop? It's not on the desk anymore."
- **Food Availability:** "I am hungry. Is there any food left from earlier?"
- **Trash Status:** "How many objects are in the trash can?"

To better reflect real-world applications, we extend the conversation dialogue with additional question-answer pairs containing actions. Furthermore, deliberate errors are introduced into the task plans, where the agent provides the user a different location than the one forwarded to the state machine. This allows us to evaluate how well the knowledge base handles inaccuracies. Beyond evaluating the specialized agents in isolated setups, we assess how effectively the routing agent delegates tasks to the appropriate specialized agent. Specifically, we test:

- **Task Planning Queries:** The three high-level commands from the task planning scenarios (see Fig. 6) and an additional low-level request ("Can I have a banana?")

- **Knowledge Base Queries:** The four follow-up questions from the knowledge base scenario.

D. Evaluation Methodology

The evaluation of the agent-orchestration system's components is based on the task scenarios and follow-up questions defined in Section IV-C. Task planning performance is evaluated by testing each model on the three task scenarios, with each scenario executed five times per model. Accuracy is measured at the object level as the percentage of correctly assigned tasks. A task is deemed correct if it satisfies the following criteria:

- **Valid JSON format**
- **Correct destination assignment**
- **Stationary Object Exclusion** (ensuring no task is assigned to items that should remain in place)

The final accuracy score represents the percentage of objects for which tasks were correctly assigned, including the implicit "no task" assignment for stationary objects (e.g., table).

The knowledge base is evaluated using four follow-up questions, each tested five times per model. Unlike the task planning agent, the knowledge base agent does not require a strict output format. It is assessed based on factual correctness, measured as the percentage of correct answers. For queries expecting multiple objects as an answer (e.g., "Which objects are in the trash?"), accuracy is based on the percentage of correctly identified objects.

The routing agent's ability to correctly assign tasks is evaluated by processing queries from the task planning scenarios and history-based questions, along with one additional query, five times per model. The final metric is quantified as the percentage of correctly assigned tasks. Gemma2, which does not support tool calling, is excluded from this test.

V. RESULTS AND DISCUSSION

This section presents the experimental results for task planning, knowledge base and agent routing.

A. Task Planning

We introduce a *lenient* evaluation metric (cf. Table I), where reasonable alternative placements based on user preferences are counted as correct. The strictly correct placements, following the intended plan as prompted to the LLM, are presented under the *strict* metric in Table I.

Table I shows that *Qwen* consistently outperforms the other models in nearly all scenarios. *LLaMA* performs notably worse in the living room scenario, with the lowest strict accuracy (40.0%). *Gemma2* falls between the two, showing higher accuracy than *LLaMA* but lower than *Qwen*.

B. Knowledge Base

The integration of RAG notably enhances the accuracy of the knowledge base's responses, even in medium-term interactions consisting of 21 question-answer pairs with approximately 4000 tokens. *Qwen* achieves the highest validity (91.3%) with RAG (cf. Table II), highlighting the potential of retrieval-augmented approaches for maintaining consistency over longer interactions.

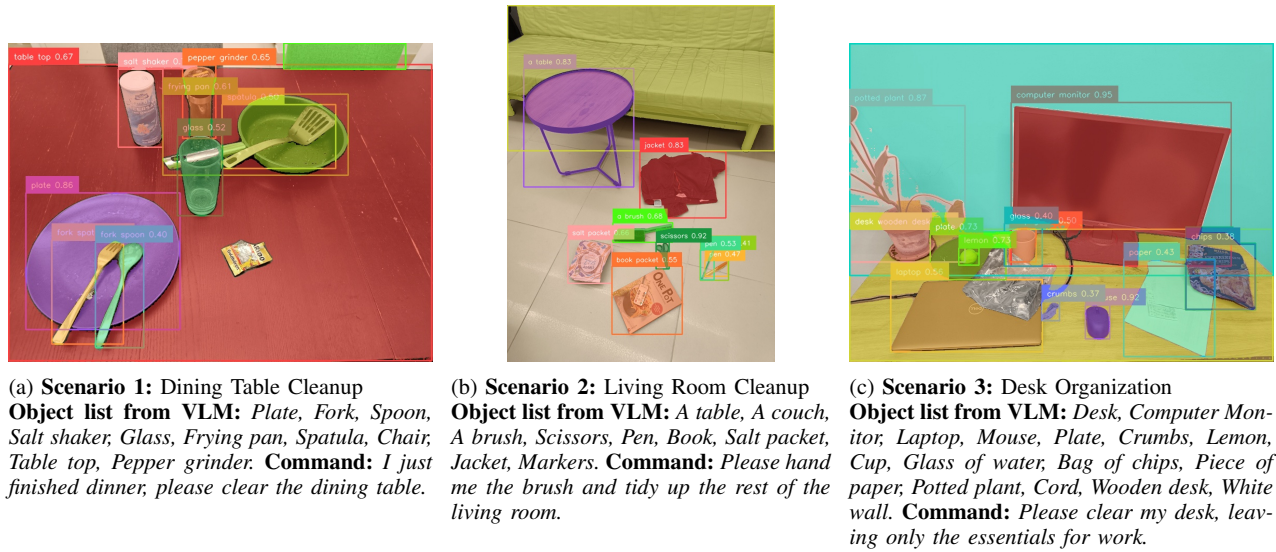


Fig. 6: The three scenarios used for task planning. For each scenario we have extracted an object list using the Vision-Language Model *LLaMa3.2-Vision*. This list is used as input for Grounded SAM [17] to perform segmentation.

Model	Dining Table		Living Room		Desk Organization		Total Accuracy (%)	
	Strict (%)	Lenient (%)	Strict (%)	Lenient (%)	Strict (%)	Lenient (%)	Strict (%)	Lenient (%)
LLaMa3.1-8B	68.0	78.0	40.0	40.0	61.3	65.3	56.4	61.1
Gemma2-27B	58.0	68.0	68.9	68.9	68.0	69.3	65.0	68.7
Qwen2.5-32B	64.0	80.0	88.9	88.9	78.7	84.0	77.2	84.3

TABLE I: Task Planning Accuracy Across Different LLMs. **Strict (%)**: Percentage of objects correctly placed according to the intended plan. **Lenient (%)**: Percentage of objects placed differently than expected, but with reasonable alternative placements based on user preferences.

Method	Model	Response Validity (%)				Total Validity (%)
		Err. Detection	Hallucination	Food Avail.	Trash Status	
Without RAG (Ablation Study)	LLaMa3.1-8B	20.0	80.0	70.0	65.0	58.8
	Gemma2-27B	0.0	80.0	10.0	60.0	37.5
	Qwen2.5-32B	0.0	80.0	60.0	75.0	53.75
With RAG	LLaMa3.1-8B	40.0	100.0	90.0	55.0	71.25
	Gemma2-27B	80.0	100.0	40.0	60.0	70.0
	Qwen2.5-32B	100.0	100.0	90.0	75.0	91.3

TABLE II: Knowledge Base Response Accuracy Across Different LLMs. **Used Embedding Model for RAG:** *BGE-M3*. **No. of question-answer pairs retrieved by RAG:** 5

C. Agent Routing

In task delegation, *LLaMA* exhibits the highest routing accuracy (92.5%), despite its weaker reasoning abilities (cf. Table III). Its structured approach to tool-calling ensures stable performance. In contrast, *Qwen*, while superior in contextual understanding, occasionally produces incorrect structured outputs, leading to execution failures.

D. Summary

Our findings highlight the potential of lightweight, open-source LLMs for memory-augmented long-horizon task planning. A combination of *LLaMA* (routing) and *Qwen* (specialized agents) achieves the best balance between structured

execution and high-level reasoning.

Evaluating task execution remains challenging due to subjective human preferences, emphasizing the need for user studies. Furthermore, integrating Vision-Language Models (VLMs) into the agent orchestrator – rather than only using them for object lists – could enhance robustness. Embedding contextual information into the latent space reduces command dependency and improves autonomy.

RAG improves factual consistency in knowledge retrieval but struggles with repeated object interactions and long histories, making full-history queries impractical. Scene graphs, as proposed by Liu et al. [10], present a promising alternative for efficient and robust knowledge integration.

Model	Task Planning Queries (%)	Knowledge Base Queries (%)	Total Success Rate (%)
LLaMa3.1-8B	85.0	100.0	92.5
Qwen2.5-32B	95.0	85.0	90.0

TABLE III: Routing Success Rate Across Different LLMs

While task delegation via the routing agent was mostly successful, certain models occasionally produced invalid structured outputs, leading to execution failures. To increase robustness, future work should explore schema validation and adaptive retry mechanisms that can automatically mitigate such issues.

In summary, open-source LLMs prove viable for long-horizon task planning. However, addressing key challenges – refining evaluation metrics, improving long-term robustness, and integrating multimodal perception – remains essential for achieving reliable household robotics.

VI. CONCLUSION

This work presents a prototype of an agent-orchestration system for household robots, utilizing local, lightweight open-source LLMs to translate high-level user commands into structured task plans for tidy-up scenarios. Memory-augmented task planning enables follow-up queries about past actions, improving user interaction and assisting in locating misplaced objects. Our evaluation shows strong task planning, routing, and knowledge retrieval with Qwen2.5 excelling in reasoning-heavy tasks and LLaMA3.1 providing a more efficient routing solution. However, RAG-based retrieval for general tasks remains a challenge, particularly for implicit queries where relevant information is not always found. Addressing these limitations is key to improving long-term reasoning and knowledge access.

Future work will focus on robust storage solutions, improved knowledge representations, broader user studies with structured datasets for evaluating and benchmarking existing approaches. Enhancing communication and tool usage in agent-orchestration will be crucial for greater adaptability and autonomy in household robotics.

ACKNOWLEDGMENT

This research is supported by the EU program EC Horizon 2020 for Research and Innovation under grant agreement No. 101017089, project TraceBot, and the Austrian Science Fund (FWF), under project No. I 6114, iChores.

REFERENCES

- [1] A. Anwar, J. Welsh, J. Biswas, S. Pouya, and Y. Chang, “Remembr: Building and reasoning over long-horizon spatio-temporal memory for robot navigation,” *arXiv preprint arXiv:2409.13682*, 2024.
- [2] A. Brohan, *et al.*, “Do as i can, not as i say: Grounding language in robotic affordances,” in *in Proceedings of the Conference on Robot Learning (CoRL)*, 2023, pp. 287–318.
- [3] Chroma, “Chromadb,” open-source vector database for AI applications. [Online]. Available: <https://www.trychroma.com/>
- [4] A. Dubey, *et al.*, “The llama 3 herd of models,” *arXiv preprint arXiv:2407.21783*, 2024.
- [5] D. Guo, *et al.*, “Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning,” *arXiv preprint arXiv:2501.12948*, 2025.
- [6] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, “Language models as zero-shot planners: Extracting actionable knowledge for embodied agents,” in *in Proceedings of the International Conference on Machine Learning (ICML)*, 2022, pp. 9118–9147.
- [7] W. Huang, *et al.*, “Inner monologue: Embodied reasoning through planning with language models,” *arXiv preprint arXiv:2207.05608*, 2022.
- [8] P. Lewis, *et al.*, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.
- [9] J. Liang, *et al.*, “Code as policies: Language model programs for embodied control,” in *in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.
- [10] Y. Liu, L. Palmieri, S. Koch, I. Georgievski, and M. Aiello, “Delta: Decomposed efficient long-term robot task planning using large language models,” *arXiv preprint arXiv:2404.03275*, 2024.
- [11] D. M. McDermott, “The 1998 AI planning systems competition,” *AI Magazine*, vol. 21, no. 2, p. 35, 2000.
- [12] OpenAI, “Swarm: Educational framework for multi-agent systems.” [Online]. Available: <https://github.com/openai/swarm>
- [13] OpenAI, *et al.*, “GPT-4 technical report,” 2024. [Online]. Available: <http://arxiv.org/abs/2303.08774>
- [14] V. Pallagani, *et al.*, “On the prospects of incorporating large language models (llms) in automated planning and scheduling (aps),” in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 34, 2024, pp. 432–444.
- [15] Qwen, *et al.*, “Qwen2.5 technical report,” 2025. [Online]. Available: <http://arxiv.org/abs/2412.15115>
- [16] K. Rana, *et al.*, “Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning,” *arXiv preprint arXiv:2307.06135*, 2023.
- [17] T. Ren, *et al.*, “Grounded sam: Assembling open-world models for diverse visual tasks,” *arXiv preprint arXiv:2401.14159*, 2024.
- [18] J. Ruan, *et al.*, “Tptu: Task planning and tool usage of large language model-based ai agents,” in *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.
- [19] T. Schick, *et al.*, “Toolformer: Language models can teach themselves to use tools,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 68 539–68 551, 2023.
- [20] T. Silver, *et al.*, “Generalized planning in pddl domains with pretrained large language models,” in *in Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, 2024, pp. 20 256–20 264.
- [21] I. Singh, *et al.*, “Progprompt: Generating situated robot task plans using large language models,” in *in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530.
- [22] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, “Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes,” in *in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 438–13 444.
- [23] G. Team, *et al.*, “Gemma 2: Improving open language models at a practical size,” *arXiv preprint arXiv:2408.00118*, 2024.
- [24] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, “Robots that use language,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 25–55, 2020.
- [25] S. H. Vemprala, R. Bonatti, A. Buckner, and A. Kapoor, “Chatgpt for robotics: Design principles and model abilities,” *Ieee Access*, 2024.
- [26] J. Wei, *et al.*, “Chain-of-thought prompting elicits reasoning in large language models,” *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.
- [27] S. Yao, *et al.*, “React: Synergizing reasoning and acting in language models,” in *in Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.
- [28] D. Zhou, *et al.*, “Least-to-most prompting enables complex reasoning in large language models,” *arXiv preprint arXiv:2205.10625*, 2022.

Low-Cost Open-Source Real-Time Communication in Industrial IoT: Using the Raspberry Pi 5 with OPC UA over TSN*

Jonathan Lukas Mandl¹, Olaf Saßnick², and Thomas Rosenstatter³

Abstract— Industry 4.0 demands optimized industrial process control and enhanced data permeability, necessitating more capable edge-level hardware. While personal computers with real-time Linux operating systems offer ample computing power at low cost, they suffer from limitations in low-level connectivity, standardized compact form factors, and uncertain long-term supplies. This paper explores the Raspberry Pi 5 as a viable alternative, highlighting its excellent low-level connectivity, compact form factor, and guaranteed long-term availability until 2036.

This study investigates the Raspberry Pi 5's performance in real-time communication scenarios using Open Platform Communications Unified Architecture (OPC UA) over Time-Sensitive Networking (TSN), a critical requirement for industrial applications. By outlining necessary modifications to the Raspberry Pi 5 and its Linux kernel, we enable real-time communication via TSN. Performance measurements in an OPC UA PubSub scenario are then compared with industrial PCs, highlighting the Raspberry Pi 5's potential as an alternative edge device.

Index Terms— Industry 4.0, industrial communication, time-sensitive networking, commodity hardware

I. INTRODUCTION

The rising need for optimized industrial process control and data permeability driven by the Industry 4.0 initiative necessitates more capable hardware at the edge-level. While personal computers with real-time Linux operating systems could provide plenty of computing power at a low cost, they come with downsides, namely: (i) a lack of low-level connectivity, (ii) limited options for standardized compact form factors, and (iii) uncertain long-term supplies.

Connectivity. Basic interfaces are missing to interface sensors, such as RS485, One-Wire, and also General-Purpose Input/Output (GPIO). One can resort to USB adapters, which, however, contribute to latency and introduce new sources of failure.

Form factor. The smallest widespread standardized form factor, Mini-ITX, still occupies a considerable amount of space and is therefore unsuitable for numerous applications.

Long-term supplies. Although personal computers are readily available, new generations are typically introduced

every 1-2 years, phasing out the previous models. Additionally, unannounced hardware revisions can significantly impact usability in industrial systems. Once a model has been thoroughly tested, it is undesirable to change and retest it.

Motivation. Considering all three options, the Raspberry Pi 5 is a viable alternative. It offers excellent low-level connectivity in a compact form factor and guarantees long-term supplies, as it will remain in production until 2036. Additionally, the Compute Module 5 [12], which features a high-density perpendicular connector, uses the same hardware. This module simplifies the hardware design process for custom solutions, allowing for a 2-layer printed circuit board. Since the NIC and processor are identical, all findings in this work are applicable to the Compute Module 5.

The Raspberry Pi 5 presents a strong candidate, given the benefits mentioned earlier. Especially in resource-constrained environments like mobile robotics, where the form factor and energy efficiency are crucial, this system could prove highly beneficial. However, it is essential to evaluate its performance in real-time communication scenarios, as this is a critical requirement for industrial applications.

Contribution. In this paper we investigate two research questions (RQs) focusing on the Raspberry Pi 5's performance with OPC UA Time-Sensitive Networking (TSN) and further compare it to industrial PCs.

- **RQ1.** How does the Raspberry Pi 5 perform in real-time communication tasks using OPC UA over TSN?
- **RQ2.** How does the Raspberry Pi 5's performance as a real-time OPC UA node compare to industrial PCs?

By addressing the above-mentioned research questions, we first outline the necessary modifications to the Raspberry Pi 5 and its Linux kernel. These modifications enable real-time communication via TSN. After setting up the Raspberry Pi 5 devices, we measure their performance in a OPC UA PubSub scenario (*RQ1*). We compare this performance with industrial PCs, which serve as a baseline. This comparison highlights the Pi 5's potential as an alternative edge device (*RQ2*).

II. BACKGROUND

A. Raspberry Pi 5 Hardware

The Raspberry Pi 5, hereafter referred to as the Pi 5, is a single-board computer from the Raspberry Pi Foundation. The term single-board computer describes the hardware, as it is integrated onto a single circuit board. Compared to its predecessor, the Pi 4, the Raspberry Pi Foundation reports a two to threefold increase in performance. Furthermore, the

*The industrial computers used in this work were provided by the Open Source Automation Development Lab (OSADL) eG.

¹Jonathan Lukas Mandl is a master student in Industrial Informatics & Robotics at the Salzburg University of Applied Sciences, AT, jmandl.iirb-m2024@fh-salzburg.ac.at

²Olaf Saßnick is with Josef Ressel Centre for Intelligent and Secure Industrial Automation, Salzburg University of Applied Sciences, AT, olaf.sassnick@fh-salzburg.ac.at

³Thomas Rosenstatter is with Josef Ressel Centre for Intelligent and Secure Industrial Automation, Salzburg University of Applied Sciences, AT, thomas.rosenstatter@fh-salzburg.ac.at

cache layout is improved on the Pi 5. L3 cache is added and the L2 cache is changed from shared to per-core cache. This helps with the isolation of CPU cores and their caches, improving timing determinism in real-time systems [15].

For this work, the Ethernet peripherals are more critical than the chosen processor. The Pi 5 features, in comparison to its predecessor, utilizes an in-house developed south bridge that connects the Ethernet MAC peripherals via PCI-E. This setup includes the Cadence Gigabit Ethernet MAC design of type GEM.GXL 1p09, which supports IEEE 1588 for precise time synchronization, a standard used in Precision Time Protocols (PTPs) applications. This design allows for time-stamping of packets, which is crucial for applications requiring synchronized timing, such as industrial automation and real-time communication systems [14].

B. Precision Time Protocol

The PTP protocol enables precise synchronization of the clocks of multiple devices in the same network. It uses a master slave topology to determine the exact travel delay between two devices which is then used to synchronize the slave clock to the master clock. Due to the fundamental properties of the protocol, hardware support from the Ethernet MAC is needed for precise synchronization. In this case the Ethernet MAC uses an internal clock inside the Network Interface Card (NIC) to timestamp incoming and outgoing packets. While software timestamping happens inside the Linux kernel and therefore adds operating system latencies. The Linux implementation uses two services: `ptp4l` is used for performing the PTP protocol, `phc2sys` synchronizes the system clock to the PTP hardware clock used for timestamping [5].

C. Time-Sensitive Networking

Time-Sensitive Networking is a series of standards that add to the Ethernet standard to improve the real-time performance of Ethernet. It mainly addresses two important features needed for real-time applications: time synchronization and traffic shaping [2].

Time synchronization and more specifically PTP, synchronizes the clocks of the nodes in a TSN network. Clock synchronization is needed to ensure that time driven communication can be correctly performed. Shaping algorithms also depend on a common and accurate time.

This work focuses solely on the time synchronization part of the TSN standard. Shaping is largely dependent on the implementing software. In the experimental setup there are only two TSN capable devices and no additional non-real-time traffic which would make shaping necessary.

D. PREEMPT_RT-patched Linux kernel

The experiments in this work were done on the version 6.6.23 and 6.6.78 of the Linux kernel and therefore needed the PREEMPT_RT patches for real-time support. The patches 6.6.23-rt28 and 6.6.78-rt51 were used. Even though dynamic preemption in the Linux kernel reduces the latency of tasks a lot, full preemption is needed to minimize latency and improve consistency of process wake up times. The patch

achieves this mostly by removing or altering non preemptable kernel code. Another significant change is the adoption of threaded interrupt handling, which allows higher priority interrupts to interrupt lower priority interrupt handlers.

Without threaded interrupt handlers the latency of network interrupts would be less predictable and generally higher. The real-time performance of the patched Linux kernel was tested with the `cyclicttest` utility and a synthetic system load. The `cyclicttest` utility from the `rt-tests` package uses `clock_nanosleep` to suspend a measuring thread. By calculating the deviation from the expected wake-up time, the utility calculates the systems latencies [19]. Furthermore, the load generating tool `stress` was used for synthetic CPU and memory load. This load ensures that the Linux kernel has to be preempted during the test [18].

E. Open Platform Communications Unified Architecture

OPC UA represents an evolution of the OPC standard which consolidates the previous OPC Classic specifications into a platform-independent framework. OPC UA is a data exchange standard that supports a variety of functions, including data transfer, method calls, and other capabilities. The OPC Foundation describes its primary use case as enabling communication from machine-to-machine and machine-to-enterprise, as well as bridging the two. A key feature is its ability to semantically describe data and organize it within complex, object-oriented structures. While OPC UA offers a wide range of functionalities, this paper focuses on its role as a foundation for data transfer.

Furthermore, the OPC UA application used in this work leverages the PubSub mechanism to transfer data between nodes. The code is provided by Pfrommer et al. [10] and is available in the `open62541` library until version 1.4

III. RELATED WORK

The Open Source Automation Development Lab eG (OS-ADL) is a laboratory dedicated to providing open-source software solutions for industrial systems. They have projects focused on real-time networking with various protocols, including OPC UA PubSub over TSN. In contrast to this paper, their research focuses on the feasibility of open-source software in industrial applications. Measurements from their project [7] can be compared with the results presented in this paper.

The work by Ulbricht et al. [17] describes TSN-FlexTest, a flexible testbed for TSN measurements. This testbed utilizes commodity off-the-shelf hardware and focuses on the TSN communication itself. They also use the same NIC (Intel I210) as the computers for our baseline comparison.

While other studies have utilized the Raspberry Pi in industrial settings using OPC UA, they do not focus on OPC UA over TSN (e.g., [6], [4]). An exception is the work by Reddy et al. [16], who employ a Raspberry Pi 3 only as a subscriber in a TSN network, although they do not provide performance details.

We have not found any comparable research investigating the potential of the Pi 5 computer respectively the Compute

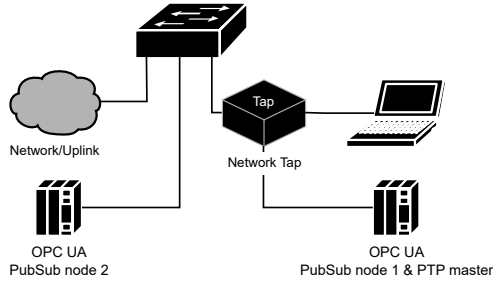


Fig. 1. Network topology of the experiment setup.

Module 5 as a replacement for industrial computers in real-time networking using TSN.

IV. EXPERIMENT SETUP

The setup comprises two Pi 5 computers, a consumer grade network switch and a network tap. A dedicated network tap is utilized to measure and improve the accuracy of packet timestamps as well as to include jitter of hardware latencies of the computers used. The Pi 5 computers are configured as OPC UA PubSub nodes. Both act as a subscriber and a publisher as presented in the TSN example program¹ in the open62541 library (until Version 1.4) [10]. Moreover, for clock synchronization between the two TSN nodes, one of the OPC UA PubSub nodes acts as a PTP master for the other node.

A. Hardware Details

In addition to this setup, two industrial PCs are used for baseline measurements. These PCs are equipped with Intel i210 NICs, which support optimization for TSN networks. The two primary settings for tuning are: defining a launch time for packets (`SO_TXTIME`) and use of multiple hardware queues for different priority packets. Additionally, in the baseline measurements, one industry PC was also used as the PTP master. This was done in order to remove all Pi 5 out of the tests and should not change the outcome. This difference also highlights that the industrial PCs can operate at very short cycle times with low jitter, as shown in Section V.

It should be noted that the used layer 2 switch does not support all features of the TSN specification. But this reinforces the purpose of this paper to use commodity off-the-shelf hardware for evaluating the performance of real-time networking. Figure 1 illustrates the testing setup. The illustrated OPC-UA nodes are the Pi 5 computers respectively the industrial PCs.

Following is a list of the hardware used in the test setup:

- ProfiShark 1G network tap
- TP-Link TL-SG105E 5 port unmanaged switch
- Raspberry Pi 5 with 4GB memory

For baseline testing with industrial PCs instead of the Pi 5 computers:

¹https://github.com/open62541/open62541/tree/v1.3.10/examples/pubsub_realtime (2025-02-06)

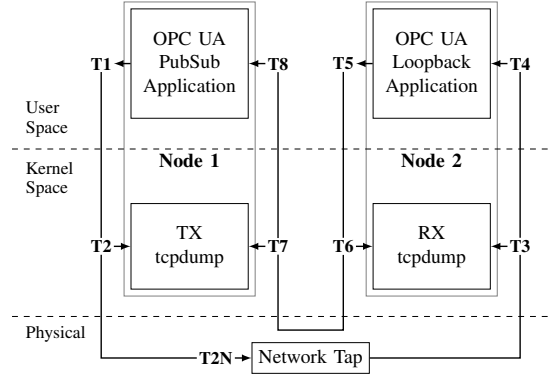


Fig. 2. Overview of the test points used for logging processing times, based on [10], with the addition of **T2N**, provided via a network tap.

- Schubert Prime Box Pico with Intel Atom Processor E3950 and two Intel i210 NICs

B. System Modifications to Pi 5

To further improve the real-time networking performance on the Pi 5 several modifications were applied to the system.

- *Deactivating Energy-Efficient Ethernet:* The Energy-Efficient-Ethernet protocol can be deactivated via an entry in the boot configuration, which means that the network card can no longer switch to an energy-saving mode when there is no network traffic. This energy-saving mode can result in added latencies because of the wake-up time of the NIC [13].
- *Ethernet coalescence:* Even though the Ethernet MAC on the Pi 5 does not support all features which could improve performance for real-time networking, some settings are available. `rx-usecs` and `tx-usecs` are available settings. Both settings should be lowered to reduce the amount of microseconds the Ethernet NIC waits until triggering an interrupt for packet processing in the kernel. A value of 0 leads to instant interrupts, but also generates an interrupt for every incoming packet. In this experiment the value was set to 0 microseconds to improve real-time networking performance. As a trade-off, more interrupts are generated, which lead to more interrupt handlers and more CPU load.
- *Isolating CPU cores:* By default the Linux kernel distributes processes across all CPU cores. This behavior is unwanted in real-time systems, because processes should have dedicated CPU cores that should not be shared with other processes. To isolate the cores, the `isolcpus` kernel command line argument can be used. To further leverage the now single process CPU cores, the scheduler is set to be tick-free on the specified cores with `nohz_full`. This setting in turn also offloads all Read-Copy-Update (RCU) callbacks to other cores [1].

V. RESULTS

For performance analysis, two types of measurements are evaluated: network capture from the network tap and

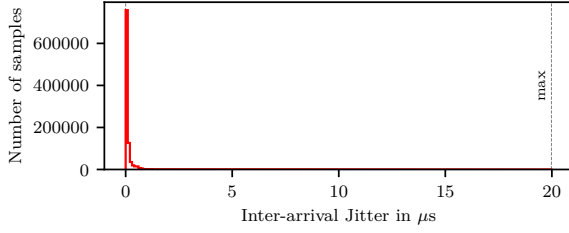


Fig. 3. Jitter on test point T1 with 1 ms cycle time on the Raspberry Pi 5.

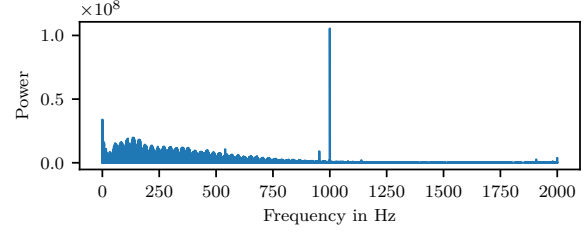


Fig. 5. Frequency spectrum of jitter on test point T1 for 250 μs cycle time on the Raspberry Pi 5.

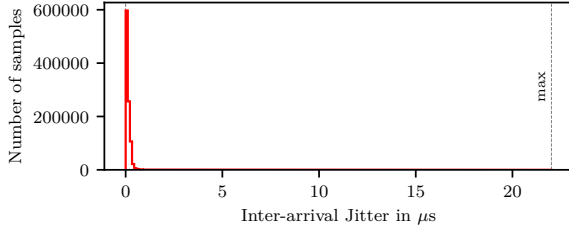


Fig. 4. Jitter on test point T1 with 250 μs cycle time on the Raspberry Pi 5.

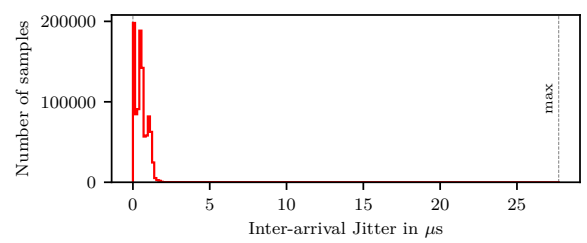


Fig. 6. Jitter of the network packets on test point T2N with 250 μs cycle time on the Raspberry Pi 5.

measurements from test points inside the application (see Figure 2).

The main difference between these two is, that the former includes latencies from the Linux kernel and Ethernet hardware. Therefore, the latter can be used to measure the real-time performance of a specific Linux kernel on the device and to indicate at which cycle times the OPC UA PubSub Protocol over TSN can be used. Even though they are different measurements and should be treated as such, the main performance indicator is shared. Jitter of the cycle time indicates irregular and unpredictable latencies in both measurement methods. Theoretically, latencies in the Linux kernel and Ethernet stack should add onto the processing and process wake-up latencies from within the application. But features in the Linux kernel implemented for better real-time networking capabilities, such as traffic control and the aforementioned `SO_TXTIME` allow for compensation and better handling of latencies. Therefore, results from the industrial PCs can have a different relation between the two measurement methods.

In the TSN example taken from the `open62541` library are multiple test points which can be used to log the exact time a packet is processed. Figure 2 provides an overview of the location of these test points.

The differences at the transmission points can be explained by differences in the operating system and processor as well as hardware architecture. Due to the different processor architecture, the operating system could not directly be duplicated from the Pi 5 to the industrial PCs. Adding onto the difference in processor architecture is that the Linux kernel version differs between the two systems, because the Linux kernel specifically adapted for the Pi 5 was used.

The jitter as defined in [11] is the variation in forwarding

delay between consecutive packets

$$J = |D_i - D_{i-1}|, \quad (1)$$

where D_i is the forwarding delay of a given packet. Given only the absolute timestamps of packet transmission in our setup, it is not possible to calculate the forwarding delay. As an alternative, the jitter is calculated based on the variation of the difference in timestamps via

$$J = |(T_{i+1} - T_i) - (T_i - T_{i-1})|, \quad (2)$$

where T_i is the absolute timestamp of a given packet. The calculation of jitter outside the application logging is not trivial due to the network tap not being synchronized with PTP. The clock drift of the recorded network tap timestamps is corrected to allow for a meaningful comparison with PTP synchronized timestamps. A linear clock drift can be corrected by computing a scaling factor

$$s = \frac{T_c}{\frac{1}{n} \sum_{i=1}^n (T_i - T_{i-1})}, \quad (3)$$

where T_c is the configured cycle time. This scaling factor is then used to adjust each non-synchronized timestamp. The remaining non-linear clock drift cannot be compensated without compromising the integrity of the results. However, due to the short recording duration, the non-linear clock drift per packet is assumed to be negligible.

The results in Figures 3 and 4 show that the jitter is very similar between the cycle time of 1 ms and 250 μs.

To further analyze unwanted cyclic jitter sources in the system the frequency spectrum of the jitter on point T1 was calculated as shown in Figure 5. From the frequency spectrum it is clear that the only cyclic jitter source are the Linux kernel timer interrupts, which were configured to 1 kHz

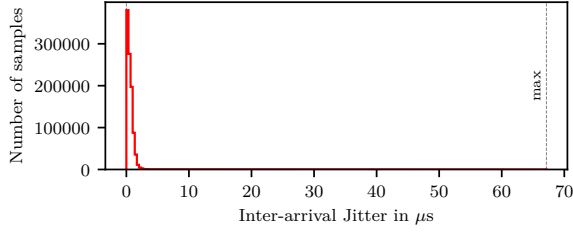


Fig. 7. Jitter on measuring point on test point T1 with 250 μ s cycle time on the industrial PC.

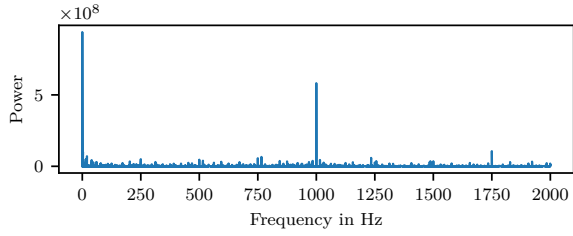


Fig. 8. Frequency spectrum of jitter on test point T1 for 250 μ s cycle time on the industrial PC.

on both systems. This behavior is expected, and the added latencies cannot be completely removed, only mitigated (see Section IV).

After measuring the performance of the real-time Linux kernel on the Pi 5 the behavior is expected and maximum jitter values do not exceed 22.02 μ s with 250 μ s cycle time. To establish a baseline the same measurements were conducted on the industrial PCs.

In Figure 7 it can be seen that the industrial PCs produce higher maximum jitter values than the Pi 5s. While testing the real-time kernel this behavior has also been observed, which indicates further that the jitter on measuring point T1 closely represents the jitter of thread wake up times. As already mentioned there is a change in processor architecture and hardware topology including but not limited to changes in cache layout between both systems [15]. These changes can have a big impact on real-time performance. Furthermore, Figure 8 shows that there is a low frequency jitter source visible, but the dominant frequency is still caused by Linux kernel timer interrupts. The low frequency jitter sources may be the System management interrupt (SMI) of Intel processors, which cause unavoidable latencies in the system. Using the `hwlatdetect`-utility (from the `rt-tests` package) the SMI interrupts were measured to cause 12 μ s of latency with a frequency of 1 Hz [19].

It has to be noted that the minimum cycle time successfully used was 250 μ s, lower cycle times lead to crashes in the application on both systems. Therefore, this does not indicate that lower cycle times are not possible on the exact hardware used.

To further compare the performance to the industrial PCs, the network capture is analyzed. As previously mentioned, the network tap adds accurate timestamps to incoming packets to prevent inaccurate timestamps from the computer

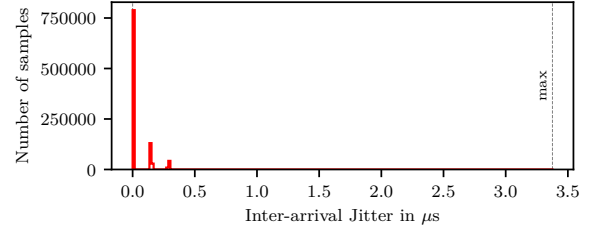


Fig. 9. Jitter of the network packets on test point T2N with 250 μ s cycle time on the industrial PCs.

TABLE I

SUMMARY OF THE MEASURED JITTER DISTRIBUTIONS ACROSS DIFFERENT TEST POINTS AND HARDWARE CONFIGURATIONS.

Device	Test Point	Rate (μ s)	Median (μ s)	25% (μ s)	75% (μ s)	99% (μ s)	Max (μ s)
RPi5	T1	250	0.075	0.037	0.166	0.610	22.016
RPi5	T1	1000	0.055	0.019	0.093	1.592	19.980
IPC	T1	250	0.466	0.213	0.810	2.158	67.140
RPi5	T2N	250	0.512	0.208	0.792	1.416	27.719
IPC	T2N	250	0.000	0.000	0.008	0.288	3.376

running the recording software. This does mean that the network tap is not synchronized to the clock of the PTP master. Therefore, the non-linear clock drift between the two clocks cannot be deducted from the packet timestamps. It is also not possible to calculate the absolute jitter of a packet from T1 to the network due to the missing common time. The jitter values in the following plots are calculated from the difference between two timestamps, which mostly removes clock drift from the results.

As Figures 6 and 9 illustrate, there is a significant difference in jitter between the two systems when comparing jitter of the network packets. The Pi 5 procures similar jitter values on the network as on measuring point T1. In contrast, the industrial PC is able to reduce the maximum jitter of network packets down to 3.38 μ s. This reduction in jitter can solely be contributed to the `SO_TXTIME` feature of the Intel i210 NIC.

VI. DISCUSSION AND FUTURE WORK

The results clearly show that industrial PCs have better real-time networking performance than the Pi 5, which was expected. The main reason is the Intel I210 NIC, which can be fine-tuned for a specific network topology and traffic to achieve better timing. Note that this finding only applies to the industrial PCs used in this experiment as there are various configurations with different NICs. The identified required features are available in Intel I210 and similar featured NICs [3].

The Pi 5, on the other hand, does not use an equally featured NIC and therefore offers reduced performance in this application. It has to be decided on a case-by-case basis whether this system is sufficient. For software based real-time networking the jitter can be compared to different testing setups including Ethercat and Powerlink from OSADL. These tests are available at [8] and [9]. At the time of writing

the maximum jitter of 5 minute intervals over 24 hours on the Powerlink setup fluctuated from 31 μ s down to 12 μ s on a 500 μ s cycle time. This setup does not utilize an Intel i210 NIC [9]. If the maximum jitter of network packets on the Pi 5 does not increase on long-term measurements, the real-time networking performance of the Pi 5 may suffice for the task of a software based Powerlink master.

To strengthen our findings, a long-term experiment using the testing setup described in this paper should be done. Furthermore, an OT device leveraging the real-time networking capabilities of the Pi 5 needs to be implemented to ensure the findings also apply outside the testing environment. Such an environment introduces non-real-time packets and therefore tests the resiliency possible on the NIC of the Pi 5.

VII. CONCLUSION

In this paper, we investigated the feasibility of using a Raspberry Pi 5 computer as a replacement of an industrial computer for real time communication. The Pi 5 computer was selected for this purpose due to its excellent low-level connectivity, compact form factor and the guaranteed long-term availability until 2036.

The findings in regard to the performance of the Pi 5 to perform real-time communication tasks using OPC UA over Time-Sensitive Networking (TSN) (*RQ1*) indicate that the Pi 5 indeed can be used as an OPC UA Node for real-time communication using the PubSub mechanism. However, we have to note that the results of this work are only applicable to small TSN networks with no additional network traffic and cross load. Furthermore, the lack of the NIC features may degrade the real-time networking performance.

The performance analysis of the Pi 5 to an industrial P (*RQ2*) showed that the latter using an Intel i210 NIC improves the jitter behavior of network packets utilizing the `SO_TXTIME` option. In detail, the maximum jitter of 3.38 μ s is significantly reduced when compared to the maximum jitter on the Pi 5, which is 27.72 μ s. One should also take into account that, industrial PCs typically offer other advanced features that enable hardware traffic control for packets with different priorities, which is important in mixed or large TSN networks.

Although the Pi 5 does not achieve a comparable result in packet jitter in the network, it offers other advantages over industrial PCs. The powerful and efficient single-board computer provides a vast range of interfaces including GPIO, Camera Serial Interface and Display Serial Interface. Furthermore, alternative variants like the Compute Module 5 make custom real-time networking hardware more accessible [14]. Moreover, we showed that the Pi 5 presents a compelling alternative, facilitating rapid prototyping of applications and significantly reducing associated costs for laboratory setups only requiring small-scale networks.

ACKNOWLEDGMENT

The financial support by the Christian Doppler Research Association, the Austrian Federal Ministry for Digital and Economic Affairs and the Federal State of Salzburg is gratefully acknowledged.

REFERENCES

- [1] "The Linux kernel documentation," accessed: 2025-03-04. [Online]. Available: docs.kernel.org
- [2] IEEE 802.1 Time-Sensitive Networking Task Group, "Time-sensitive networking (TSN) task group," 2024, accessed: 2025-03-04. [Online]. Available: <https://1.ieee802.org/tsn/>
- [3] Intel Corporation, "Intel® Ethernet controller I210 datasheet," jan 2021, revision Number: 3.7. [Online]. Available: <https://www.intel.de/content/www/de/de/products/sku/64402/intel-ethernet-controller-i210it/specifications.html>
- [4] M. Ladegourdie and J. Kua, "Performance analysis of OPC UA for industrial interoperability towards industry 4.0," *IoT*, vol. 3, no. 4, pp. 507–525, 2022. [Online]. Available: <https://www.mdpi.com/2624-831X/3/4/27>
- [5] Linux PTP Project, "The Linux PTP project," 2019, accessed: 2025-03-04. [Online]. Available: <https://linuxptp.sourceforge.net/>
- [6] A. Morato, S. Vitturi, F. Tramarin, and A. Cenedese, "Assessment of different OPC UA implementations for industrial IoT-based measurement applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–11, 2021.
- [7] Open Source Automation Development Lab *OSADL* eG, "OSADL QA farm on real-time of mainline Linux," accessed: 2025-03-04. [Online]. Available: <https://www.osadl.org/OSADL-QA-Farm-Real-time.linux-real-time.0.html>
- [8] —, "OSADL QA farm on real-time of mainline Linux: Real-time Ethernet *ethercat* worst-case round-trip time monitoring," accessed: 2025-03-04. [Online]. Available: <https://www.osadl.org/Real-time-Ethernet-Ethercat-worst-case.qa-farm-rt-ethernet-recording.0.html>
- [9] —, "OSADL QA farm on real-time of mainline Linux: Real-time Ethernet *powerlink* packet interval and jitter analysis," accessed: 2025-05-04. [Online]. Available: <https://www.osadl.org/Real-time-Ethernet-Powerlink-jitter-an.qa-farm-rt-powerlink-jitter.0.html>
- [10] J. Pfrommer, A. Ebner, S. Ravikumar, and B. Karunakaran, "Open source OPC UA PubSub over TSN for realtime industrial communication," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, Sept. 2018, pp. 1087–1090. [Online]. Available: <https://ieeexplore.ieee.org/document/8502479/>
- [11] S. Poretsky, S. Erramilli, J. Perser, and S. Khurana, "Terminology for Benchmarking Network-layer Traffic Control Mechanisms," RFC 4689, Oct. 2006. [Online]. Available: <https://www.rfc-editor.org/info/rfc4689>
- [12] Raspberry Pi Ltd, "Raspberry Pi compute module 5 datasheet: A Raspberry Pi for deeply embedded applications," 2024, accessed: 2025-03-04. [Online]. Available: <https://datasheets.raspberrypi.com/cm5/cm5-datasheet.pdf>
- [13] —, "Raspberry Pi documentation: config.txt," 2024, accessed: 2025-03-04. [Online]. Available: <https://www.raspberrypi.com/documentation/computers/config.txt.html>
- [14] —, "Raspberry Pi RP1 peripherals datasheet," 2024, accessed: 2025-03-04. [Online]. Available: <https://datasheets.raspberrypi.com/rp1/rp1-peripherals.pdf>
- [15] —, "Raspberry Pi 5 product brief," 2025, accessed: 2025-03-04. [Online]. Available: <https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf>
- [16] G. P. Reddy, Y. V. P. Kumar, Y. J. Reddy, S. R. Maddireddy, S. Prabhudesai, and C. P. Reddy, "OPC UA implementation for industrial automation - part 2: Integrating PubSub model with TSN," in *2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS)*, 2023, pp. 1–6.
- [17] M. Ulbricht, S. Senk, H. K. Nazari, H.-H. Liu, M. Reisslein, G. T. Nguyen, and F. H. P. Fitzek, "TSN-FlexTest: Flexible TSN measurement testbed," *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 1387–1402, 2024.
- [18] A. Waterland, "stress(1) - linux man page." [Online]. Available: <https://linux.die.net/man/1/stress>
- [19] K. Williams, "rt-tests - suite of real-time tests" [Online]. Available: <https://web.git.kernel.org/pub/scm/utils/rt-tests>

Multi-Modal 3D Mesh Reconstruction from Images and Text

Melvin Reka¹, Tessa Pulli¹, and Markus Vincze¹

Abstract—6D object pose estimation for unseen objects is essential in robotics but traditionally relies on trained models that require large datasets, high computational costs, and struggle to generalize. Zero-shot approaches eliminate the need for training but depend on pre-existing 3D object models, which are often impractical to obtain. To address this, we propose a language-guided few-shot 3D reconstruction method, reconstructing a 3D mesh from few input images. In the proposed pipeline, receives a set of input images and a language query. A combination of GroundingDINO and Segment Anything Model outputs segmented masks from which a sparse point cloud is reconstructed with VGGSfM. Subsequently, the mesh is reconstructed with the Gaussian Splatting method SuGaR. In a final cleaning step, artifacts are removed, resulting in the final 3D mesh of the queried object. We evaluate the method in terms of accuracy and quality of the geometry and texture. Furthermore, we study the impact of imaging conditions such as viewing angle, number of input images, and image overlap on 3D object reconstruction quality, efficiency, and computational scalability.

Index Terms—Vision Language Models, Language-guided Reconstruction, Few-shot Reconstruction

I. INTRODUCTION

6D object pose estimation for unseen objects is a critical task in robotics. Traditional methods estimate instance object poses using trained networks [1–4]. However, training models for object pose estimation is a limitation as it requires large annotated datasets, has high computational costs, and encounters difficulties in generalizing to unknown objects or environments [5]. An alternative to these methods are training-free zero-shot approaches [6, 7]. Methods such as ZS6D [6] utilize a ground-truth object model to find 2D-3D correspondences between the model and the images from

which the pose is computed using a PnP algorithm [8]. Although these methods offer a considerable advantage due to their zero-shot capabilities, they are limited by the requirement for a ground-truth 3D object model [2, 6]. Obtaining high-quality 3D models can be labor-intensive, expensive, and impractical for large-scale or real-time applications [9]. With the advent of diffusion models, recent works have proposed methods for few-shot [10, 11] or even single-shot [10, 12] 3D model reconstruction based on images. By combining SuGaR [11] with Segment Anything Model [13] (SAM) and GroundingDINO [14], we introduce a novel method to reconstruct 3D models based on images and language prompts. As input, we receive several RGB images of a scene. According to the language input, the queried object is detected with GroundingDINO [14]. Based on the bounding box, SAM [13] generates masked images depicting only the queried object. This serves as input for a sparse reconstruction with VGGSfM [15]. The sparse point cloud is then processed by SuGaR [11] to reconstruct a 3D mesh. In an automated post-processing step, artefacts are removed, resulting in a cleaned mesh. Finally, the reconstruction is evaluated on several experiments in terms of accuracy and quality of the geometric reconstruction as well as the reconstructed texture.

In summary, the paper has the following key contributions:

- We propose a novel language-guided few-shot reconstruction approach that allows 3D model reconstruction.
- Evaluation of the few-shot reconstruction method, including an analysis of the required input images considering efficiency and performance of the reconstruction.

The rest of this work is organized as follows: Section II introduces the related work of 6D object pose estimation, 3D model reconstruction, and language-guided segmentation. Section III describes the pipeline of our proposed methods. In

¹ all authors are with the Automation and Control Institute, TU Wien Vienna, Austria: e12102393@student.tuwien.ac.at; {pulli, vincze}@acin.ac.tuwien.at

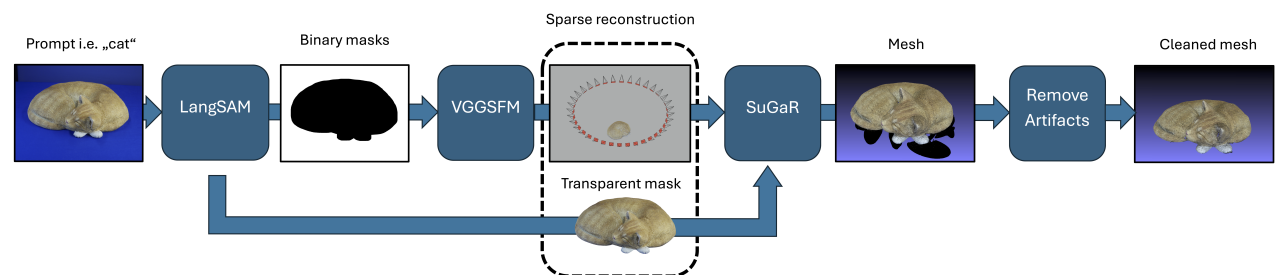


Fig. 1. Images of an object, accompanied by a descriptive text prompt, are processed through the pipeline. A sparse reconstruction using COLMAP via VGGSfM is then performed to generate and refine the mesh with SuGaR.

Section IV, the experimental setup is presented while Section V discusses the evaluation. Section VI concludes the paper with a summary and outlook.

II. RELATED WORK

In this section, we discuss the related work by revisiting 6D object pose estimation, few-shot reconstruction methods, and language-guided object segmentation.

A. 6D Object Pose Estimation

6D object pose estimation of unseen objects is a core task in robotics. Classical methods estimate the pose of objects using trained networks for object instances [1–3]. These methods require large annotated datasets, which are costly to acquire while requiring high computational power during training [5]. Furthermore, these models struggle to generalize to unseen objects [1, 3, 4], limiting their applicability in real-world scenarios. These challenges can be overcome with zero-shot object pose estimation methods [6, 7]. These approaches eliminate the need for extensive training by leveraging prior knowledge by assuming that object models exist. Given the reference model, 2D-3D correspondences between the object’s model and a set of input images can be established [8]. With this information available, the object pose is computed by a PnP/RANSAC algorithm [16]. Although zero-shot methods offer a significant advantage by overcoming the training phase, they still require high-quality ground-truth 3D models. This dependency presents challenges in practical applications, as the acquisition of accurate 3D models is labor-intensive and requires expensive equipment [17].

B. Few-Shot 3D Reconstruction Methods

Recent advances in generative models, particularly diffusion models, have opened new possibilities to acquire 3D meshes. Works such as Wonder3D [12], Gaussian Splatting [18], Dreamfusion [19], and Sugar [11] have demonstrated the potential of generating 3D object representations from a limited number of 2D images. Typically, these methods use 2D diffusion models to generate novel views from different camera poses. From these novel views, a 3D model is reconstructed with a stochastic 3D reconstruction framework [19] or can be parameterized as a voxel radiance field from which the mesh is extracted with a marching cubes procedure [20]. Because existing reconstruction methods typically assume clean, isolated object inputs, several methods introduced a pre-processing step to delete the background of the input images to avoid noise while reconstructing the images [21].

C. Language Guided Object Segmentation

The integration of vision-language models (VLMs) such as CLIP [22] has significantly expanded the capabilities of computer vision systems, allowing them to understand and process images based on textual descriptions. CLIP [22] has been incorporated into a wide range of applications, allowing methods including scene understanding [23], object

recognition [24], and generative modeling [19]. An area where this integration has proven particularly beneficial is object segmentation [13], where language-guided approaches allow for more intuitive and adaptable object selection. Recent advances in segmentation models provide a method to segment objects in an image using minimal user input. SAM [13] allows for object selection through different means, such as points, bounding boxes, or language-based prompts, making it an effective tool for isolating objects in complex scenes. By leveraging SAM’s capabilities, objects can be accurately segmented and masked before being passed into a 3D reconstruction pipeline.

III. METHOD

As shown in Figure 1, images of the object, along with a descriptive text prompt, are processed through the pipeline. LangSAM [25] combines GroundingDINO [14] and SAM [13] for text-driven segmentation. GroundingDINO processes the text prompt to generate bounding boxes around relevant objects, which SAM then uses to create binary masks with a 50-pixel padding. This padding proved helpful when handling semi-transparent objects and reduces artifacts introduced during mesh reconstruction. Focusing on essential scan areas enhances reconstruction quality while maintaining computational efficiency.

The generated masks, along with the original images, are then used for 3D reconstruction with VGGSFm [15]. As a fully differentiable structure-from-motion pipeline, VGGSFm estimates camera parameters, determines camera positions, and reconstructs a sparse point cloud by tracking corresponding 2D points across multiple views. This end-to-end differentiable approach enhances the accuracy and robustness of the reconstruction process by eliminating the need to chain pairwise matches and enabling simultaneous recovery these[15].

Using the resulting COLMAP[26] dataset and the extracted RGB masks, a textured mesh is generated with SuGaR[11], which employs Gaussian Splatting to efficiently optimize and extract a high-resolution 3D surface. However, since SuGaR can introduce artifacts during mesh generation, an automated script utilizing PyMeshLab[27] is applied to remove these artifacts.

IV. EXPERIMENTS

In this section, the experimental setup is discussed. During our evaluation, we assess the accuracy and quality of the geometric construction as well as the reconstructed texture.

A. Implementation details:

All experiments are conducted on a system equipped with an AMD Ryzen 9 5950X CPU, 128GB RAM, and an NVIDIA RTX 3090 GPU with 24GB VRAM. The implementation is containerized using Docker to ensure reproducibility across different hardware environments.

B. Evaluation Metrics

To assess the accuracy and quality of both geometric reconstruction and texture extraction, we distinguish between geometric metrics and texture similarity.

Geometric Metrics

We evaluate the reconstructed 3D geometry using Chamfer Distance [28] and Intersection over Union [29]. CD quantifies the average squared distance between nearest neighbors in the predicted and ground-truth meshes. It is defined as:

$$CD(P, Q) = \frac{1}{|P|} \sum_{p \in P} \min_{q \in Q} \|p - q\|^2 + \frac{1}{|Q|} \sum_{q \in Q} \min_{p \in P} \|q - p\|^2 \quad (1)$$

where P and Q are the sets of points in the predicted and ground-truth meshes. Low CD values indicate a more precise geometric alignment.

IoU measures the volumetric similarity between the reconstructed and ground-truth models:

$$IoU = \frac{|V_P \cap V_Q|}{|V_P \cup V_Q|} \quad (2)$$

where V_P and V_Q represent the volumetric reconstructions of the predicted and ground-truth meshes. IoU measures the proportion of the shared volume, with higher values indicating better alignment.

Texture Similarity

To assess texture extraction accuracy, we use the three key metrics employed in SuGaR [11].

The Peak Signal-to-Noise Ratio is defined as:

$$PSNR = 10 \log_{10} \left(\frac{MAX^2}{MSE} \right) \quad (3)$$

where MAX is the maximum possible pixel value, and MSE is the mean squared error between the predicted and ground-truth textures. Higher values indicate better pixel-wise preservation, but do not reflect human perception.

The Structural Similarity Index [30] takes into account luminance, contrast, and texture integrity, reflecting human perception. SSIM is computed as:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4)$$

where μ_x and μ_y are the mean intensities, σ_x^2 and σ_y^2 are the variances, and σ_{xy} is the covariance between the predicted and ground-truth images.

The Learned Perceptual Image Patch Similarity [31] is given by:

$$LPIPS(x, y) = \sum_l w_l \|F_l(x) - F_l(y)\|_2^2 \quad (5)$$

where F_l represents the feature maps at layer l of a pretrained network, and w_l are learned weights. LPIPS captures high-level perceptual differences, making it effective for identifying distortions and artifacts beyond pixel-wise comparisons.

C. Experimental Setup

Our experiments evaluate the impact of various imaging conditions on the quality of 3D object reconstruction. We investigate how the viewing angle θ influences feature extraction and reconstruction accuracy, as different angles affect feature visibility. We also examine the effect of the number of input images on reconstruction convergence, assessing how multi-view stereo improves model quality. The overlap between input images, determined by rotation step sizes ϕ , is another factor influencing reconstruction accuracy. Mesh quality is assessed by comparing the texture extraction accuracy and alignment with ground truth. Finally, a runtime analysis measures the scalability of computational costs with the number of input images and processing steps, balancing efficiency and accuracy in real-time applications. These experiments aim to understand the influence of these parameters on reconstruction quality, efficiency, and robustness.

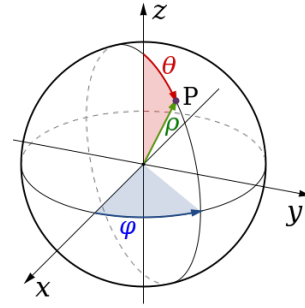


Fig. 2. Spherical coordinates, where we refer to the polar angle θ as viewing angle, and to the azimuthal angle ϕ as

Furthermore, we assess mesh quality, focusing on the accuracy of texture extraction and the overall fidelity of the reconstructed surfaces. Finally, we conduct a runtime analysis to measure how computational cost scales with the number of input images and processing steps, balancing efficiency and accuracy in multi-view stereo reconstruction.

D. Dataset

To evaluate the capabilities of the proposed work, we utilize the MVS dataset [32] consisting of two sets of multi-view images, their camera parameters, and the ground-truth mesh models. It includes multiple views of each scene, captured from different angles to provide a diverse set of perspectives. The viewing angles are characterized by two key parameters: theta (θ), the polar angle, which defines the elevation or vertical angle from which the scene is viewed. 2.

$$\theta_{\text{cat}} \in \{30^\circ, 45^\circ, 75^\circ\} \wedge \theta_{\text{dog}} \in \{45^\circ, 90^\circ\}$$

Phi ϕ , the azimuthal angle, which represents the horizontal rotation around the scene. By capturing images across varying θ and ϕ angles, the dataset offers a comprehensive range of viewpoints. Figure 3 shows examples of the dataset and its target objects which are figurines of a cat and a dog. The images are taken with the camera by changing the height of

the camera with three different viewing angles. θ is the polar angle between the z-axis and the camera position, which we refer to as θ , as it can be seen in Figure 3.

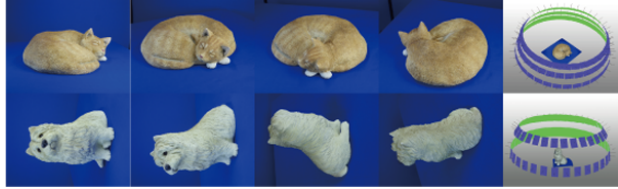


Fig. 3. THU Multi-view stereo datasets [32] of a cat and a dog. The left side shows the input images captured from various viewpoints, while the right side displays the corresponding camera viewpoints and the target objects.

V. EVALUATION

This evaluation assesses the performance of the proposed method based on input images, focusing on the trade-off between accuracy and efficiency. Specifically, we analyze which input configurations yield the most precise reconstructions while maintaining computational efficiency.

Number of Input Images

In a first experiment, we show the impact of the number of input images on the geometric reconstruction quality. We used three different sets of images according to the three camera viewing angle θ (30° , 45° , 75°). While several combinations of rotation angles between images are possible, we chose the best result for each number of images, neglecting factors such as overlap between images as these are investigated in the following experiments. Figure 7 shows that for each of these data sets, the model converges for both CD and IoU approximately from 15 images onward while the best performance is achieved with the maximum number of input images of 36. However, some outliers deviate significantly from this trend, which can be attributed to the effects of overlap and coverage.

Runtime

In Figure 4, the model exhibits a linear runtime increase with the number of images up to 18. At 36 images, a drop-off occurs as VGGSfM is downscaled by half to fit within the available VRAM, resulting in a lower-quality reconstruction but a more accurate overall outcome. The runtime is divided into three parts: segmentation (negligible), sparse reconstruction (scales linearly with image count), and mesh extraction (consistently 4–5 minutes).

Viewing Angle Theta θ

In the first experiment, we investigate how different camera angles θ affect reconstruction quality. Therefore, the relationship between the number of images and different camera angles is used on the example of the cat and dog figurines. Figure 5 presents the best reconstruction results in terms of IoU and CD for each tested viewing angle θ . While multiple configurations are possible, we report only the optimal results for each angle.

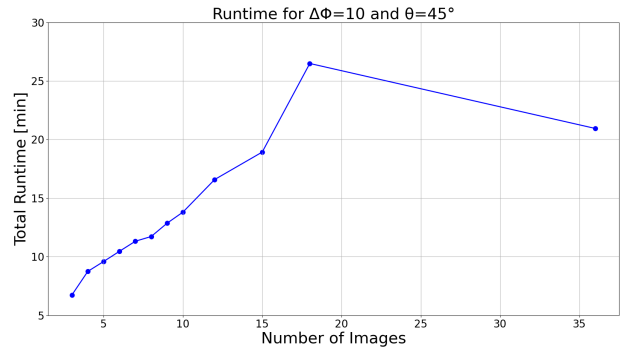


Fig. 4. Runtime vs. input images for $\theta = 45^\circ$, $\Delta\phi = 10^\circ$

Our findings indicate that the ideal angle of incidence for both objects is 45° . At this angle, VGGSfM achieves the most effective feature extraction, as it captures both the top and front of the object within the same image. This results in a higher number of extracted points without increasing the total number of input images, outperforming alternative angles such as 30° and 75° for the cat and 90° for the dog.

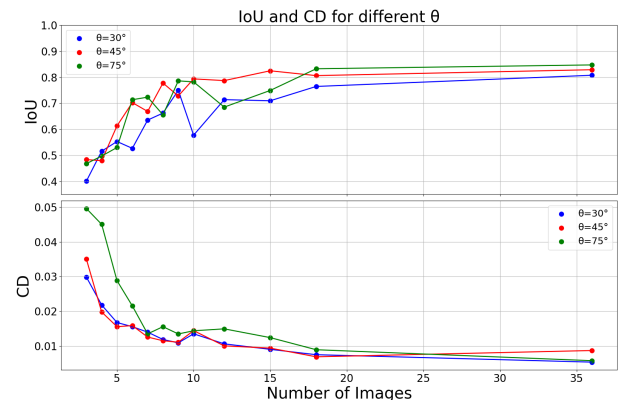


Fig. 5. IoU and Chamfer Distance for cat figurine at different θ angles.

Overlap of Input Images

To reduce runtime and the computational effort required for mesh reconstruction, the following experiments explore how the input image set can be optimized. A key factor in this process is the overlap of input images. With a small rotation step ($\Delta\phi$) and few images, the visible areas are well-reconstructed, but limited coverage lowers benchmark scores despite reasonable results for small datasets. On the other hand, using few images and a large rotation step ($\Delta\phi$), leads to insufficient feature overlap and therefore an incomplete mesh. Increasing the number of images with a small $\Delta\phi$ enhances reconstruction quality, but the results are still limited to the visible areas. A balanced approach, such as 12 images with $\Delta\phi = 30^\circ$, achieves comparable quality to using three times as many images, improving both accuracy and coverage.

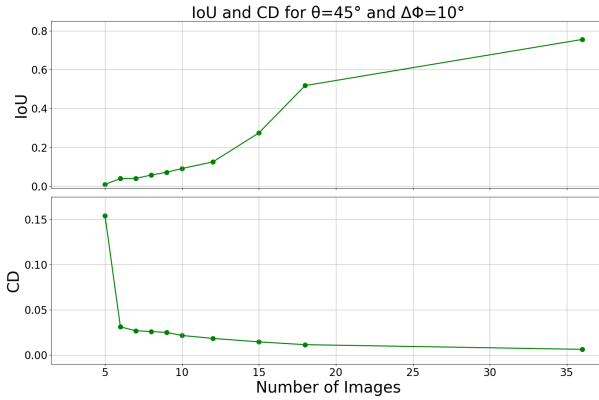


Fig. 6. IoU and Chamfer Distance for dog figurine at fixed θ and $\Delta\phi$

Nr. of images	$\Delta\phi$ [DEG]	CD $\downarrow$	IoU [%] $\uparrow$	Runtime [min] $\downarrow$
4	10°	0.0376	47.91	8.75
4	90°	0.0277	43.52	9.70
12	10°	0.0162	63.32	16.59
12	30°	0.0100	78.72	16.06
36	10°	0.0087	82.91	21.95

TABLE I

OVERLAP EXTREMES VS. IMAGE COUNT FOR CAT AT $\theta = 45^\circ$

Mesh Quality

As shown in Table II, images taken with smaller θ angles, such as top-down views, result in better texture similarities. Top-down perspectives provide a clearer view of surface details, making it easier to capture fine-grained textures. In contrast, frontal views with higher θ values lead to lower quality meshes due to occlusions and the lack of sufficient top-view coverage, making it difficult to accurately reconstruct the surface. Optimizing the pipeline and capturing images from the optimal angle leads to significantly improved SSIM and LPIPS scores, which, in human perception, translates to textures that closely resemble the ground truth.

Figurine	θ [DEG]	PSNR [dB] $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
Cat	30°	33.95	0.967	0.0373
Cat	45°	31.75	0.947	0.0546
Cat	75°	13.52	0.644	0.2516
Dog	45°	34.35	0.972	0.0353
Dog	90°	22.66	0.843	0.1195

TABLE II

TEXTURE QUALITY AT DIFFERENT θ WITH 36 IMAGES. VALUES PRINTED IN **BOLD** INDICATE THE BEST RESULT FOR EACH FIGURINE

VI. SUMMARY AND OUTLOOK

This paper analyzes how the angle of incidence and the angular distance between input images affect photogrammetric reconstruction quality. We show that an optimal balance between image count and angular separation significantly enhances mesh quality, while excessive gaps hinder feature matching. A key challenge identified is scale estimation,

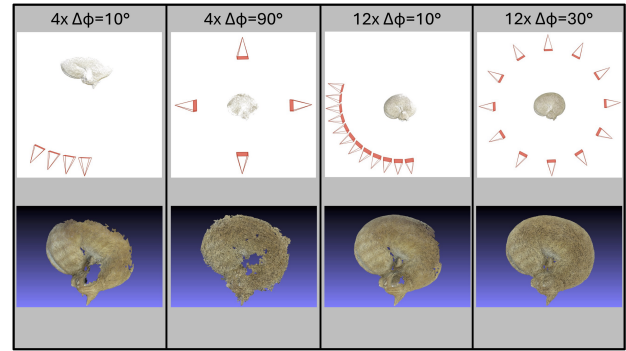


Fig. 7. Overlap of input images comparison for $\theta = 45^\circ$

which could be improved by integrating a reference object for automatic scaling. Additionally, sparse reconstruction is a major computational bottleneck, suggesting the need for more efficient alternatives. Future work should focus on optimizing reconstruction pipelines to improve runtime and scale consistency, making the approach more suitable for large-scale or real-time applications.

ACKNOWLEDGMENT

We gratefully acknowledge the support of the EU-program EC Horizon 2020 for Research and Innovation under project No. I 6114, project iChores.

REFERENCES

- [1] G. Wang, F. Manhardt, F. Tombari, and X. Ji, “Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 611–16 621.
- [2] Y. Su *et al.*, “Zebropose: Coarse to fine surface encoding for 6dof object pose estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6738–6748.
- [3] K. Park, T. Patten, and M. Vincze, “Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7668–7677.
- [4] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, “Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes,” *arXiv preprint arXiv:1711.00199*, 2017.
- [5] D. Bauer, P. Hönig, J.-B. Weibel, J. García-Rodríguez, M. Vincze, *et al.*, “Challenges for monocular 6d object pose estimation in robotics,” *IEEE Transactions on Robotics*, 2024.
- [6] P. Ausserlechner, D. Habegger, S. Thalhammer, J.-B. Weibel, and M. Vincze, “Zs6d: Zero-shot 6d object pose estimation using vision transformers,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 463–469.

- [7] S. Thalhammer, J.-B. Weibel, M. Vincze, and J. Garcia-Rodriguez, “Self-supervised vision transformers for 3d pose estimation of novel objects,” *arXiv preprint arXiv:2306.00129*, 2023.
- [8] P. Hönig, S. Thalhammer, and M. Vincze, *Improving 2d-3d dense correspondences with diffusion models for 6d object pose estimation*, 2024. arXiv: 2402.06436 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2402.06436>.
- [9] Q. Shen, X. Yang, and X. Wang, “Anything-3d: Towards single-view anything reconstruction in the wild,” *arXiv preprint arXiv:2304.10261*, 2023.
- [10] G. Qian *et al.*, “Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors,” *arXiv preprint arXiv:2306.17843*, 2023.
- [11] A. Guédon and V. Lepetit, *Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering*, 2023. arXiv: 2311.12775 [cs.GR]. [Online]. Available: <https://arxiv.org/abs/2311.12775>.
- [12] X. Long *et al.*, “Wonder3d: Single image to 3d using cross-domain diffusion,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 9970–9980.
- [13] A. Kirillov *et al.*, “Segment anything,” *arXiv:2304.02643*, 2023.
- [14] S. Liu *et al.*, “Grounding dino: Marrying dino with grounded pre-training for open-set object detection,” *arXiv preprint arXiv:2303.05499*, 2023.
- [15] J. Wang, N. Karaev, C. Rupprecht, and D. Novotny, *Vggsfm: Visual geometry grounded deep structure from motion*, 2023. arXiv: 2312.04563 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2312.04563>.
- [16] F. Li *et al.*, “Nerf-pose: A first-reconstruct-then-regress approach for weakly-supervised 6d object pose estimation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2123–2133.
- [17] J. Choi *et al.*, “Tmo: Textured mesh acquisition of objects with a mobile device by using differentiable rendering,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 674–16 684.
- [18] P. Dai, J. Xu, W. Xie, X. Liu, H. Wang, and W. Xu, “High-quality surface reconstruction using gaussian surfels,” in *ACM SIGGRAPH 2024 Conference Papers*, Association for Computing Machinery, 2024.
- [19] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, “Dreamfusion: Text-to-3d using 2d diffusion,” *arXiv preprint arXiv:2209.14988*, 2022.
- [20] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, “Zero-1-to-3: Zero-shot one image to 3d object,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 9298–9309.
- [21] J. Xu, W. Cheng, Y. Gao, X. Wang, S. Gao, and Y. Shan, “Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models,” *arXiv preprint arXiv:2404.07191*, 2024.
- [22] A. Radford *et al.*, *Learning transferable visual models from natural language supervision*, 2021. arXiv: 2103.00020 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2103.00020>.
- [23] R. Chen *et al.*, “Clip2scene: Towards label-efficient 3d scene understanding by clip,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7020–7030.
- [24] Y. Zhang *et al.*, “Recognize anything: A strong image tagging model,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 1724–1732.
- [25] L. Medeiros, *Langsegmentanything: A repository for segmentation using language models*, <https://github.com/luca-medeiros/lang-segment-anything/tree/0a12766ced0503f16ee50e6fa99a9a4f5fbd4ea5>, Accessed: 2025-02-08, 2023.
- [26] J. L. Schönberger and J.-M. Frahm, “Structure-from-motion revisited,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [27] A. Muntoni and P. Cignoni, *PyMeshLab*, Jan. 2021. DOI: 10.5281/zenodo.4438750.
- [28] F. Williams, *Point cloud utils*, <https://www.github.com/fwilliams/point-cloud-utils>, 2022.
- [29] P.-S. Wang, Y. Liu, and X. Tong, “Dual octree graph networks for learning adaptive volumetric shape representations,” *ACM Transactions on Graphics (SIGGRAPH)*, vol. 41, no. 4, 2022.
- [30] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004. DOI: 10.1109/TIP.2003.819861.
- [31] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *CVPR*, 2018.
- [32] S. SAKAI, K. ITO, T. AOKI, T. WATANABE, and H. UNTEN, “Phase-based window matching with geometric correction for multi-view stereo,” *IEICE Transactions on Information and Systems*, vol. E98.D, no. 10, pp. 1818–1828, 2015. DOI: 10.1587/transinf.2014EDP7409.

Sensorized Adaptive Grasping: ROS2 Based Integration of UR3e and Schunk SVH with Force Sensors

Youssef Aboud¹, Andrew Johnson¹, Gidugu Lakshmi Srinivas¹ and Mathias Brandstötter¹

Abstract—Robotic grasping is a critical challenge in automation, requiring precise control to handle objects of varying shapes and fragility. While industrial robotic arms offer reliable motion control, their ability to adapt gripping force dynamically is often limited. This work addresses the need for force-sensitive grasping by integrating the Universal Robot UR3e with the Schunk SVH robotic hand in a ROS2-based framework. The key innovation lies in a real-time force-controlled grasping system, where force sensors embedded in the fingers provide continuous feedback to regulate applied force. The system operates within a closed-loop control structure, ensuring that no additional force is applied to the object once the required force is reached. This prevents deformation or slippage, enabling safer and more adaptive handling. The framework was validated through simulated grasping tasks involving objects such as a ball, a square block, and an apple. Each task tested the system's ability to adjust its grip in response to sensor feedback. The integration process included configuring ROS2-based communication, developing motion planning using MoveIt2, and visualizing robot trajectories in RViz. The UR3e trajectories were tested in Gazebo to simulate grasping interactions before real-world deployment, ensuring reliable execution. Future work will focus on enhancing object detection by integrating computer vision modules into the study. A camera system automatically identifies and localizes objects, reducing reliance on predefined grasping positions. This addition will enable autonomous grasp selection, making robotic manipulation more adaptive in unstructured environments.

Index Terms—ROS2 robotic manipulation, Schunk SVH, Universal Robot UR3e, Adaptive force feedback, closed-loop control system, Sensorised grasping

I. INTRODUCTION

Robotic grasping remains a fundamental challenge in industrial automation, service robotics, and human-robot interaction. While robotic arms have achieved high precision in motion execution, their ability to handle objects with varying shapes and fragility remains limited. Traditional position-controlled grasping methods lack adaptability, often leading to excessive force application or unstable grip performance [1]. Force-controlled grasping, where tactile sensors provide real-time feedback, enables robots to interact safely and effectively with objects [3]. Several industries, including manufacturing [15], healthcare [17], and logistics [11], demand robotic systems that can grasp objects without predefined parameters. Integrating force sensors in robotic hands enhances adaptability, ensuring secure and precise manipulation without damaging delicate objects [12]. Existing research has explored sensor-driven grasping using various

robotic hands [14], but there is still a gap in seamlessly integrating force feedback within ROS2-based control architectures. This work addresses this limitation by developing a real-time force-controlled grasping system for the Universal Robots UR3e and Schunk SVH hand, fully integrated within the ROS2.

Several studies have focused on enhancing robotic grasping through sensor integration and adaptive control. Researchers have explored tactile sensor-based grasping, demonstrating improved grip stability using force sensors on robotic fingers [16], [6]. The Schunk SVH hand has been studied for its human-like dexterity [4], but its potential for adaptive grasping in a ROS2-based environment remains under-explored. While such five-fingered, highly sensitive grippers offer impressive manipulation capabilities, they are not widely adopted in industrial applications due to their complexity and cost. As a result, their use is still largely confined to research environments, where more advanced dexterity and nuanced control are of interest. The Universal Robots UR3e has been widely used in ROS-based applications [13], with works focusing on motion planning using MoveIt2 [7] and real-time execution with RTDE [9]. However, previous studies [18] often rely on position control rather than force feedback, limiting adaptability. Simulated environments like Gazebo have proven effective for testing robotic grasping strategies [8]. Studies integrating ROS and Gazebo have focused on collision-free grasping and trajectory optimization [10], yet a complete pipeline combining force sensing, ROS2, and dynamic control has not been fully realized. Additionally, recent works on sensor fusion for robotic grasping highlight the importance of integrating multiple sensing modalities, including force sensors and vision systems, to achieve optimal grasping strategies [2]. Additionally, learning-based approaches leveraging reinforcement learning have demonstrated improved adaptability by enabling robots to refine their grasping techniques dynamically in unstructured environments [5].

The primary objectives of this paper are to design and implement a ROS2-based control framework that seamlessly integrates the UR3e robotic arm and the SVH five-fingered robotic hand, to develop advanced algorithms for real-time, sensor-driven grip adjustment, and to evaluate the system's performance in dynamic manipulation tasks rigorously. By leveraging the high precision and repeatability of the UR3e, the human-like dexterity and grasping capabilities of the SVH hand, and the adaptability enabled through continuous sensor feedback, this work seeks to push the boundaries of robotic manipulation in complex, unstructured environments.

¹All authors are with ADMiRE Research Center, Carinthia University of Applied Science, Villach, Austria {youssef.aboud, edu.johand001}@edu.fh-kaernten.ac.at and {l.gidugu, m.brandstoetter}@fh-kaernten.at

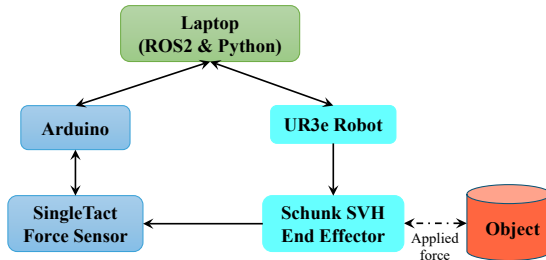


Fig. 1. The hardware and software architecture for ROS2-based adaptive grasping with UR3e, Schunk SVH, Arduino, and force sensors

Through this integration, the paper aims to contribute to more autonomous, flexible, and robust robotic systems capable of performing nuanced tasks in real-world applications.

II. METHODOLOGY

A. Hardware and Software Architecture

The system integrates hardware and software components to enable adaptive force-controlled grasping using the UR3e robotic arm and Schunk SVH end effector. The primary objective is to create a closed-loop control mechanism that dynamically adjusts grip force based on real-time sensor feedback, ensuring reliable and effective manipulation of objects. This architecture is built around the ROS2 middleware, Python for data processing and control, an Arduino microcontroller for sensor communication, and SingleTact force sensors for force measurement, as shown in Fig. 1.

ROS2 is the core communication framework facilitating data exchange between different system components. It provides real-time messaging and distributed computing capabilities that allow nodes running on the control laptop, UR3e robotic arm, and Schunk SVH end effector to interact efficiently. The ROS2 middleware is responsible for processing position commands, reading joint states, and handling interrupts triggered when predefined force thresholds are exceeded. This ensures that grasping actions are dynamically adjusted based on real-time feedback, making it suitable for handling fragile or deformable objects without damage. Python is crucial for system monitoring, sensor data processing, and implementation of control logic. A Python script running on the control laptop continuously listens to incoming data from the force sensors transmitted via the Arduino. The script evaluates whether the grip force exceeds a predefined threshold and, if necessary, publishes an interrupt message to ROS2. This interrupt triggers the system to read the current position of the Schunk SVH fingers and adjust the grip trajectory accordingly. Once the target force level is reached, the Schunk hand is activated to hold the object securely. By leveraging Python's real-time processing capabilities, the system ensures that gripping forces remain within optimal limits, preventing excessive force application and object slippage.

The Arduino microcontroller intermediates the force sensors and the ROS2-based control system. The force sensors are physically attached to the fingers of the Schunk SVH

end effector, and their readings are collected and processed by the Arduino. These readings are then forwarded via a serial connection to the control laptop, where Python scripts interpret and analyze the data. The Arduino operates continuously, ensuring that sensor values are relayed in real-time without delay. This low-level sensor acquisition and communication process is essential for the higher-level control algorithms that govern adaptive grasping. A key component in the system's force-sensing capability is the SingleTact force sensor, which provides precise and high-resolution force measurements. The sensor is integrated with a control board that converts analog signals into digital values, ensuring stability and accuracy under different grasping conditions. These digital readings are transmitted to the Arduino through an I2C (Inter-Integrated Circuit) interface, allowing for efficient data acquisition. The control board also handles sensor calibration, ensuring the force readings remain reliable and consistent. Integrating these sensors into the system enables continuous force feedback, allowing for fine-tuned adjustments to the grip strength of the robotic hand.

B. UR3e Integration and Control in ROS2

The UR3e was integrated within the ROS2-based system to provide precise motion control and seamless coordination with the Schunk SVH end-effector. The setup involved configuring network communication, installing the ROS2 driver, validating hardware functionality, and developing custom scripts for motion planning and control. To ensure compatibility with ROS2, the system was set up with Ubuntu and ROS2 Humble. Network communication was established by assigning static IP addresses to both the control PC and the UR3e teach pendant, enabling direct Ethernet-based communication.

The Universal Robots ROS2 driver was obtained from the official GitHub repository¹ and compiled within the ROS2 workspace. A communication program was created on the UR3e teach pendant to enable external command execution. Once the driver was launched on the PC, remote operation of the UR3e was activated, allowing full control via ROS2. To verify hardware functionality, the UR3e was tested using ROS2 service calls and topic-based communication, ensuring proper joint state updates and motion command execution. For simulation, Gazebo was installed, along with the official Universal Robots ROS2-Gazebo integration package². This setup provided a virtual testing environment, allowing trajectory validation before real-world execution. The combination of simulation and physical validation enabled safe experimentation with different control strategies, ensuring reliable robot performance.

C. Motion Control and Trajectory Execution

The motion control of the UR3e robotic arm was implemented using MoveIt2 in ROS2, allowing for joint and

¹https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver

²https://github.com/UniversalRobots/Universal_Robots_ROS2_Gazebo_Simulation

Cartesian-based movement execution. Python scripts were developed to control the UR3e's motion using inverse kinematics and trajectory planning. MoveIt2 provided advanced motion planning features, including collision avoidance and optimized path execution, ensuring smooth and adaptive manipulation. ROS2 nodes were programmed to adjust the robot's movement based on sensor feedback dynamically, enabling precise grasping and interaction with objects. The implementation leveraged action servers to execute motion commands efficiently, ensuring real-time adaptability in robotic grasping tasks.

1) *Joint Control Using MoveIt2*: Joint-based control regulated individual joint angles, allowing precise control over the UR3e's motion. Instead of defining Cartesian coordinates, this method focused on achieving a specific joint configuration. The MoveIt2 framework computed time-parameterized trajectories that guide each joint to its target position while respecting joint-level velocity and acceleration constraints. Inverse kinematics was performed using the default KDL solver in MoveIt2, which provides joint configurations that are locally optimal in terms of minimal displacement from the current joint state. This ensures smooth, continuous motion that is well-suited for real-time grasping tasks. The ROS2 action server sent motion commands, ensuring reliable execution. This approach is particularly useful for structured tasks such as pick-and-place operations, where predefined joint configurations are essential. The combined joint and cartesian control implementation process is provided as a pseudo-code, as shown in Algorithm 1.

Algorithm 1 Combined Pseudo Code for UR3e Joint & Cartesian Control Using MoveIt2 in ROS2

- 1: **Initialize** the ROS2 system and create a node for MoveIt-based control.
- 2: **Establish** an Action Client for MoveIt2's MoveGroup action server.
- 3: **Wait** for the `/move_action` server to become available.
- 4: **Define a function for Joint Control (MoveJ):**
 - Set target joint positions.
 - Specify velocity and acceleration scaling factors.
 - Create MoveIt2 joint constraints and assign them to UR3e joints.
 - Send the MoveJ command via MoveIt2 Action Client.
 - Execute MoveJ with chosen velocity and acceleration.
- 5: **Define a function for Cartesian Control (MoveL):**
 - Set a target position in Cartesian space (x, y, z) .
 - Apply end-effector constraints for straight-line motion.
 - Use a bounding box or waypoints for accurate positioning.
 - Send the MoveL command to the action server.
 - Execute MoveL with controlled speed and accuracy.
- 6: **Keep** the ROS2 node running for continuous operation.
- 7: **Shutdown** the node upon completion.

2) *Cartesian Control Using MoveIt2*: Cartesian control was implemented to enable the UR3e's end-effector to reach specific positions in Cartesian space (X, Y, Z) rather than following predefined joint angles. This approach relied on the inverse kinematics to calculate the required joint positions dynamically. MoveIt2 generated collision-free trajectories, ensuring smooth and precise motion. Cartesian constraints, such as bounding boxes and position constraints on the wrist, were applied to maintain accuracy. This method is particularly beneficial for applications with critical end-effector positioning, such as assembly tasks and object manipulation.

Fig. 2 demonstrates motion control of a UR3e robot using ROS2 and MoveIt2, showcasing both joint-based and Cartesian-based control methods. In the left section (a), MoveJ is used for joint-space motion planning, where the robot moves through a series of predefined joint angles for precise articulation. In the right section (b), MoveL is applied for Cartesian-space motion, ensuring the end-effector follows a straight-line trajectory in 3D space. The terminal outputs confirm the successful execution of both control commands, with MoveJ handling overall joint positioning and MoveL ensuring smooth linear movements. These approaches allow flexible motion planning, depending on the task requirements, such as obstacle avoidance or precise end-effector placement.

D. Schunk SVH end-effector

Integrating the Schunk SVH five-fingered hand within the ROS2-based system followed a structured approach to ensure reliable operation and precise control. The integration process consisted of system setup, library installation, hardware validation, and the development of custom scripts for joint control. The software environment was configured by installing Ubuntu with ROS2 Humble, ensuring compatibility with the Schunk SVH ROS2 driver. Dependencies were verified and installed to facilitate seamless communication between ROS2 and the end effector. The Schunk SVH ROS2 driver was obtained from the official GitHub repository³. Due to limited documentation of the SVH ROS2 driver, script

³https://github.com/SCHUNK-SE-Co-KG/schunk_svh_ros_driver

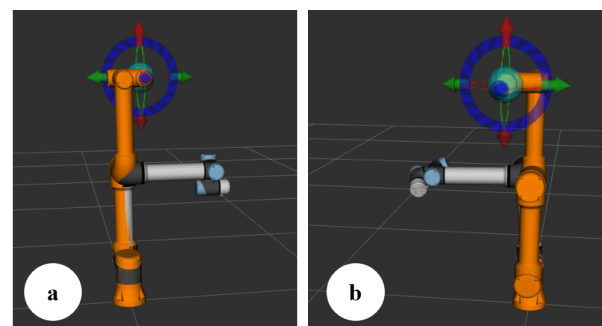


Fig. 2. Comparison of joint-space (MoveJ) and Cartesian-space (MoveL) motion control for a UR3e robot using ROS2 and MoveIt2

development required debugging and adaptation of provided examples. These custom scripts formed the foundation for reliable communication between ROS2 and the SVH hand. The hand was connected via USB, and communication with the ROS2 framework was achieved without issues. The provided ROS2 script examples were adapted to control each joint of the Schunk SVH hand. These scripts were used to execute test sequences, verifying the accuracy and repeatability of joint movements. The inverse kinematics was applied to the end effector to accurately position the end effector (EE) for the pick-and-place tasks. The transformation matrix represents the position and orientation of the end effector relative to the target object, which was used as input to the inverse kinematics algorithm. The solution provides the necessary position and orientation of the end effector to ensure that the robot's arm places it precisely at the required location. This calculation considers the robot's physical constraints and ensures that the end effector reaches the target with the correct pose without recomputing the inverse kinematics of the entire arm.

E. Force sensors

1) *Sensor Selection and Setup*: A SingleTact capacitive force sensor was selected due to its high sensitivity and compact form factor, making it suitable for integration into the Schunk SVH robotic hand. The sensor is small enough to be affixed to the inner gripping surfaces of the fingers, enabling direct measurement of contact forces during object manipulation. The sensor was connected to its control board, which provided signal conditioning and a digital output accessible via an I2C interface. An Arduino Uno was used to interface between the control board and ROS2. To ensure reliable force measurements, the sensor was placed on the distal phalanx of the robotic thumb, as shown in Fig. 3. This location was chosen to ensure contact with the sensor while grasping objects of varying shapes.

2) *Serial Communication and Data Parsing*: The force sensor data was acquired using an Arduino Uno microcontroller, which then transmits the sensor readings at a 57600 baud rate over a serial connection. The flowchart provides the communication process, as shown in Fig. 4. The Arduino firmware transmits raw integer values, which are then parsed and processed in a Python-based ROS2 node (stopper.py) running on an Ubuntu-based control system. The node reads

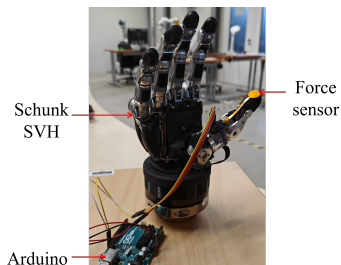


Fig. 3. The placement of SingleTact force sensor on the distal phalanx of SVH

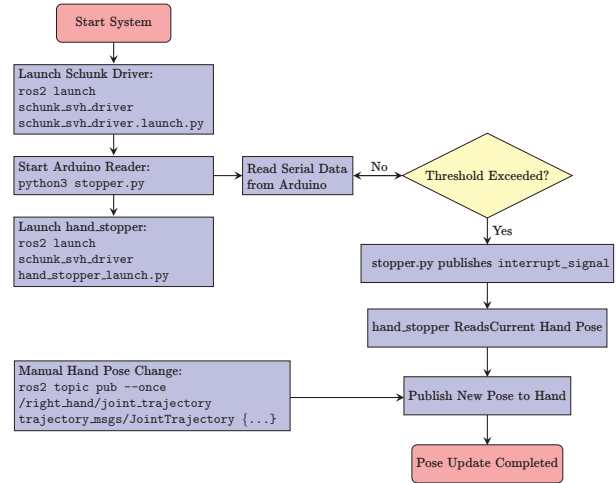


Fig. 4. The flowchart of Schunk SVH communication using ROS2

sensor data in real time and triggers a hand-stop interrupt signal once the force reaches a predetermined threshold. A rising edge detection mechanism ensures that an interrupt signal is only published when the sensor value exceeds a threshold for the first time, preventing redundant commands.

3) *ROS2 Node hand_stopper*: Once an interrupt signal is triggered by stopper.py, the hand_stopper node executes a defined script to halt the hand's movement immediately. This is achieved by retrieving the most recent joint positions from the /joint_states topic and sending this as a new trajectory command to maintain the current pose.

The node subscribes to /joint_states to continuously update an internal dictionary containing the latest joint positions of the right hand. When an interrupt is received, the node:

- Checks if valid joint states have been recorded. If no valid positions are available, it does not issue a stop command to avoid unintended behavior.
- Retrieves the most recent joint positions for the right-hand fingers.
- Constructs a JointTrajectory ROS2 message with these positions as the target.
- Publishes this trajectory to /right_hand/joint_trajectory, ensuring the hand holds its last known position.

To achieve a smooth stop, the trajectory message includes a short time delay (e.g., 50ms) in order to prevent high jerk. This ensures a rapid but controlled halt, preventing excessive force while maintaining stability. The hand remains in this position until a new command is issued, preventing unnecessary fluctuations in grip force.

4) *Performance Analysis*: The system mitigated excessive gripping force by dynamically adjusting the hand's pose in response to high-pressure readings. Key results include a significant reduction in grasping force, ensuring the safe handling of fragile objects. Additionally, real-time data processing enabled immediate response to pressure fluctuations, allowing for precise and timely grip adjustments. Adaptive pose control enhanced the gripper's efficiency and simplified

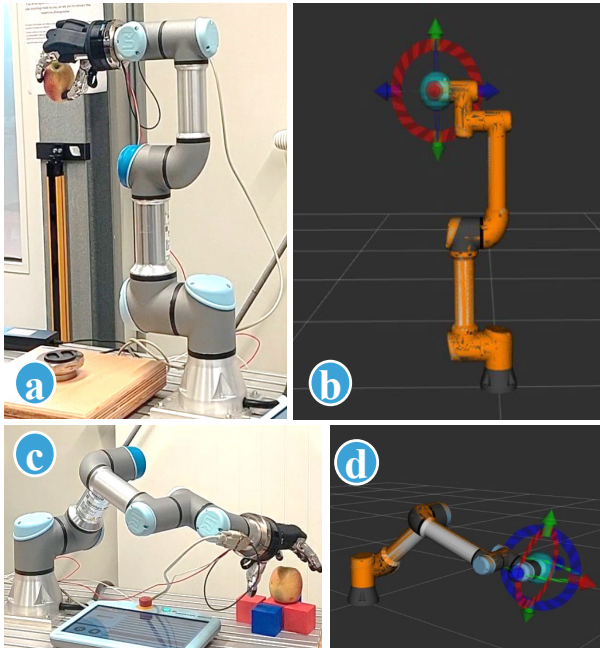


Fig. 5. UR3e home and object-picking position in real and simulated environments

overall control, making it more responsive to varying object shapes and material properties.

III. RESULTS AND DISCUSSIONS

1) *UR3e Motion Execution in Simulation and Real-World:* The validation of robotic grasping systems relies on ensuring that simulated motion planning closely mirrors real-world execution. To achieve this, the UR3e robotic arm was first tested in a simulated environment before deploying the same motion sequences in real-world trials. The Gazebo simulation platform generated and refined motion trajectories, ensuring that the robotic arm's planned movements were accurate and feasible. Two positions are provided to demonstrate the alignment between simulation and reality. Fig. 5 (a) captures the UR3e in a real-world setup at its home position, while Fig. 5 (b) presents the corresponding simulated model in Gazebo. Similarly, Fig. 5 (c) and (d) showcase the robot at the object-picking position in both real and simulated environments. The consistency in joint configurations and movement sequences across both domains highlights the effectiveness of the ROS2-based motion control system in ensuring reliable robotic manipulation. During execution, the UR3e robot follows a structured sequence, beginning from a predefined home position before transitioning into object interaction tasks. The home position establishes a stable and repeatable starting point, improving trial consistency. From this state, the robot moves towards the target object following a planned trajectory, ensuring smooth transitions and avoiding unintended deviations. The inverse kinematics solver calculates the optimal joint configurations, which are first validated in Gazebo before real-world execution. This

step ensures that the simulated robot's movement precisely mirrors the physical robot's behavior, reducing potential errors during deployment. The comparison between simulation and real-world execution confirms the robustness of trajectory planning and motion replication. The UR3e successfully follows pretested motion paths, demonstrating the reliability of ROS2-based control for adaptive robotic applications. The seamless transition from simulation to real execution minimizes risk, improves efficiency, and ensures safe and repeatable grasping operations.

2) *Force-Controlled Grasping and Object Handling:* As the Schunk SVH hand approaches the target, it gradually applies force until reaching a predefined threshold, ensuring a controlled grasp. The threshold varies based on the object's properties, with 5 N used for this demonstration, as shown in Fig. 6. Excessive force application stops once the threshold is met, and the robot moves toward the endpoint while maintaining a stable grip. Upon reaching the target, the robotic hand gradually releases the force, ensuring smooth object placement. This adaptive control prevents slippage, reduces the risk of damage, and ensures secure handling. The results confirm the effectiveness of the force-controlled grasping strategy, where the robotic hand dynamically adjusts its grip to accommodate different objects. The system prevents excessive force while maintaining stability, demonstrating the ROS2-based closed-loop control's reliability. The force trajectory in Fig. 6 highlights stable gripping and controlled release, validating its suitability for adaptive robotic manipulation.

3) *Sequential Adaptive Grasping Demonstration:* The sequence illustrates the adaptive grasping process of a robotic hand, showcasing its transition from an open resting state to precise object manipulation, as shown in Fig. 7. The robotic hand is initially fully open, relaxed, and ready for action. It then spreads its fingers to maximum extension, demonstrating flexibility before gradually moving towards a half-closed state, signaling the beginning of a grasping motion. As the thumb flexes inward, the hand adjusts its posture for an impending grasp. During this transition, the robotic hand momentarily forms expressive gestures, including the "rock and roll" and "peace sign," highlighting its dexterity and human-like articulation. Moving beyond expressive gestures, the hand focuses on functional grasping, positioning itself precisely over an object in the hovering phase, preparing for contact. It then executes a precision grip, delicately

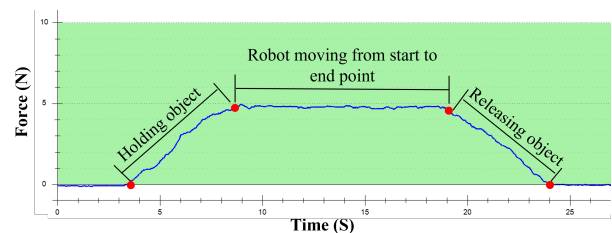


Fig. 6. Force profile of adaptive grasping and object handling

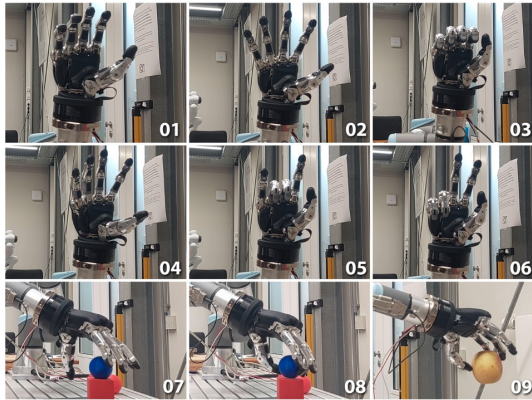


Fig. 7. Sequential demonstration of robotic hand gestures and grasping

securing a small spherical object, emphasizing controlled finger movements. Finally, the robotic hand can gently handle fragile objects, carefully lifting an apple, ensuring a secure yet sensitive grasp. This sequence effectively conveys the robot's capability for expressive gestures and intricate object manipulation, reinforcing its potential for advanced robotic applications.

IV. CONCLUSION

This work demonstrated a ROS2-based force-controlled grasping system, integrating the UR3e robotic arm and Schunk SVH hand with force sensors for adaptive and precise object manipulation. A closed-loop control mechanism was implemented, where force feedback from the sensors dynamically regulated grip strength, ensuring secure yet non-damaging handling of objects. The system was fully developed within ROS2, utilizing MoveIt2 for motion planning, RTDE for real-time execution, and Gazebo simulations for safe validation before deployment. The experimental validation demonstrated that the robotic hand successfully adjusted its grip in response to sensor feedback, preventing excessive force application while maintaining a stable grasp. Simulated trajectories in Gazebo closely mimicked real-world execution, confirming the accuracy and reliability of the ROS2-based motion planning and control framework. The results highlight the effectiveness of the force-regulated grasping strategy, allowing the system to handle fragile and rigid objects with appropriate force levels. The force profile analysis showed smooth transitions in gripping, transporting, and releasing objects, validating the system's adaptability. The structured motion execution, starting from a home position to object interaction, further ensured repeatability and consistency across trials. Transferring simulation-based planning to real-world execution minimized errors and enhanced efficiency, making the approach viable for various robotic manipulation tasks.

Future work will integrate computer vision-based object recognition to enable autonomous pick-and-place operations. This will allow the robot to adjust force thresholds based on detected object properties dynamically, improving adaptabil-

ity. Expanding the system's multi-finger coordination will enhance grasping dexterity, making it suitable for complex industrial automation, assistive robotics, and logistics applications. The proposed approach provides a scalable and efficient solution for adaptive robotic grasping in unstructured environments.

REFERENCES

- [1] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 348–353.
- [2] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on robotics*, vol. 30, no. 2, pp. 289–309, 2013.
- [3] M. R. Cutkosky *et al.*, "On grasp choice, grasp models, and the design of hands for manufacturing tasks," *IEEE Transactions on robotics and automation*, vol. 5, no. 3, pp. 269–279, 1989.
- [4] C. Della Santina, C. Piazza, G. Grioli, M. G. Catalano, and A. Bicchi, "Toward dexterous manipulation with augmented adaptive synergies: The pisa/iiit soft-hand 2," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1141–1156, 2018.
- [5] B. K. Farkas, P. Galambos, and K. Széll, "Advances in autonomous robotic grasping: An overview of reinforcement learning approaches," in *2024 IEEE 6th International Symposium on Logistics and Industrial Informatics (LINDI)*. IEEE, 2024, pp. 000 213–000 220.
- [6] Z. Kappassov, J.-A. Corrales, and V. Perdereau, "Tactile sensing in dexterous robot hands," *Robotics and Autonomous Systems*, vol. 74, pp. 195–220, 2015.
- [7] Z. Kingston and L. E. Kavraki, "Robowflex: Robot motion planning with moveit made easy," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 3108–3114.
- [8] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. Ieee, 2004, pp. 2149–2154.
- [9] A. P. Lindvig, I. Iturrate, U. Kindler, and C. Sloth, "ur_rtde: An interface for controlling universal robots (ur) using the real-time data exchange (rtde)," in *2025 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 2025, pp. 1118–1123.
- [10] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*. PMLR, 2018, pp. 734–743.
- [11] M. Q. Mohammed, L. C. Kwek, S. C. Chua, A. Al-Dhaqm, S. Nahavandi, T. A. E. Eisa, M. F. Miskoon, M. N. Al-Mhiqani, A. Ali, M. Abaker, *et al.*, "Review of learning-based robotic manipulation in cluttered environments," *Sensors*, vol. 22, no. 20, p. 7938, 2022.
- [12] U. E. Ogenyi, J. Liu, C. Yang, Z. Ju, and H. Liu, "Physical human-robot collaboration: Robotic systems, learning methods, collaborative strategies, sensors, and actuators," *IEEE transactions on cybernetics*, vol. 51, no. 4, pp. 1888–1901, 2019.
- [13] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, 2009, p. 5.
- [14] S. W. Ruehl, C. Parltitz, G. Heppner, A. Hermann, A. Roennau, and R. Dillmann, "Experimental evaluation of the schunk 5-finger gripping hand for grasping tasks," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*. IEEE, 2014, pp. 2465–2470.
- [15] B. Siciliano, O. Khatib, and T. Kröger, *Springer handbook of robotics*. Springer, 2008, vol. 200.
- [16] J. Tegin and J. Wikander, "Tactile sensing in intelligent robotic manipulation—a review," *Industrial Robot: An International Journal*, vol. 32, no. 1, pp. 64–70, 2005.
- [17] W. Wang, J. Wang, Y. Luo, X. Wang, and H. Song, "A survey on force sensing techniques in robot-assisted minimally invasive surgery," *IEEE Transactions on Haptics*, vol. 16, no. 4, pp. 702–718, 2023.
- [18] B. Zhang, J. Zhou, Y. Meng, N. Zhang, B. Gu, Z. Yan, and S. I. Idris, "Comparative study of mechanical damage caused by a two-finger tomato gripper with different robotic grasping patterns for harvesting robots," *Biosystems Engineering*, vol. 171, pp. 245–257, 2018.

Category-Level and Open-Set Object Pose Estimation for Robotics

Peter Hönig, Matthias Hirschmanner, and Markus Vincze

Abstract—Object pose estimation enables a variety of tasks in computer vision and robotics, including scene understanding and robotic grasping. The complexity of a pose estimation task depends on the unknown variables related to the target object. While instance-level methods already excel for opaque and Lambertian objects, category-level and open-set methods, where texture, shape, and size are partially or entirely unknown, still struggle with these basic material properties. Since texture is unknown in these scenarios, it cannot be used for disambiguating object symmetries, another core challenge of 6D object pose estimation. The complexity of estimating 6D poses with such a manifold of unknowns led to various datasets, accuracy metrics, and algorithmic solutions. This paper compares datasets, accuracy metrics, and algorithms for solving 6D pose estimation on the category-level. Based on this comparison, we analyze how to bridge category-level and open-set object pose estimation to reach generalization and provide actionable recommendations.

Index Terms—object pose estimation, symmetry handling, instance level, category-level, novel object, open set

I. INTRODUCTION

Object pose estimation is necessary in robotics for tasks such as robotic grasping [1]. If the geometry of a target object is known, its 6D pose, defined by the rotation $\mathbf{R}$ and translation $\mathbf{t}$, is sufficient to locate it in a $SE(3)$ space. This definition is insufficient as soon as the target object geometry is only partially known, as in the case of category-level object pose estimation. Open-set object pose estimation is even more complex than category-level object pose estimation since both geometry and texture are entirely unknown. In category-level and open-set object pose estimation, a sole 6D pose leaves unknowns to describe the object for tasks such as grasping. In these cases, additional information is necessary, such as a 7D pose ($\mathbf{R}$, $\mathbf{t}$, and s for scale), a 9D pose ($\mathbf{R}$, $\mathbf{t}$, and s , a 3D vector with x, y, z dimensions of the aligned bounding box), or a 6D pose in combination with a shape reconstruction. The differences between instance-level, category-level, and open-set object pose estimation are illustrated in Fig. 1. The three circles in Fig. 1 represent the prior knowledge available to a pose estimation algorithm during a training or onboarding stage. During the inference stage existing category-level algorithms [2], [3], [4] do not require a 3D object model. However, in open-set object pose estimation, recent methods do require an object model during inference [5], or they reconstruct an object mesh from multiview RGB images during an onboarding stage [6]. These reconstructions however are prone to reconstruction noise and the full object surface needs to be visible in order

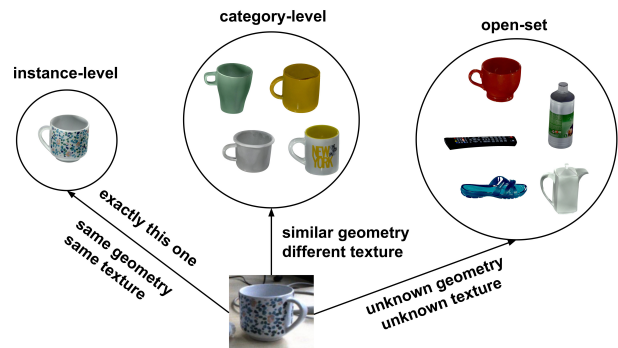


Fig. 1. Comparison of instance-level, category-level, and open-set object pose estimation. The complexity of the pose estimation task is increasing from left (instance-level) to right (open-set) due to the number of unknowns.

for a full reconstruction. This does not allow one-shot pose estimation, where only a single frame is available.

In order to make 3D object models during inference obsolete, knowledge has to be induced during a training stage. Consequently, canonicalization is required. Canonicalization describes the centering and alignment of objects canonically in the $SE(3)$ space. To address the issues of texture and geometry variation, a color encoding of object geometry is proposed in [2], namely the Normalized Object Coordinate Space (NOCS). NOCS describes a dimensionless $1 \times 1 \times 1$ cube, where the x, y , and z coordinates of canonically oriented objects are mapped to RGB values. This geometrical color encoding is used in other category-level and open-set object pose estimation solutions [7], [8], [9], [10], [11], [3], [12], becoming the de-facto standard intermediate representation in the field.

Besides dealing with texture and shape variations, properly disambiguating object symmetries remains challenging in category-level object pose estimation. Not accounting for object symmetries may prohibit optimization algorithms from converging correctly. While NOCS does disambiguate geometrically symmetric objects with distinct textures, it does not account for textureless objects and pointcloud-only input data. While instance-level object pose estimation uses distinct textures to disambiguate geometric symmetries, category-level object pose estimation deals with variation in object texture. This texture variety is handled differently during training of neural networks for category-level object pose estimation [11], [3].

This comparative study analyzes category-level object pose estimation papers [2], [7], [8], [13], [9], [10], [4], [14], [11], [3], [15], [12] focusing on symmetry handling and how

All authors are with the Automation and Control Institute, Faculty of Electrical Engineering, TU Wien, 1040 Vienna, Austria {hoenig, hirschmanner, vincze}@acin.ac.tuwien.at

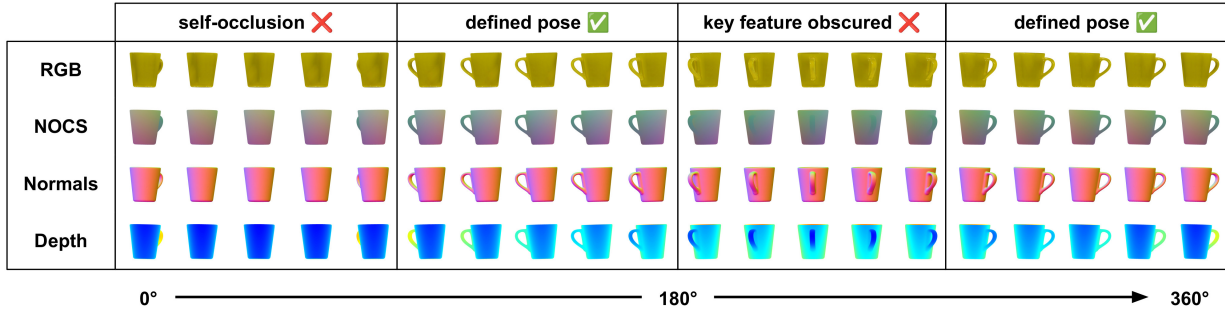


Fig. 2. **Comparing Input Modalities.** A mug is rotated 360° around the y-axis to showcase how a key feature (the handle) is self-occluded in a 90° range, clearly exposed in a 180° range, and obscured due to the uniform texture in a 90° range. The various input modalities are shown to highlight the mug handle stronger (depth, normals) or weaker (RGB, NOCS).

to bridge category-level and open-set object pose estimation. We limit our investigation to algorithms predicting the object pose from a single frame, with no 3D object model available during inference. We only consider algorithms which are evaluated either on the CAMERA [2] or REAL275 [2] dataset. Algorithms that use stereo vision [16] or consecutive frames [17] or are solely evaluated on other datasets are not included in this study. While previous works summarize and review the state of the art in object pose estimation in general [1], [18], [19], we explicitly focus on category-level object pose estimation and how to bridge this technique to the open-set domain, without requiring 3D object models during the inference step. Our contributions can be summarized as follows:

- A concise review of the state of the art in category-level object pose estimation with an emphasis on input modalities, network architectures, 6D pose solvers, and rotational symmetry handling.
- Actionable recommendations for tailoring future category-level object pose estimation methods for generalization beyond known categories to bridge to the open-set domain.

The following section starts by discussing types of input modalities. We will discuss how input modalities influence symmetry disambiguation. The subsequent section reviews network architectures. In this section we will discuss how network architectures can handle symmetric objects differently and how network types relate to model performances. Next, we discuss types symmetry handling, their advantages and limitations. We continue with a section about 6D pose solvers, discussing deterministic and learned variants and we elaborate on pose estimation performance metrics used for comparison. We conclude with experiments and results, discussing current performances of category-level object pose estimation algorithms and give actionable recommendations for potential improvements based on the findings during our literature research. The papers are compared in Table I.

II. INPUT MODALITIES

The choice of input modalities influences how symmetries are resolved [1]. For specific object geometries, RGB-only

leads to ambiguous views, where key features are not clearly exposed, e.g., when the handle of a textureless mug directly points toward the camera. This phenomenon is depicted in Fig. 2. Depth input provides complementary geometric cues, which resolves such ambiguities. Furthermore, since texture and shape are varying in category-level object pose estimation, depth or normals input provide actual geometric surface information. Since category-level object pose estimation algorithms do not access 3D object meshes during the inference stage, algorithms using depth or normals are in favor. In robotic applications, depth data is abundant [1], [20] making algorithms that use depth modality the preferred option.

The papers [2], [7], [8], [9], [3], [15], [12] use RGB input for their algorithms, as listed in Table I. [2], [7] use RGB for establishing 2D-3D correspondences and depth for transforming normalized 3D to metric 3D coordinates. [9], [15] use RGB and depth for predicting 2D-3D correspondences and depth again for 6D pose solving between normalized and metric 3D. [13], [10], [4], [14], [11] rely on depth only for pose estimation and use RGB for the object detection stage to estimate region proposals. [8], [12] use solely RGB data, while [12] extracts DINOv2 features from RGB before processing further. [3] use RGB, depth, and DINO features.

III. NETWORK ARCHITECTURES

Network architectures vary between the selected papers. As Table I indicates, recent advancements in computer vision research are reflected in the development of novel category-level object pose estimation methods. While traditional CNN architectures dominated from 2019 to 2021, Graph Convolutional Networks (GCNs) [25] were adopted in 2022. [3] use diffusion in 2024, and [11], [12] use transformer models in 2023 and 2024. When comparing pose estimation accuracy in the subsequent chapters, one must consider that GCNs, transformers, and diffusion models need extended training and inference times compared to CNNs due to their different network characteristics. This is due to transformers processing the input in multiple heads [26], diffusion models requiring multiple denoising steps [27] and GCNs dealing with not only 2D but 3D data.

TABLE I

COMPARISON OF CATEGORY-LEVEL OBJECT POSE ESTIMATION PAPERS. PAPERS PRESENTING ALGORITHMS PREDICTING CATEGORY-LEVEL OBJECT POSE FROM SINGLE FRAMES.

Year	Paper	Input		NOCS	Symmetry Handling	Network	6D Pose Solver
		RGB	Depth				
2019	Wang et al. [2]	✓	✓	✓	sym. transform loss [21]	Mask R-CNN like	Umeyama [22]
2020	Tian et al. [7]	✓	✓	✓	sym. transform loss [21]	Encoder-decoder CNN	Umeyama [22]
2021	Lee et al. [8]	✓	✗	✓	none	Encoder-decoder CNN	Umeyama [22]
2021	Chen et al. [13]	✗	✓	✗	sym. transform loss [21]	Encoder-decoder CNN	Direct regression
2021	Wang et al. [9]	✓	✓	✓	none	Recurrent reconstruction CNN	Umeyama [22]
2022	Zhang et al. [10]	✗	✓	✓	sym. transform loss [21]	3D GCN	Direct regression
2023	Wan et al. [4]	✗	✓	(✓)	none	3D GCN	Anisotropic scaling
2023	Wang et al. [14]	✗	✓	✓	none	CNNs + MLPs	Direct regression
2023	Zou et al. [11]	✗	✓	✓	sym. transform loss [21]	Transformer	Umeyama [22]
2023	Remus et al. [23]	✓	✓	✗	none	Encoder-decoder CNN	Direct regression
2024	Ikeda et al. [3]	✓	✓	✓	probabilistic	Diffusion model	TEASER++ [24]
2024	Fan et al. [15]	✓	✓	✓	none	Encoder-decoder CNN	Umeyama [22]
2024	Krishnan et al. [12]	✓	✗	✓	none	Transformer	Direct regression

IV. 6D POSE SOLVER

None of the selected papers directly regress the 6D, 7D, or 9D pose from the input without intermediate steps. Regressing object poses without intermediate representations was shown to be inefficient [28]. All selected papers first predict an intermediate representation and solve the 6D pose subsequently. The papers [2], [7], [8], [9], [10], [11], [14], [3], [12] use NOCS as intermediate representation while [4] use an adapted version of NOCS, namely the Semantically-aware Object Coordinate Space (SOCS), a representation similar to NOCS with additional parameters to highlight semantically meaningful regions around keypoints. [13] perform pointcloud reconstruction and regress $\mathbf{R}$, $\mathbf{t}$, and s directly from latent features of the encoder-decoder network.

For networks that estimate the NOCS (or SOCS) pointcloud $\mathbf{P}_N$ in normalized space N , a final step for transforming $\mathbf{P}_N$ to the metric space M is needed to acquire $\mathbf{P}_M$. This step involves solving the equation: $\mathbf{P}_M = s \cdot \mathbf{R} \cdot \mathbf{P}_N + \mathbf{t}_M$, where $\mathbf{R}$, $\mathbf{t}_M$, and s are unknown. The authors of [2], [7], [8], [9], [11], [15] use the Umeyama algorithm to solve for $\mathbf{R}$, $\mathbf{t}_M$, and s , and use metric depth data from a sensor to acquire $\mathbf{P}_M$. [8] uses a Metric Scale Object Shape (MSOS) branch to estimate a metric 3D model parallel to predicting $\mathbf{I}_N$. Therefore, they predict both $\mathbf{I}_N$ and $\mathbf{P}_M$ without using depth data from a sensor. The independence from depth comes with the drawback of increased runtime and limited performance since the MSOS branch can only interpolate between object models encountered during training. Objects with measurements beyond the ones seen in the training data (e.g., a realistic model car, vastly smaller than an actual car but with the same semantic properties) will lead to wrong results. [3] use the TEASER++ algorithm to solve for $\mathbf{R}$, $\mathbf{t}_M$, and s , and use depth data from a sensor to acquire $\mathbf{P}_M$. [12] directly regress $\mathbf{R}$ from $\mathbf{I}_N$, and regress $\mathbf{t}_M$ and s after the Dense Prediction Transformer (DPT) backbone [29]. [12] predict NOCS from a full scene with full semantic context, which helps to learn metric object size without depth data. Still, object sizes outside of the distribution seen during training will be challenging for such a model, similar to [8].

In regards to generalization, the deterministic algorithms Umeyama and TEASER++ do have the advantage of object category agnosticity. While [12] performs direct regression for 6D pose solving, added depth and a deterministic pose solver would most likely result in improved performance.

V. ROTATIONAL SYMMETRY HANDLING

While the pose of the textured object is distinct in all four views, the pose of the textureless object is not. The selected papers of this comparative study handle symmetry differently. Besides no explicit symmetry handling, three major symmetry handling techniques are used.

A. No explicit symmetry handling

[8], [9], [4], [14], [15], [12] do not mention any implicit nor explicit symmetry handling technique. Pose estimation performance is reported for the whole dataset, not single object categories.

B. Orthogonal vectors

[13] do not predict $\mathbf{R}$ as a 3x3 matrix but instead use two decoders that estimate two perpendicular vectors to describe $\mathbf{R}$. For objects with continuous symmetries around one axis the loss weight for one of the two perpendicular vectors is set to 0. This eliminates the constraint of predicting a fully defined pose, but is only applicable to continuous symmetries. Since this form of symmetry handling is explicit, symmetry types have to be manually annotated. For handling discrete symmetries [13] use the symmetry transform loss [21] described in the following section.

C. Symmetric transform loss

To handle discrete and continuous symmetries [2], [7], [13], [11], [10] use the symmetric transform loss described by [21]. The symmetric transform loss $\mathcal{L}_{\text{sym}}$ can be defined as:

$$\mathcal{L}_{\text{sym}} = \min_{\mathbf{R} \in \mathcal{S}} \mathcal{L}(\mathbf{P}_{\text{est}}, \mathbf{R} \cdot \mathbf{P}_{\text{gt}}),$$

where:

- $\mathbf{P}_{\text{est}}$ is the estimated NOCS point cloud.

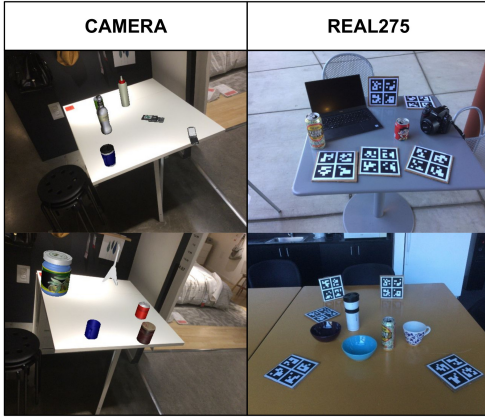


Fig. 3. Exemplary images of the CAMERA and REAL275 datasets. While the CAMERA dataset features rendered object instances on real backgrounds, the REAL275 dataset solely features real image data.

- $\mathbf{P}_{\text{gt}}$ is the ground truth NOCS point cloud.
- $\mathcal{S} = \{\mathbf{R}_1, \mathbf{R}_2, \dots, \mathbf{R}_n\}$ is the set of all rotational symmetry transformations.
- $\mathcal{L}(\mathbf{P}_1, \mathbf{P}_2)$ is the loss function measuring the distance between two point clouds.

The symmetry transform loss requires handcrafted annotations of symmetrical poses for all objects in the training set. Annotation requires the symmetries to be discrete (e.g., a rotational symmetry has to be divided into n discrete symmetries around that axis). Furthermore, since real-life objects are either reconstructed or modeled with CAD, an arbitrary choice of symmetry is necessary. Reconstructed meshes are never fully symmetric due to reconstruction noise, and CAD parts may break symmetry only by a minor part of the object in relation to the full size. While these alterations technically break the symmetry, defining them as symmetric may still lead to better convergence when predicting NOCS.

D. Probabilistic symmetry handling

[3] presents an implicit symmetry handling approach for learning pose probabilities by sampling an additional noise input to the diffusion model. Compared to the other papers, they sample multiple noise inputs for each training sample drawn from the dataloader. Consequently, each instance of the training set has newly generated noise samples in each epoch during training. After predicting the NOCS the inlier rate of the TEASER++ pointcloud registration result is used as an additional loss for backpropagation. An example of this noise sampling approach is illustrated in Fig. 4. The advantage of this type of symmetry handling is that no discrete symmetry annotations are necessary.

VI. EXPERIMENTS

The selected papers report results on the REAL275 and Context-Aware MixEd ReAlity (CAMERA) datasets. Both datasets were introduced in [2]. Exemplary images are shown

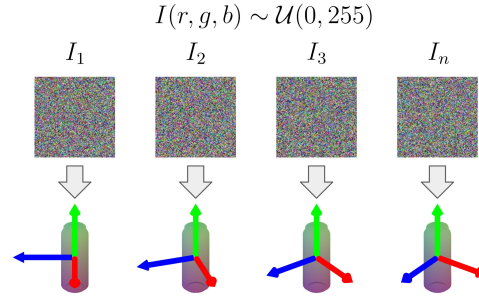


Fig. 4. Illustration of the probabilistic loss used by DiffusionNOCS. For each sample around the symmetry axis r,g,b values from a uniform Gaussian distribution are sampled. This introduces an adversarial character to the NOCS prediction, prohibiting the network from converging to local minima that do not capture the full symmetry of the object. By introducing this adversarial effect the network learns the symmetry of the object implicitly.

in Fig. 3. The CAMERA dataset consists of 300k synthetically rendered object images pasted onto real backgrounds. The REAL275 dataset contains 2750 test images across 18 different scenes. Both datasets contain the same object instances of 6 different object categories (bottle, bowl, camera, can, laptop, and mug), consisting of non-rotation-symmetric (camera, laptop, mug) and continuously symmetric (bottle, bowl, can) objects. Since the CAMERA dataset also uses synthetic object renderings for the test data, its relevance for evaluating algorithms for real-world pose estimation capabilities is questionable. If the test data is synthetic, the performance of algorithms on the real-world domain can not be evaluated.

Regarding accuracy metrics, the authors of [2], [7], [8], [12] report the mean Average Precision (mAP) of $3D_{25}$, $3D_{50}$, and $3D_{75}$ scores, which represent the Intersection over Union (IoU) between ground truth and estimated 3D bounding box at 25%, 50%, and 75% respectively. The authors of [2], [7], [8], [3] papers also report the mAP below error thresholds of $\mathbf{R}$ and $\mathbf{t}$. The specific thresholds in the papers vary. Therefore, the most common ones are selected ($5^\circ 5cm$, $10^\circ 5cm$, and $10^\circ 10cm$).

VII. RESULTS

Table II shows the results presented in the selected papers on the CAMERA and REAL275 datasets. Out of all, the two papers that use solely RGB data, [8], [12] perform worst. This confirms the hypothesis that regressing object size and translation without depth data during runtime is challenging. While [8] report 75.4% of 3D bounding boxes within an IoU of 25%, the numbers drop to 32.4% at an IoU of 50% and 5.1% at an IoU of 75%. An opposite trend can be observed for the methods using solely depth. The best performing methods for 3D bounding box and $\mathbf{R}$, $\mathbf{t}$ prediction rely on depth only. Especially [4], [14] appear to be the best performing methods on the REAL275 dataset for predicting $\mathbf{R}$, $\mathbf{t}$. [4] is slightly in the lead with 56.0% mAP $5^\circ 5cm$ and 82.0% $10^\circ 5cm$. [14] is close with 54.7% mAP $5^\circ 5cm$ and 81.6% $10^\circ 5cm$. Both [4], [14] do not perform explicit symmetry handling. While performance

TABLE II
EVALUATION ON THE CAMERA AND REAL275 DATASETS. MAP SCORES FOR 3D BOUNDING BOXES, **R** AND **t**. "-" INDICATES SCORES NOT REPORTED IN THE RESPECTIVE PAPERS. BEST PERFORMANCE IN **BOLD**, WORST PERFORMANCE / RGB-ONLY UNDERLINED

[†] USE SYNTHETICALLY RENDERED TRAINING DATA, EVALUATE ON REAL TEST DATA.

Dataset	Year	Paper	Input	Priors	mAP					
					3D ₂₅	3D ₅₀	3D ₇₅	5°5cm	10°5cm	10°10cm
CAMERA	2019	Wang et al. [2]	RGB-D	end-to-end	91.1	83.9	69.5	40.9	64.6	65.1
	2020	Tian et al. [7]	RGB-D	Mask R-CNN [30]	-	93.2	83.1	59.0	81.5	-
	2021	Lee et al. [8]	RGB	Mask R-CNN [30]	<u>75.4</u>	<u>32.4</u>	<u>5.1</u>	-	-	<u>19.2</u>
	2021	Chen et al. [13]	D	YOLOv3 [31]	-	-	85.2	62.0	-	-
	2022	Wang et al. [9]	RGB-D	Mask R-CNN [30]	-	93.8	88.0	76.4	87.7	-
	2023	Zhang et al. [10]	D	Mask R-CNN [30]	-	-	86.8	75.5	87.4	-
	2023	Wan et al. [4]	D	Mask R-CNN [30]	-	-	-	-	-	-
	2023	Wang et al. [14]	D	Mask R-CNN [30]	-	92.3	88.6	83.9	90.5	-
	2023	Zou et al. [11]	D	Mask R-CNN [30]	-	92.5	86.9	76.5	88.7	89.9
	2024	Ikeda et al. [3]	RGB-D	Mask R-CNN [30]	-	-	-	-	-	-
	2024	Fan et al. [15]	RGB-D	Mask R-CNN [30]	-	93.7	89.6	75.1	89.5	-
	2024	Krishnan et al. [12]	RGB	Mask R-CNN [30]	-	-	-	-	-	-
REAL275	2019	Wang et al. [2]	RGB-D	end-to-end	84.9	80.5	30.1	10.0	26.7	26.7
	2020	Tian et al. [7]	RGB-D	Mask R-CNN [30]	-	77.3	53.2	21.4	54.1	-
	2021	Lee et al. [8]	RGB	Mask R-CNN [30]	<u>62.0</u>	<u>23.4</u>	<u>3.0</u>	-	-	<u>9.6</u>
	2021	Chen et al. [13]	D	YOLOv3 [31]	95.1	92.2	63.5	28.2	60.8	64.6
	2021	Wang et al. [9]	RGB-D	Mask R-CNN [30]	-	79.3	55.9	34.3	60.8	-
	2022	Zhang et al. [10]	D	Mask R-CNN [30]	84.0	81.1	52.0	33.9	69.1	71.0
	2023	Wan et al. [4]	D	Mask R-CNN [30]	-	82.0	75.0	56.0	82.0	-
	2023	Wang et al. [14]	D	Mask R-CNN [30]	-	82.9	76.0	54.7	81.6	-
	2023	Zou et al. [11]	D	Mask R-CNN [30]	-	82.0	70.4	53.8	77.7	79.8
	2024	Ikeda et al. [3] [†]	RGB-D	Mask R-CNN [30]	-	-	-	35.0	66.6	-
	2024	Fan et al. [15]	RGB-D	Mask R-CNN [30]	-	82.3	66.6	41.3	67.0	-
	2024	Krishnan et al. [12]	RGB	Mask R-CNN [30]	<u>43.5</u>	<u>10.6</u>	-	-	-	-

metrics for individual object categories are missing in the papers, it appears that the 3D GCN of [4] and the CNNs + MLP network of [14] handle the symmetries of objects well enough. Both papers also use learning-based 6D pose solving, namely anisotropic scaling [4] and direct regression [14]. This sophisticated 6D pose solving techniques could be the reason for the superior performances. On the other hand, [11] comes close to the results of [4], [14] while employing the deterministic Umeyama method [22] for rigid point cloud alignment. The probabilistic loss of [3] does not lead to improved performance. However, since the authors of [3] use synthetic renderings for training and real data for evaluation, a fair comparison is not possible.

When comparing pose estimation performance, the source of object detection priors have to be taken into account. While [7], [8], [9], [10], [4], [14], [11], [3], [15], [12] use pre-computed Mask R-CNN location priors to ensure fair comparison, [2] use the location priors of their end-to-end trainable network and [13] use YOLOv3 for location priors. This results in superior results for 3D bounding box estimation on REAL275 by [13].

Overall, results are better on the CAMERA dataset compared to the REAL275 dataset. This can likely be attributed to the stronger domain shift between the training and test data of REAL275. Not only objects but also scenes are different, including other lighting, shadow and reflection. The domain shift for pose estimation only consists of object difference between training and test dataset.

Lastly, a clear correlation between improved performance and newer network architectures such as transformers or

diffusion models cannot be observed.

VIII. CONCLUSION

This paper compares category-level object pose estimation methods which are evaluated on the CAMERA and REAL275 datasets. The methods differ regarding input modalities, symmetry handling, network types, and 6D pose solver algorithms. A comprehensive comparison was conducted, focusing on symmetry handling and its potential impact on model performance. After reviewing input modalities, network architectures, 6D pose solvers, symmetry handling, experiments, and results, the following conclusions are drawn:

- Omitting depth as done by [8], [12] drastically reduces performance as compared to RGB and RGB-D based methods.
- While the absence of depth data worsens results, the depth-only methods perform best overall, indicating that depth data is crucial for improving category-level object pose estimation.
- The two best performing methods use no explicit symmetry handling, suggesting that implicit symmetry handling is not mandatory if model architecture is allowing for it.
- While methods with learning-based 6D pose solvers excel regarding pose estimation performance, papers using deterministic methods such as Umeyama [22] achieve results almost on par. This is crucial since bridging category-level and open-set pose estimation benefits

from deterministic 3D geometry-agnostic algorithms for 6D pose solving.

- The usage of different 2D object detection priors hinders a fair comparison, since improvement cannot clearly attributed to either detection or pose estimation.

Future research should build upon this comparative paper by analyzing the individual aspects of category-level object pose estimation further.

ACKNOWLEDGMENT

This research is supported by the EU program EC Horizon 2020 for Research and Innovation under grant agreement No. 101017089, project TraceBot, and the Austrian Science Fund (FWF), under project No. I 6114, iChores.

REFERENCES

- [1] S. Thalhammer, D. Bauer, P. Hönig, J.-B. Weibel, J. García-Rodríguez, and M. Vincze, "Challenges for monocular 6-d object pose estimation in robotics," *IEEE Transactions on Robotics*, vol. 40, pp. 4065–4084, 2024.
- [2] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6D object pose and size estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2642–2651.
- [3] T. Ikeda, S. Zakharov, T. Ko, M. Z. Irshad, R. Lee, K. Liu, R. Ambrus, and K. Nishiwaki, "Diffusionnocs: Managing symmetry and uncertainty in sim2real multi-modal category-level pose estimation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 7406–7413.
- [4] B. Wan, Y. Shi, and K. Xu, "Socs: Semantically-aware object coordinate space for category-level 6d object pose estimation under large shape variations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 14 065–14 074.
- [5] P. Ausserlechner, D. Habegger, S. Thalhammer, J.-B. Weibel, and M. Vincze, "Zs6d: Zero-shot 6d object pose estimation using vision transformers," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 463–469.
- [6] E. P. Örnek, Y. Labbé, B. Tekin, L. Ma, C. Keskin, C. Forster, and T. Hodan, "Foundpose: Unseen object pose estimation with foundation features," in *European Conference on Computer Vision*. Springer, 2024, pp. 163–182.
- [7] M. Tian, M. H. Ang, and G. H. Lee, "Shape prior deformation for categorical 6d object pose and size estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2020, pp. 530–546.
- [8] T. Lee, B.-U. Lee, M. Kim, and I. S. Kweon, "Category-level metric scale object shape and pose estimation," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8575–8582, 2021.
- [9] J. Wang, K. Chen, and Q. Dou, "Category-level 6d object pose estimation via cascaded relation and recurrent reconstruction networks," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4807–4814.
- [10] R. Zhang, Y. Di, F. Manhardt, F. Tombari, and X. Ji, "Ssp-pose: Symmetry-aware shape prior deformation for direct category-level object pose estimation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7452–7459.
- [11] L. Zou, Z. Huang, N. Gu, and G. Wang, "Gpt-cope: A graph-guided point transformer for category-level object pose estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.
- [12] A. Krishnan, A. Kundu, K.-K. Maninis, J. Hays, and M. Brown, "Omninocs: A unified nocs dataset and model for 3d lifting of 2d objects," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2024, pp. 127–145.
- [13] W. Chen, X. Jia, H. J. Chang, J. Duan, L. Shen, and A. Leonardis, "Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 1581–1590.
- [14] R. Wang, X. Wang, T. Li, R. Yang, M. Wan, and W. Liu, "Query6dof: Learning sparse queries as implicit shape prior for category-level 6dof pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 14 055–14 064.
- [15] Z. Fan, Z. Song, Z. Wang, J. Xu, K. Wu, H. Liu, and J. He, "Acr-pose: Adversarial canonical representation reconstruction network for category level 6d object pose estimation," in *Proceedings of the International Conference on Multimedia Retrieval (ICMR)*. Association for Computing Machinery, 2024, p. 55–63.
- [16] K. Chen, S. James, C. Sui, Y.-H. Liu, P. Abbeel, and Q. Dou, "Stereopose: Category-level 6d transparent object pose estimation from stereo images via back-view nocs," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 2855–2861.
- [17] M. Zaccaria, F. Manhardt, Y. Di, F. Tombari, J. Aleotti, and M. Giorgini, "Self-supervised category-level 6d object pose estimation with optical flow consistency," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2510–2517, 2023.
- [18] G. Marullo, L. Tanzi, P. Piazzolla, and E. Vezzetti, "6d object position estimation from 2d images: A literature review," *Multimedia Tools and Applications*, vol. 82, no. 16, pp. 24 605–24 643, 2023.
- [19] Z. Fan, Y. Zhu, Y. He, Q. Sun, H. Liu, and J. He, "Deep learning on monocular object pose detection and tracking: A comprehensive overview," *ACM Computing Surveys*, vol. 55, no. 4, pp. 1–40, 2022.
- [20] D. Bauer, T. Patten, and M. Vincze, "Verefine: Integrating object pose verification with physics-guided iterative refinement," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4289–4296, 2020.
- [21] G. Pitteri, M. Ramamonjisoa, S. Ilıc, and V. Lepetit, "On object symmetries and 6d pose estimation from images," in *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 614–622.
- [22] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 13, no. 04, pp. 376–380, 1991.
- [23] A. Remus, S. D'Avella, F. D. Felice, P. Tripicchio, and C. A. Avizzano, "i2c-net: Using instance-level neural networks for monocular category-level 6d pose estimation," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1515–1522, 2023.
- [24] H. Yang, J. Shi, and L. Carbone, "Teaser: Fast and certifiable point cloud registration," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2020.
- [25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.
- [26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.
- [27] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
- [28] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5745–5753.
- [29] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 12 179–12 188.
- [30] W. Abdulla, "Mask r-cnn for object detection and instance segmentation on keras and tensorflow," *GitHub repository*, 2017.
- [31] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018. [Online]. Available: <https://arxiv.org/abs/1804.02767>

Simulation-Driven Optimization of Stanley Controller Gains for Enhanced Tracking in Autonomous Navigation Robots

Héctor Pérez-Villeda¹, Clemens Mühlbacher¹, Konstantin Mautner-Lassnig¹

Abstract—Fine-tuning controllers for robotic systems is a tedious process that often requires significant time for convergence and can lead to mechanical component wear. Having an accurate simulation of the robotic system and its environment can help reduce this effort and accelerate the tuning process.

This work presents an optimization-based approach that leverages simulations to optimize control parameters before transferring them to a real mobile robot, significantly reducing fine-tuning effort and the need for extensive real-world testing. The method follows a two-stage process: first, calibrating the simulator to closely replicate the mobile robot's trajectory, and second, using the refined simulation to optimize the Stanley controller's gains. By aligning the simulator's behavior with real-world performance, we ensure that control tuning is both effective and time-efficient, allowing optimized parameters to be directly applied to the real system.

The methodology is validated through experiments comparing simulated and real-world trajectories, demonstrating that the optimized gains improve tracking accuracy. Additionally, we provide an estimation of the achieved improvements, including tracking error reduction, time savings, and energy consumption minimized by our approach, highlighting its efficiency in the fine-tuning process.

Index Terms—Autonomous navigation, Stanley controller, simulator optimization, control tuning, simulation-to-reality transfer, parameter optimization.

I. INTRODUCTION

Autonomous navigation is a critical capability for mobile robots operating in dynamic environments. A key challenge in this domain is ensuring accurate trajectory tracking, which is essential for applications such as autonomous vehicles [6], warehouse automation [9], and field robotics [5]. Stanley controller is widely used due to its effectiveness in minimizing lateral errors and maintaining stability during navigation [11]. However, achieving optimal tracking performance requires careful tuning of the controller's gains, a process that is often time-consuming and tedious when performed directly on a physical robot.

To address this challenge, we propose a simulation-driven optimization framework that enhances the efficiency of Stanley controller gain tuning. Instead of manually adjusting control parameters in real-world experiments, our approach leverages automated hyperparameter optimization techniques to systematically refine both the simulator parameters and

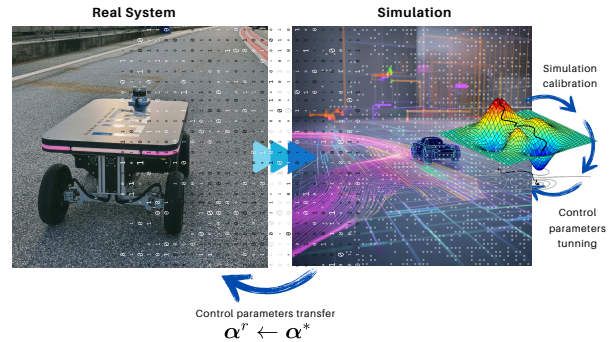


Fig. 1. On the left: CHASI, the ARTI robotic platform used in the experiments. Our method aims to calibrate the simulation environment to accurately replicate the real-world robot behavior. Once calibrated, control parameters are fine-tuned in simulation before transferring the optimized parameters to the real robot. This approach accelerates the tuning process while minimizing mechanical wear on the physical system.

controller gains. Specifically, we employ Optuna [2], an efficient framework for hyperparameter search, to explore optimal configurations while minimizing trajectory tracking errors. By optimizing in simulation before transferring the learned parameters to the real system, our method reduces the need for extensive physical testing while maintaining real-world applicability [7], [8], [14].

The remainder of this paper is structured as follows: Section 2 describes the robot model used in this work. Section 3 defines the environment and the mathematical framework used throughout the paper. Section 4 details the proposed methodology, while Section 5 presents the experimental results along with their analysis. Finally, Section 6 concludes the paper and discusses directions for future work.

II. RELATED WORK

Accurate control tuning for mobile robots is a critical challenge, particularly in scenarios where real-world testing is expensive, risky, or time-consuming. A common strategy to address these challenges involves the use of high-fidelity simulators to replicate the behavior of robotic systems. However, discrepancies between simulation and actual performance, commonly referred to as the sim-to-real gap, can significantly reduce the reliability of control policies if the simulator is not properly calibrated.

Sim-to-real transfer techniques have received increasing attention in robotics. Domain randomization [13] and system identification methods [10] are often employed to bridge the simulation-reality gap by either increasing robustness

* This project is funded by the Austrian Research Promotion Agency (FFG). www.ffg.at, under the project: AVASI (Autonomous Vehicle Advanced Simulation)

¹ARTI — AI software solutions for autonomous robots, Website: <https://arti-robots.com>, Emails: h.villeda@arti-robots.com, c.muehlbacher@arti-robots.com, k.ml@arti-robots.com

to environmental variation or aligning simulator dynamics with the real system. DROPO [12] focus on estimating domain randomization ranges for improving transferability of optimized policies. These methods often assume that the simulation environment and its physical model provides an adequate representation of the real-world system.

Hybrid learning strategies have been proposed to incorporate real-world data for improving simulation fidelity and learning efficiency [4]. Similarly, [1] evaluate various simulators, revealing limitations in modeling elastic impacts and complex motions, even with contact parameter tuning—highlighting the need for more accurate physical calibration.

While these works typically address either simulation calibration or control optimization in isolation, our method introduces an integrated three-stage approach: first refining simulation parameters to match real-world robot behavior, then optimizing control gains within the calibrated environment and finally transfer the learned parameter to the real robot. This ensures that the resulting policies are both physically grounded and readily transferable, minimizing reliance on real-world trials.

III. ROBOT MODEL

A. Ackerman Kinematic Model

The Ackerman kinematic model is widely used to describe the motion of wheeled vehicles with nonholonomic constraints [3]. It assumes no lateral slip and is based on the geometry of steering. The vehicle's motion is governed by the following equations:

$$\begin{aligned}\dot{x} &= v \cos(\theta), \\ \dot{y} &= v \sin(\theta), \\ \dot{\theta} &= \frac{v}{L} \tan(\delta).\end{aligned}\quad (1)$$

where x and y represent the vehicle's position coordinates, θ is the heading angle, v denotes the velocity, L is the wheelbase length, and δ corresponds to the steering angle.

B. ARTI-Controller

The ARTI-Controller is based on the Stanley method [11], a widely used approach for autonomous vehicle path following that minimizes cross-track and heading errors to ensure smooth trajectory convergence.

The steering control law is defined as:

$$\delta_v = \theta_e + \tan^{-1} \left(\frac{k e_{fa}}{v + v_{\min}} \right) \quad (2)$$

where δ_v is the steering output, $\theta_e = \theta - \theta_p$ is the heading error, and e_{fa} is the cross-track error from the front axle to the closest path point (c_x, c_y) . The gain k adjusts the influence of the cross-track error, while v is the vehicle speed, and $v_{\min}$ ensures stability at low speeds.

To enhance robustness and adaptability at different speeds, the ARTI-Controller applies gain scheduling for k across velocity ranges, enabling dynamic tuning of the control response. The gain values are summarized in Table III-B. The

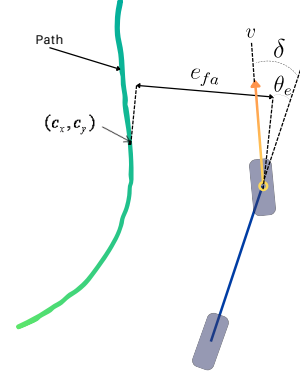


Fig. 2. Diagram illustrating the variables used by the Stanley controller to compute the control output

geometric relationship of the control variables is illustrated in Figure 2.

TABLE I

STANLEY CONTROL GAINS FOR DIFFERENT VELOCITY RANGES.

Velocity	$0 \leq v < 0.5$	$0.5 \leq v < 0.8$	$0.8 \leq v \leq 1.0$
k	k_1	k_2	k_3

IV. ENVIRONMENT DEFINITION

A. Real-World (Physical) System Representation

The real system is modeled as a discrete-time nonlinear state-space system:

$$\mathbf{x}_{i+1}^r = f^r(\mathbf{x}_i^r, \mathbf{u}_i^r, \boldsymbol{\alpha}^r, \mathbf{T}^d) \quad (3)$$

where $\mathbf{x}_i^r = [x_i^r, y_i^r, \theta_i^r]^T$ represents the robot's position and orientation at time step i , and $\mathbf{u}_i^r = [\delta_{v_i}^r, v_i^r]^T$ denotes the steering and linear velocity control inputs. The dynamics function $f^r(\cdot)$ is defined according to the Ackermann model in Equation 1.

The closed-loop controller is parameterized by $\boldsymbol{\alpha}^r = \{k_1^r, k_2^r, k_3^r\}$, which affect the robot's tracking behavior. The desired trajectory is defined as:

$$\mathbf{T}^d = \left\{ \mathbf{x}_i^d = [c_{x_i}, c_{y_i}]^T \mid i = 1, \dots, N^d \right\} \quad (4)$$

where $\mathbf{x}_i^d$ are the target waypoints in Cartesian coordinates.

A sampled trajectory from the real system consists of a discrete sequence of the robot's states, defined as:

$$\mathbf{T}^r_{\boldsymbol{\alpha}^r} = \{\mathbf{x}_i^r \mid i = 1, \dots, N^r\} \quad (5)$$

where each $\mathbf{x}_i^r$ is a recorded state of the robot under the controller parameters $\boldsymbol{\alpha}^r$, and N^r is the number of sampled steps.

B. Simulated System Representation

The simulated system is represented as:

$$\mathbf{x}_{i+1}^s = f^s(\mathbf{x}_i^s, \mathbf{u}_i^s, \boldsymbol{\alpha}^s, \boldsymbol{\beta}, \mathbf{T}^d) \quad (6)$$

where $\mathbf{x}_i^s = [x_i^s, y_i^s, \theta_i^s]^T$ is the simulated state, and $\mathbf{u}_i^s = [\delta_{v_i}^s, v_i^s]^T$ is the control input, with $\delta_{v_i}^s$ as the steering angle

and v_i^s as the linear velocity. The parameters $\alpha^s = \{k_1^s, k_2^s, k_3^s\}$ configure the Stanley controller in simulation, while $\beta = \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5\}$ defines configurable simulation parameters defined in the Table II used to more accurately capture the interaction between the robot and the real system. The function $f^s(\cdot)$ approximates the simulated system's transition dynamics.

A simulated trajectory consists of a sequence of discrete states:

$$\mathbf{T}_{\alpha^s, \beta}^s = \{\mathbf{x}_i^s \mid i = 1, \dots, N^s\} \quad (7)$$

where N^s is the number of recorded samples. The trajectory $\mathbf{T}^s$ depends on the control parameters in the simulation environment α^s and simulation parameters β .

C. System Discrepancy Measure

To quantify the difference between two trajectories $\mathbf{T}_1 = [\mathbf{T}_{1x}, \mathbf{T}_{1y}]$ and $\mathbf{T}_2 = [\mathbf{T}_{2x}, \mathbf{T}_{2y}]$, we define a cost function that integrates well-known metrics: Mean Square Error (MSE) and Dynamic Time Warping (DTW).

$$\mathcal{L}(\mathbf{T}_1, \mathbf{T}_2) = \text{MSE}(\text{DTW}(\mathbf{T}_{1x}, \mathbf{T}_{2x}), \text{DTW}(\mathbf{T}_{1y}, \mathbf{T}_{2y})) \quad (8)$$

DTW first aligns the x - and y -coordinate sequences to account for temporal variations; MSE then measures the deviation between the aligned pairs, yielding a robust similarity metric even under phase shifts or speed differences.

D. Problem Definition

Our objective is to minimize the tracking error between the real trajectory (5) of the physical robot (3) and the desired trajectory (4). This is achieved by fine-tuning the control parameters α . We formulate the optimization problem as:

$$\alpha^* = \arg \min_{\alpha} \mathcal{L}(\mathbf{T}^d, \mathbf{T}_{\alpha^r}^r) \quad (9)$$

where $\mathcal{L}(\mathbf{T}^d, \mathbf{T}_{\alpha^r}^r)$ represents the discrepancy between the desired trajectory $\mathbf{T}^d$ and the real trajectory $\mathbf{T}_{\alpha^r}^r$.

To achieve this, we first use a simulator to capture the robot's initial real-world behavior through a calibration process. Then, we fine-tune the controller gains in the calibrated simulation before transferring these optimized gains to the real system.

V. METHOD

This section provides a detailed description of our method. Figure 3 illustrates the overall process, while Table 1 summarizes the key steps for clarity. Our approach is divided into four main steps:

- 1) **Real-World Data Collection:** Gather the robot's trajectory $\mathbf{T}_{\alpha_0^r}^r$ following the desired trajectory $\mathbf{T}^d$ using the initial control parameters α_0^r .
- 2) **Simulation Calibration:** The goal of this step is to ensure that the robot's simulated trajectory $\mathbf{T}^d$ closely resembles the real trajectory $\mathbf{T}_{\alpha_0^r}^r$ obtained using the initial control parameters. To achieve this, we transfer the initial control parameter values from the real

TABLE II
NOTATION AND NAME OF VARIABLES

Symbol	Description
$\mathbf{T}^d$	Desired trajectory
$\mathbf{T}_{\alpha^r}^r$	Real trajectory from the robot with control parameters α^r
$\mathbf{T}_{\alpha^s, \beta}^s$	Simulated trajectory with parameters α^s, β
α^r, α^s	Control parameters (real-world & sim.)
α_0^r, α_0^s	Initial control parameters (real-world & sim.)
α^*	Optimized control parameters
β	Set of simulation parameters
β_0	Initial simulation parameters
β^*	Optimized simulation parameters
β_1	Delay velocity
β_2	Delay steering
β_3	Max. allowed acceleration
β_4	Max. allowed angular velocity
β_5	Angular acceleration

robot to the simulation, i.e., $\alpha_0^s \leftarrow \alpha_0^r$. Afterwards, we calibrate the simulation to ensure that the simulated robot's behavior closely approximates the real-world system, i.e., $\mathbf{x}_i^s \approx \mathbf{x}_i^r$ by fixing the values of the initial control parameters α_0^s and optimizing the set of configurable simulation parameters β through the following minimization problem:

$$\begin{aligned} \beta^* &= \arg \min_{\beta} \mathcal{L}(\mathbf{T}_{\alpha_0^r}^r, \mathbf{T}_{\alpha_0^s, \beta}^s) \\ &\Rightarrow \mathbf{x}_i^s \approx \mathbf{x}_i^r, \quad \forall i \in \{1, \dots, N\} \end{aligned} \quad (10)$$

The function $\mathcal{L}(\mathbf{T}_{\alpha_0^r}^r, \mathbf{T}_{\alpha_0^s, \beta}^s)$, introduced in (8), quantifies the discrepancy between the simulated and real trajectories.

This process involves executing the path-following task in simulation using the same desired trajectory $\mathbf{T}^d$ from (4) and iteratively adjusting β until the optimal parameters β^* are obtained.

- 3) **Fine-tuning of control parameters:** Once the simulated robot accurately replicates the real robot's behavior, the objective of this step is to optimize the control parameters to ensure the simulated robot closely follows the desired trajectory $\mathbf{T}^d$. To achieve this, after calibrating the simulation state $\mathbf{x}_i^s$, we fix the optimal simulation parameters β^* obtained in the previous step and fine-tune the control gains α^s by solving the following minimization problem:

$$\alpha^* = \arg \min_{\alpha^s} \mathcal{L}(\mathbf{T}^d, \mathbf{T}_{\alpha^s, \beta^*}^s), \quad (11)$$

This optimization process improves tracking accuracy, ensuring that the simulated trajectory $\mathbf{T}^s$ closely aligns with the desired trajectory $\mathbf{T}^d$.

- 4) **Transfer to Real System:** After optimizing the control parameters to ensure the simulated robot closely follows the desired trajectory, we transfer the tuned parameters $\alpha^r \leftarrow \alpha^*$ to the physical system (3) and validate their performance under real-world conditions.

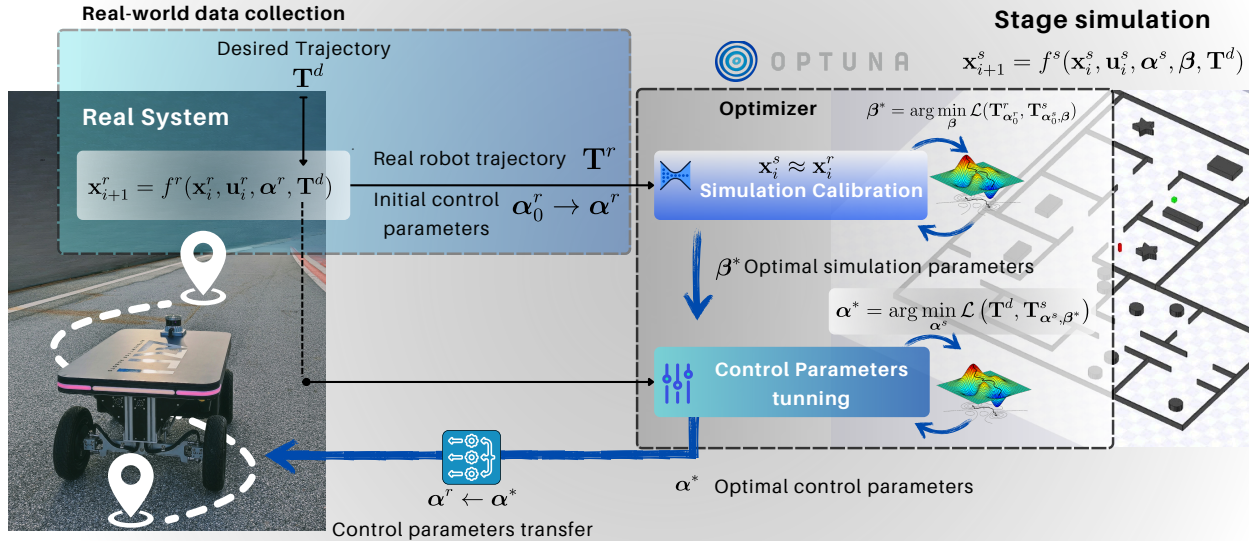


Fig. 3. Pipeline of our method: The process begins by collecting the real robot's trajectory while following the desired path. This trajectory is then used to calibrate the simulation environment iteratively until the simulated behavior closely matches the real-world performance. Once calibrated, control parameters are fine-tuned in simulation and subsequently transferred to the real robot to enhance its tracking accuracy.

Algorithm 1 Simulation-Based Control Gain Optimization

```

1: Input: Desired trajectory  $T^d$ 
2: Output: Optimized control parameters  $\alpha^*$ 
3: Step 1: Real-World Data Collection
4:  $T_{\alpha^r}^r = \{x_i^r \mid i = 1, \dots, N^r\}$ 
5: Step 2: Simulation Calibration  $x_i^{s'} \approx x_i^r$ 
6: Initialize  $\alpha_0^s \leftarrow \alpha_0^r$ ;  $\beta \leftarrow \beta_0$   $\triangleright$  Load simulation default values
7: while  $\mathcal{L}(T_{\alpha^r}^r, T_{\alpha^s, \beta}^s) \geq \epsilon$  do
8:    $T_{\alpha^s, \beta}^s = \{x_i^s \mid i = 1, \dots, N^s\}$ 
9:    $\beta = \beta - \eta \nabla_{\beta} \mathcal{L}(T_{\alpha^r}^r, T_{\alpha^s, \beta}^s)$ 
10: end while
11: Return  $\beta^* \leftarrow \beta$   $\triangleright$  Final optimized parameters
12: Step 3: Fine-Tuning of Control Parameters in sim.
13:  $\beta \leftarrow \beta^*$   $\triangleright$  Load optimized simulation parameters
14: while  $\mathcal{L}(T^d, T_{\alpha^s, \beta}^s) \geq \epsilon$  do
15:    $T_{\alpha^s, \beta}^s = \{x_i^s \mid i = 1, \dots, N^s\}$ 
16:    $\alpha^s = \alpha^s - \eta \nabla_{\alpha^s} \mathcal{L}(T^d, T_{\alpha^s, \beta}^s)$ 
17: end while
18: Return  $\alpha^* \leftarrow \alpha^s$   $\triangleright$  Final optimized control parameters
19: Step 4: Control parameters transfer
20:  $\alpha^r \leftarrow \alpha^*$ 
    
```

VI. EXPERIMENTS AND RESULTS

We evaluated our approach using the CHASI robotic platform, a mobile robot with an Ackermann steering configuration (0.8 m $\times$ 0.6 m $\times$ 0.45 m). Simulations were conducted in the Stage simulator, a lightweight 2D tool that efficiently models sensor data and robot motion. Our navigation stack was fully integrated into Stage, enabling controlled and repeatable testing.

For real-world validation, we collected trajectory data

in a 14 m $\times$ 2 m test area and compared it with the simulated results. Simulation and control parameter tuning were optimized using Optuna, a widely used hyperparameter optimization framework.

A. Real-World Collected Data

For this experiment, we used the desired trajectory illustrated in Figure 4 (a), denoted as T^d . This trajectory consists of both straight-line segments and two sharp curves, designed to assess and optimize the robot's performance in both linear and curved path-following scenarios.

The actual trajectory followed by the robot, denoted as $T_{\alpha_0^r}^r$ using the initial default control parameters α_0^r , is also shown in Figure 4 (a). It can be observed that the robot successfully tracks the straight-line segments of the trajectory. However, when navigating the curved sections, the robot struggles to maintain accurate path tracking, exhibiting a noticeable error gap between the desired and actual trajectories.

B. Simulation calibration

This step aimed to align the simulator with the real robot's behavior. Figure 4 (b) shows the simulated trajectory $T_{\alpha_0^s, \beta_0}^s$ using default control parameters α_0^s and simulation parameters β_0^s . While the trajectory appears to follow T^d , a discrepancy with the real trajectory $T_{\alpha_0^r}^r$ reveals inaccuracies in the simulation.

To improve fidelity, we used the real robot's trajectory $T_{\alpha_0^r}^r$ as a reference, aiming to match its deviations while tracking T^d . Keeping the control parameters fixed at α_0^s , we optimized the simulation parameters β using Optuna. This adjustment resulted in a refined simulated trajectory, $T_{\alpha_0^s, \beta^*}^s$, which more closely aligned with the real-world trajectory. As shown in

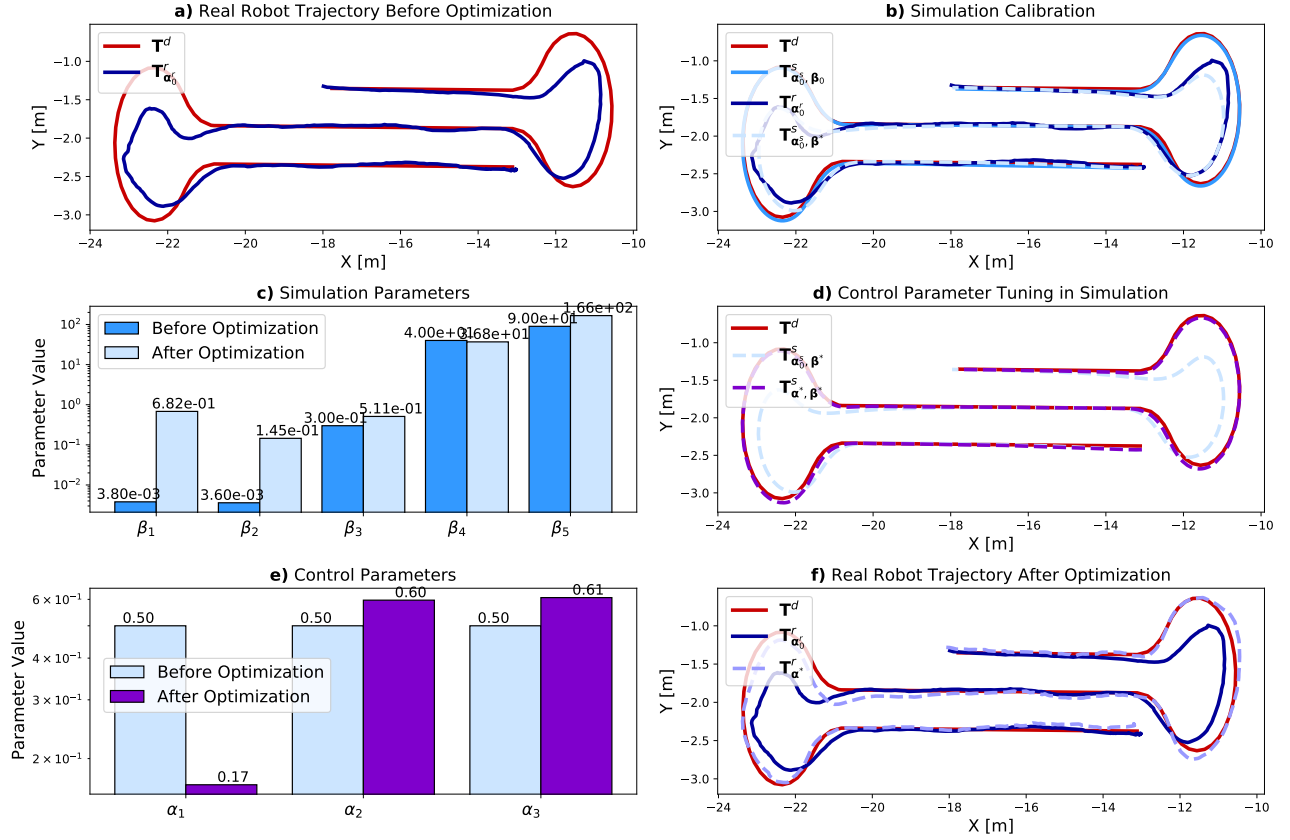


Fig. 4. (a) Robot performance before control parameter optimization. (b) Simulation inaccuracy before calibration and the improvement afterward. (c) Changes in simulation parameters after calibration. (d) Simulation improvement after tuning the control parameters. (e) Adjustments in control parameters after fine-tuning. (f) Final improvement in real-robot tracking using the fine-tuned control parameters.

Figure 4 (b), this calibrated trajectory better represents the real robot's performance.

Figure 4 (c) presents a comparison between the default initial simulation parameters β_0 and the optimized parameters β^* , highlighting the changes introduced through the calibration process. The most significant adjustments can be observed in the velocity delay and steering delay, suggesting that the real robot experiences inherent delays when executing both velocity and steering commands. Additionally, the maximum allowable acceleration was increased, while the maximum angular velocity was slightly reduced. Conversely, the steering angle acceleration was increased. These adjustments enable the simulated robot to better replicate the real-world robot's behavior, effectively capturing hardware-induced limitations and response delays.

C. Control Parameter Tuning

In this step, we fine-tuned the control parameters in simulation, α^s , while keeping fixed the optimized simulation parameters β^* from the previous stage. Since β^* already captures the real robot's characteristics, the goal was to adjust α^s to ensure the simulated robot closely follows the desired trajectory, T^d . After optimization, we obtained an improved trajectory, $T^s_{\alpha^*, \beta^*}$, which better aligns with the

reference trajectory. The resulting trajectories are shown in Figure 4 (d), while Figure 4 (e) compares the initial control parameters, α_0^s , with the optimized parameters, α^* .

D. Transfer Learning

In this step, the optimized control parameters are transferred to the real robot $\alpha^r \leftarrow \alpha^*$. The robot is then tested again in the same environment, following the initial reference trajectory to evaluate its performance.

Figure 4 (f) illustrates the desired trajectory T^d , the real robot's trajectory before optimization, $T^r_{\alpha_0^s}$, and the trajectory obtained after applying the optimized control parameters, $T^r_{\alpha^*}$. As observed, the optimized trajectory follows the desired path more closely. While the robot already performed well on straight-line segments, the most significant improvement is evident in tracking the curved sections, demonstrating the effectiveness of our method for this trajectory.

E. Achieved Improvements

This section presents the improvements in three key aspects: tracking error, time savings, and energy consumption. The results are summarized in Table III, with the corresponding calculations detailed in Appendix VIII. These estimations are based on approximate data.

The table shows that our method achieved a 51.08% reduction in tracking error, saved approximately 3.84 hours of execution time, and reduced energy consumption by 9.175 kWh.

TABLE III
PERFORMANCE AND RESOURCE CONSUMPTION IMPROVEMENTS

Metric	Original Gains	Optimal Gains	Improved
Tracking Error (2D-DTW)	3.72	1.82	51.08%
	Estimated Real System Consumption	Optimized Method Consumption	Reduction
Time (hours)	9.34	5.50	3.84
Energy Consumption (kWh)	9.34	0.165	9.175

VII. CONCLUSION

We presented a simulation-driven framework to optimize the Stanley controller for autonomous navigation. The approach follows a two-stage process: first calibrating simulation parameters to reflect real-world dynamics, then optimizing controller gains within the calibrated simulator. This method reduces real-world experimentation, lowers mechanical wear, and improves trajectory tracking. The resulting control parameters transferred effectively to the physical robot, validating the framework's reliability.

Future work will explore broader trajectory variations to derive more generalized gains and incorporate sensor noise modeling—particularly from laser scans—as additional tuning parameters, given their significant impact on navigation in dynamic environments.

VIII. APPENDIX

A. Tracking Error Improvement Calculation

The tracking error improvement is computed using the following equation:

$$\text{Improvement\%} = \frac{e_{ig} - e_{og}}{e_{ig}} \times 100 \quad (12)$$

where $e_{ig} = 3.72$ represents the tracking error with the initial gains, whereas $e_{og} = 1.82$ represents the tracking error with the optimized gains obtained using our method. This formulation quantifies the relative improvement achieved through optimization.

B. Time Savings Calculation

To estimate the time required for real-system optimization, we define:

- t_a = time to complete one trajectory (170 sec). This value has been obtained directly from $\mathbf{T}_{\alpha_0}^r$ during the data collection.
- σ_a = repositioning time before a new trial (120 sec). This time was obtained experimentally in past experiments.
- N_o = total trials required for the control parameter optimization (116)

The total estimated time required on the real system is:

$$\text{Estimated Real System Time} = \frac{(t_a + \sigma_a) \cdot N_o}{3600}$$

Using our method, the time per simulation trial was $t_s = 45$ sec, with $N_s = 324$ trials needed for simulation calibration. The total time spent in simulation is:

$$\text{Estimated Simulation Time} = \frac{(t_s \cdot N_o) + (t_s \cdot N_s)}{3600}$$

C. Energy Consumption Calculation

To estimate the energy consumption required for tuning the control parameters on the real robot, we use $p_r \cdot t_{op}$, where $p_r = 1kW$ is the robot's power consumption, obtained from technical datasheet; $t_{op} = 9.34h$ is the estimated time required to complete the tuning process on the real robot.

To compute the energy consumption using our method, we consider a standard laptop's power consumption of $p_c = 0.030kW$. Given that the total simulation time is $t_s = 5.5h$, the energy consumption is calculated as $p_c \cdot t_s$

REFERENCES

- [1] B. Acosta, W. Yang, and M. Posa, "Validating robotics simulators on real world impacts," *CoRR*, vol. abs/2110.00541, 2021. [Online]. Available: <https://arxiv.org/abs/2110.00541>
- [2] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," 2019.
- [3] T. D. Gillespie, *Fundamentals of Vehicle Dynamics*. SAE International, 1992.
- [4] K. Kang, S. Belkhal, G. Kahn, P. Abbeel, and S. Levine, "Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight," *CoRR*, vol. abs/1902.03701, 2019. [Online]. Available: <http://arxiv.org/abs/1902.03701>
- [5] J. U. Kreber, J. Schlechtriemen, J. Boedecker, and W. Burgard, "Learning a terrain- and robot-aware dynamics model for autonomous mobile robot navigation," *Robotics and Autonomous Systems*, 2024, <https://arxiv.org/html/2409.11452v1>.
- [6] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.
- [7] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3803–3810.
- [8] F. Ramos, R. C. Possas, and D. Fox, "Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators," 2019. [Online]. Available: <https://arxiv.org/abs/1906.01728>
- [9] E. O. Sodiya, U. J. Umoga, O. O. Amoo, and A. Atadoga, "Ai-driven warehouse automation: A comprehensive review of systems," *GSC Advanced Research and Reviews*, vol. 18, no. 02, pp. 272–282, 2024. [Online]. Available: <https://gsconlinepress.com/journals/gscarr/sites/default/files/GSCARR-2024-0063.pdf>
- [10] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," 2018. [Online]. Available: <https://arxiv.org/abs/1804.10332>
- [11] S. Thrun, M. Montemerlo, H. Dahlkamp, *et al.*, "Stanley: The robot that won the darpa grand challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.
- [12] G. Tiboni, K. Arndt, and V. Kyrki, "Dropo: Sim-to-real transfer with offline domain randomization," *Robotics and Autonomous Systems*, p. 104432, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0921889023000714>
- [13] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," 2017. [Online]. Available: <https://arxiv.org/abs/1703.06907>
- [14] Z. Wang, X. Li, J. Yang, Y. Liu, and S. Jiang, "Sim-to-real transfer via 3d feature fields for vision-and-language navigation," 2024. [Online]. Available: <https://arxiv.org/abs/2406.09798>

Investigating 2.5D path-planning methods for autonomous mobile robots in complex unstructured off-road scenarios*

Andre Koczka¹, and Gerald Steinbauer-Wagner¹

Abstract—Most of the existing literature focuses on path planning in 2D, where the 3D world is converted to a 2D grid map. There is little literature on methods that can natively utilize 2.5D or 3D information and thus use a less compressed representation of the environment for planning. In this work, methods from both groups were systematically compared. A suitable simulator and physics engine have been chosen to enable a realistic evaluation of 2.5D navigation in a simulation. For the methods using the 2D view, classical and widely used planning algorithms were used. To generate the map for the classical methods, a 2.5D map was converted into a 2D map using slope information. The classical search algorithms find a path based on costs on the 2D map. To test a method that uses native 2.5D data for planning, a novel approach was developed that uses the robot's orientations on a 2.5D elevation map. This method samples different locations on the 2.5D map and considers the attitude of the footprint for each position to generate the cost. The evaluation showed that the proposed method, which uses 2.5D data directly, planned shorter and faster paths in most scenarios, while the journey remained safe and reliable for the robot. The results for the classical, 2D methods showed that they are especially useful in scenarios where low computational power is available.

Index Terms—Path planning, Autonomous robots, rapidly-exploring random tree

I. INTRODUCTION

Path planning for autonomous ground vehicles is usually done on 2D costmap [6] using a search algorithm. The most popular way of using a costmap is projecting objects detected in data from a sensor, like LIDAR or camera, to a cell on the map, where each position is represented either to be occupied or free. More advanced solutions take advantage of the value-range of a 2D costmap in the Robot Operating System (ROS) [11] and map a probabilistic value to each cell, which results in a gradient of values instead of binary, lethal, and non-lethal costs. Both approaches compress the information gathered by a 3D LIDAR or depth camera while losing information on the terrain. While this approach can work well if tuned correctly, it often leads to longer paths, higher energy consumption, and potentially missed opportunities due to the conservative representation of the robot-terrain interaction in 2D. To combat this issue, state-of-the-art solutions use a 2.5D representation of the environment to increase the available information on the map and plan more intelligently by being aware of the actual terrain surface. Planning in 2D also makes it harder to take the physical

properties of the robot, such as the maximum tilt angle, into account while planning on off-road areas.

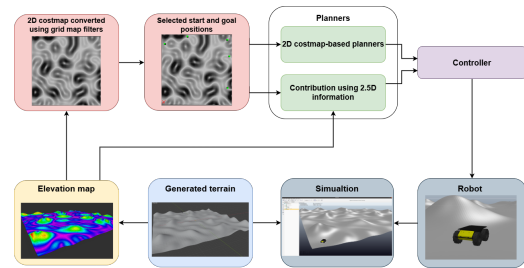


Fig. 1. Flowchart showing the implemented pipeline for testing and evaluation.

The aim of this paper is to compare planning methods in kinematically challenging, accurately simulated off-road environments to gain a better understanding of the properties and limits of path planning algorithms for 2.5D environmental representations. As 2.5D perception goes beyond the scope of this work, it is assumed to be perfect. For the simulation, the widely used robot Husky from Clearpath Robotics [2] will be used, which is a rigid-body off-road differential-drive robot. There are also a lot of resources available to be able to set up a realistic simulation for the Husky.

The contribution of this work can be summarized as follows:

- implementation of a method that compresses 2.5D data to a 2D costmap with minimized loss of information, that can be used by standard search algorithms.
- a method using 2.5D information natively to estimate traversability for complex terrains.
- a simulation pipeline with realistic physics and terrain generation to ensure close-to-life results.
- an evaluation pipeline that is modular, and can be easily adapted to other path planning techniques in the future for further research.

II. RESEARCH

For planning a path, in general, the robotics community has a broad range of well-established methods. Advances in formulating and solving search and optimization problems have finally enabled advanced research in path planning in higher dimensions for drones and off-road vehicles, which would have been impossible previously. Most solutions rely on a well-tuned 2D obstacle map where lethal and non-lethal obstacles are defined to cover the worst-case scenarios and

*This work was funded by the Austrian defense research program FORTE of the Federal Ministry of Finance (BMF) under the project PATH.

¹{akoczka, gerald.steinbauer-wagner}@tugraz.at, Institute of Software Engineering and Artificial Intelligence, Graz University of Technology, Graz, Austria.

allow for safe traversal. These solutions are computationally efficient, however, they lack a full understanding of complex environments. However, novel methods exploit 2.5D data natively to allow for more intelligent path planning.

The ArtPlanner [17] by Fankhauser et al. from ANYbotics [1] is one of the few planning pipelines using native 2.5D information for planning. Their approach requires knowledge about the robot's shape and kinematic abilities. They utilize a sampling-based approach with LazyPRM* at its core, which checks the feasibility of a pose at any given sample point. Their approach uses the whole body of the robot to check for collisions and kinematically feasible positions using an elevation map.

Traversability-Based RRT* [16] by Takemura et al. was originally developed for a planetary rover model. The authors approach the problem without a mapping algorithm, planning the path directly on the perceived pointcloud. They use RRT* to sample the points and project the robot's footprint to the terrain, which they use to calculate an orientation-dependent cost. The quality of the path in this case is highly dependent on the LIDAR's update frequency and point density.

"Risk-Aware Mapping and Planning"(RAMP) [14] from Sharma et al. uses a compressed 2D representation of the environment. The authors of the paper approach the problem from the controller's perspective, improving known tools and algorithms to better understand traversability of known and unknown terrain. The problem the authors describe is the lack of awareness of known and unknown spaces in the control phase. The authors use elevation mapping and compress the 2.5D representation into a 2D costmap to make it more efficient and use an improved path planning approach on the 2D representation to solve the described problem.

In the paper "STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-road Navigation" [8] from Fan et al. a complete navigation pipeline is presented. The pipeline aims to solve planning challenges in extreme off-road situations. The authors approach the problem similarly to RAMP by using elevation mapping for the global planning problem and converting the 2.5D representation into a 2D traversability map with a custom cost function. They, however, couple this approach with the control and path following problem as well, including the full 2.5D representation of the environment in the control problem, by calculating a kinematics-based cost for traversal. They approach this similarly to [16] from Takemura et al. , by using the orientation of the robot on a 2.5D map.

III. EVALUATION DESIGN

As presented in the previous section, state-of-the-art works in off-road planning take one of two approaches:

- pipelines, that use a combination of mapping techniques and costmap calculations, but plan using standard search algorithms on a 2D costmap
- methods which extend their cost calculations with native 2.5D data, using the kinematic properties of the robot

In order to evaluate if the two approaches are suitable for path planning in challenging off-road environments instances

of each approach are prepared and systematically evaluated in a standardized, simulated setting. The first part of the implementation and tests, corresponding to the first bullet point in the above list, was influenced by the works STEP [8], and RAMP [14], using a selection of search algorithms based on the suggestions of the paper [10]. The second part of the evaluation, expanding an algorithm to use 2.5D data natively has been inspired by the techniques used by the ArtPlanner [17] and Traversability-based RRT* [16]. This method has been implemented from scratch, as no openly available solution was available to test at the time of writing. The two distinct approaches are implemented in ROS and simulated using CoppeliaSim [12], which supports both ROS1 and ROS2. We perform the evaluation in simulation first as extensive tests with hardware and real environments are not feasible. It has been determined in previous work, that the best physics engine for simulating rolling and bouncing (both of which are important properties for the Husky robot) is Bullet. This is important for the realistic simulation of robot-terrain interactions. This is also supported by the benchmark of Kang et al. in [5] and Farley et al. in [9]. The work of Farley et al. [9] includes the model of the Husky robot and a corresponding Lua script for CoppeliaSim which will be used as a basis and extended to extract more data for the evaluation.

A. Evaluation metrics

The most problematic property of off-road traversal is the fact that uneven terrain easily pushes the hardware to its limits. This means that finding routes that don't exhaust the robot's capabilities to a dangerous level while keeping the path reasonably short and quickly traversable, while consuming as little energy as possible, are the most important goals of such a path planner. Thus, we developed a set of metrics for our evaluation of path planning algorithms for off road environments:

- **standard deviations of roll and pitch** are expected to be lower for the 2D-based algorithms, as these will try to minimize the cost on the slope-based costmap.
- **the length of the planned path** is estimated from 2D coordinates returned by the algorithms.
- **traveled path length calculated using the odometry** is expected to deviate from the generated path due to slippage and sharp turns.
- **required energy** to complete a path gives a good indication of planning quality. Paths produced by the two methods are expected to deviate in energy consumption.
- **average and standard deviation of torque of the robot joints** indicate the average of the momentary efforts the robot had to make along the path. This metric is expected to be higher for more aggressive plans.
- **travel time from sending the goal to reaching the goal** is expected to be correlated with path length, however, it might deviate due to slippage if a low quality path is provided which is harder to travel along.
- **planning time** is the time from issuing the goal to receiving a path from the planner.

- **successful traversal** of the generated path gives an indication if the planner has provided a traversable path. This metric uses a timeout of 6 minutes for reaching the goal.

IV. METHODOLOGY AND IMPLEMENTATION

In order to evaluate the quality of a path provided by the path planners, a complete system of perception, planning, and execution is needed, as the robot will be executing the plans in the simulation system. The building blocks of this system are described in this section.

A. 2.5D grid map and conversion to 2D

The first method, which follows the style of papers [8] and [14], implements a conversion mechanism, that filters the 2.5D grid map and converts it into a 2D costmap while maintaining terrain awareness. By doing this, we ensure, that the initial source of information (the elevation map) is the same for both methods, while retaining the maximum amount of information in 2D. The conversion is done using a slope filter, which converts slope data into cost values on the 2D costmap. This method works by applying a mathematical filter to the elevation map, which calculates the normal vector of every cell on the map, and takes the arc-cosine of these normal vectors. This results in a map of slope values in radians. It is known from the datasheet of the Husky, that its maximum claimable slope is 30 degrees. It is known from initial testing, that the realistically climbable slope on a smooth, rocky surface is around 25 degrees. By adjusting the maximum degree of slope to be 25 degrees, any slope that's over this value will be clipped to the maximum lethal value on the map. This results in a gradient of slope values between 0 and 25 degrees, which is a good representation of where the robot can safely traverse. The cost of a cell on the map thus looks as follows:

$$C(x,y) = \min(255 \cdot \frac{\alpha(x,y)}{27}) \quad (1)$$

where α represents the calculated slope value in degrees at given map coordinates. The algorithms that will be used for standard planning on a 2D map are **A***, **Theta***, **D***, and **RRT*** which were selected based on the decision tree shown in [10] by Gargano et al. Even though **A*** would be sufficient to test as a graph-based search algorithm, due to its wide usage and popularity, its newer iterations, **D*** and **Theta*** have also been investigated to see if they have any disadvantage for future work. Thus, the algorithms **Theta***, and **D*** are expected to perform similarly or even the same as **A***, because the underlying heuristics guarantee an optimal solution for each of these algorithms. **Theta*** might perform worse than **A***, due to it connecting line-of-sight nodes without considering the costs between them. **RRT*** is considered for its speed, efficiency, and asymptotic optimality. This is also the algorithm, which will be extended to use 2.5D information directly. The cost function of each algorithm, including heuristics, looks as follows:

$$C(n) = g(n) + h(n) + \gamma(n) \quad (2)$$

where $\gamma(n)$ represents the cost of the cell at position n shown in Equation 1. $C(n)$ is then the estimated cost of the path from the start node until the goal, taking node n . The cost $g(n)$ is the already accumulated cost until the last node before expansion, and $h(n)$ is the heuristic cost, which in the case of **A*** is the Euclidean distance to the goal.

B. Extended RRT*

The second method uses the height and terrain data of the 2.5D grid map natively during planning, similar to the works in [17], [16] and [8] by using the robot's footprint to determine the attitude. The most important factor when planning on rough terrain is the locomotion capabilities of the robot. This is strictly bounded by its kinematic limits, like tipping angle, and max climbing angle. Following the idea of [16] **RRT*** is extended with an additional function, which projects the robot's footprint onto the 2.5D data and determines a cost from its attitude. Instead of doing this on a pointcloud as shown in the paper [16], it is done on a 2.5D grid map. This approach has the advantage, that a uniformly distributed map is used, regardless of point density, and independent from the capabilities of the sensor. The choice of **RRT*** ensures that given enough time and samples, the solution should converge to an optimal path. It is however planned for future work to also test other sampling-based methods as a basis for this contribution, like **LazyPRM***, that is also used by the **ArtPlanner** [17]. By projecting the robot's footprint onto the grid at any given point, the representation of the robot's position and attitude can be described with the translation and attitude of its footprint in space. The projected plane (called pseudo-plane in [16]) has a normal vector on the surface of the plane at an arbitrary sampling point, which at any given time is solely dependent on the plane's position relative to the grid map. From this normal vector, the roll and pitch values can be extracted and used in a cost function. The cost function is a weighted sum of the values, similarly to the method shown the paper [16]. The yaw or rotation of the currently sampled point on the plane is dependent on the previous (parent) node in case of **RRT***, and is calculated by taking the angle between the parent and currently sampled node in reference to the map's frame. The four wheels of the robot form a rectangle. Projecting a rigid rectangle onto a height map is not a trivial problem. If two opposing corners of a rectangle are at different heights, it leads to a diagonal split in the middle, which separates the rectangle into 2 triangles, which leads to two possible normal vectors, one for each half, without any obvious method to choose one. To simplify this problem, an assumption is made: the two rear wheels of the robot will only be considered at one point, in the middle, forming a triangle of the robot. While this isn't ideal, it is still a good representation of the possible orientation of a rigid-body robot when projected onto a plane. The footprint is illustrated for the Husky robot in Figure 2 on the left. Now, projecting these three points onto a height map will always result in a triangle with an even surface. More importantly, the normal vector can be easily calculated now, by taking the cross product of two

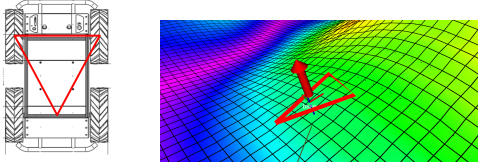


Fig. 2. The triangle formed between the front two wheels and the average of the rear wheels illustrated on the Husky robot on the left, and the normal vector along with the projected triangle footprint visualized in RViz on the 2.5D grid map on the right.

sides of this triangle.

An obvious issue with this method is the fact that if a small, but lethal obstacle falls into the area of the triangle, it won't be considered by the algorithm at all. This wasn't an issue for the tests conducted in work, but this case must be handled in the future for real-life tests. Taking the cross product of two of the edges of the projected triangle, and normalizing it results in the normal vector of the plane. To calculate the pitch and roll of the normal vector, the following function is used:

$$g(k) = \begin{bmatrix} \phi_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} \text{atan2}(n_x, n_z) \\ -\text{atan2}(n_y, n_z) \end{bmatrix} \quad (3)$$

where n_x , n_y and n_z represent the components of the calculated normal vector. The resulting normal vector is visualized in RViz in Figure 2 on the right.

Pitch and roll can then be used to form the following cost function:

$$C(k) = (W_\phi \frac{|\phi_k|}{N_\phi} + W_\theta \frac{|\theta_k|}{N_\theta}) W_s \quad (4)$$

ϕ represents pitch, and θ represents the roll values calculated in the previous step. For simplicity, in this proof-of-concept implementation, only the absolute value of these are taken, however, the option to use the sign of these variables is left for future work, as it might be useful to penalize going up or downhill. The factors N_ϕ and N_θ are normalization values, which have been defined empirically to be 100 and serve the purpose of scaling the values, such that they aren't dominant in comparison to the distance cost. Furthermore, the weight factors W_ϕ and W_θ have been defined to be able to adjust the influence of each factor independently, while W_s adjusts the influence of the entire expression. The weight values have been set up to be adjustable using a feature of ROS called "dynamic reconfigure" [4]. Additionally to the cost-calculation, an initial feasibility-check is done to ensure that the roll and pitch values are within the maximum range, and the sample is thrown away if it isn't the case.

C. Terrain generation and simulation setup

The ground truth terrains are generated using Blender with its built-in geometry node plugin using ridged-multifractal noise [3] which uses Perlin noise[7] at it's basis. The point cloud for elevation mapping is also exported during this process with high density. The main limitations of map size are the file size for storage and the processing power

needed to use them. The maps used are as large as it was practically possible, with 50m x 50m. The tested algorithms are assumed to work seamlessly on a real setup, using a much smaller, rolling map, if they are able to achieve good enough performance on such a large map. For this work, a total of 5 random terrains have been generated randomly to represent different terrain difficulties. Moreover, a 6th one was created manually to visualize the benefit of the 2.5D planning method. This will be shown in Section V. The terrains are then imported manually into CoppeliaSim. The starting position of the robot has been chosen to be a random corner of each map. To increase the number of planning problems, the opposite corner is used on each map as a second starting position. This effectively leads to 10 different terrains from the perspective of the planning algorithm. For each terrain, 5 goal positions are distributed on the map. These goal positions have been chosen empirically, by driving the robot manually and making sure that the positions are actually reachable. In future work, this process shall also be automated to be able to test an arbitrary number of terrains and goal positions. 5 terrains with 2 starting positions and 5 goals each lead to 50 unique planning problems for each of the tested algorithms. An example terrain with marked start and goal positions is shown as a 2D costmap in Figure 3. To

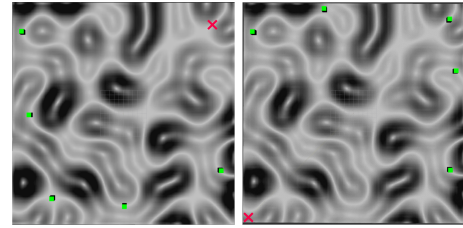


Fig. 3. Example goal positions on a generated terrain. The robot's initial positions are indicated by a red X in the bottom left corner and in the upper right corner.

evaluate energy consumption, the Lua script in CoppeliaSim has been extended to progressively calculate and publish the cumulative energy used by the virtual motors of the Husky robot to a ROS topic.

D. Evaluation

For testing, a Python script has been developed which automates the process, loads the simulation, and saves all the recorded data. For every algorithm, the program iterates through generated terrains, and for each terrain, the corresponding navigation tasks (goal positions) are executed. During execution, the script receives data from ROS messages continuously. All data is saved into .csv data frames and array files, which are evaluated later using another automated script. Saving all raw data also makes it possible to evaluate any run using other criteria in the future. The flow of the automation script is shown in Figure 4. For path execution, a basic version of Timed-Elastic-Bands (TEB) [13] has been used as a controller (without object avoidance), and tuned empirically to follow the generated paths as strictly as possible, with as little influence as possible.



Fig. 4. The flowchart showing the process of the evaluation, executed by the test script. The script is started manually and iterates through a list of registered algorithms, terrains and goal positions, which can be of arbitrary length.

V. RESULTS

The results will show averaged metrics over all planning problems. The final results have only been calculated for runs that all algorithms completed successfully so that the comparison stays fair (union of all data where execution succeeded). A successful execution was defined by reaching the goal within 6 minutes after generating a path.

First, the number of failed executions will be shown (see Figure 5). Please note that for RRT* and the proposed, extended RRT*, which will be called "RRT* Kin." (kinematic) in the plots, the worst-case number of failures of 5 full sets of navigation tasks is given. The most frequent cause of a failed

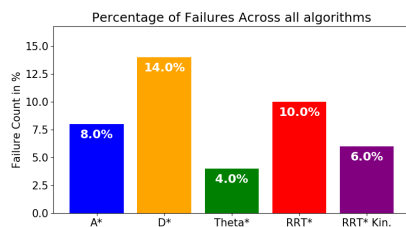


Fig. 5. The percentage of total failures across algorithms. RRT* and RRT* kinematic shows the worst case failures of all the runs.

execution was turning on slopes, where a differential drive robot struggles the most, as it loses friction while also having a shifted weight distribution. The proposed method has only failed due to this issue. Other algorithms have frequent cases of getting stuck in tight corridors or valleys due to the lack of terrain awareness.

Perhaps the most important finding of all is the average effort in watt-hours (see Figure 6). It was expected that the proposed extended method, RRT* kinematic, would produce paths that consume more energy because it takes more aggressive paths. However, due to the fact that on average it produced 5% shorter paths than the next best algorithm while having a low variance in yaw (less turning) made it the best regarding power consumption in the tested scenarios.

By taking shortcuts at safe places, which are within the robot's kinematic capabilities, the proposed method was less susceptible to turning on slopes and also made less sharp turns leading to an overall shorter path. This behavior is also confirmed by the results shown in Figure 7, which shows the cumulative height difference.

Looking at Figure 7 it can be concluded, that the higher cumulative height difference did not have a negative influence on travel times, making the proposed method the fastest by a small margin. It also shows that travel times are more

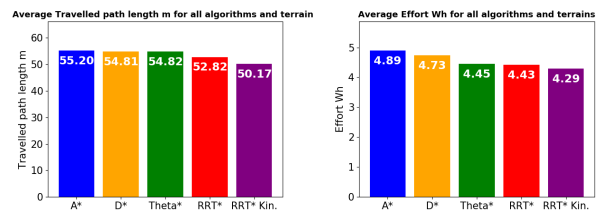


Fig. 6. The averaged traveled path lengths in meter on the left, and the averaged energy needed (effort) in Wh across all algorithms, terrains, and paths.

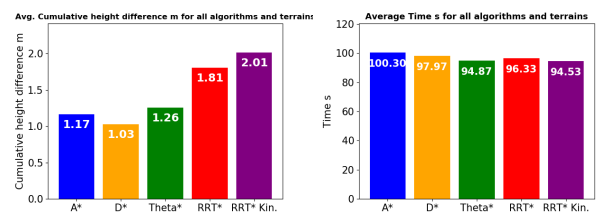


Fig. 7. The cumulative height difference of traveled paths on the left and the mean of travel time on the right, averaged over all paths and terrains.

correlated with the path length than with the cumulative height, which means that all algorithms managed to find more-or-less easily traversable trajectories for the robot.

Looking at the mean standard deviation of roll and pitch shown in Figure 8 together with traversal time in Figure 7 suggests that taking a smoother path only has a small negative impact on travel time. Thus, in some cases it might be more feasible to take a flatter path. It was expected that the standard 2D costmap-based algorithms would produce a path with fewer variations as they only use a limited amount of information, and prefer lower-cost cells.

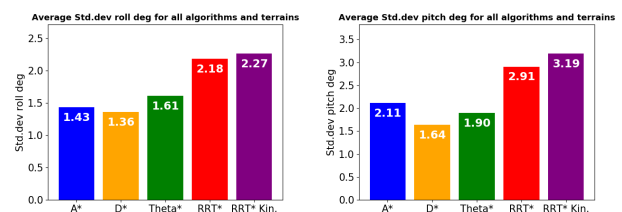


Fig. 8. The averaged standard deviation roll and pitch across all algorithms, terrains, and paths.

Unexpectedly, the vanilla RRT* algorithm produced a high cumulative height difference, while also traveling longer.

This could be explained by the fact that it produces similar paths to Theta*, with lots of straight sections, but having much more of these sections, leading to more unwanted zig-zag turns. Figure 6 showed that the proposed method actually consumed about 3% less energy than the next best algorithm. Taking the standard deviation of yaw, pitch, and roll, and the cumulative height into account, it can be concluded that it's most of the time more efficient to go above a hill than to go around it if it can be guaranteed that the robot can operate within its limits.

Until now, the results are promising, especially due to the fact that the proof-of-concept method already performs well in the initial version. Unfortunately, a significant disadvantage of the proposed method is planning time. On average, the proposed solution returns a path in 7630ms for the given map size, which is very high in comparison to the 30-300ms path-return times of the classical 2D search algorithms. However, looking at the unmodified RRT*, the planning time only decreases by about 900ms to 6725ms. This shows that the added cost calculations using the robot's footprint have a relatively low impact on the planning time. Thus, the implementation has been determined to be inefficient, and as a next step, the solution will be re-implemented using the popular Open Motion Planning (OMPL) [15] library to improve speed.

In order to visualize the advantage of the proposed method even better, a map with a single ramp with a slope of 29 degrees has been made. This is above the set worst-case cost factor of the classical algorithms, and thus all of them fail to provide a path to the goal. However, the proposed solution can find a skewed path that stays within the bounds of the robot, as the slope is less steep from a skewed perspective. This is shown in Figure 9.

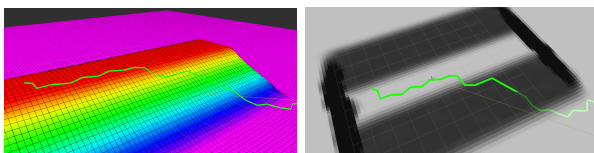


Fig. 9. Planning skewed to the slope makes the slope less steep.

VI. SUMMARY AND OUTLOOK

The evaluations provided us with valuable information regarding planning on uneven terrain. It has been shown that using 2.5D information natively is beneficial for planning in off-road scenarios by evaluating different metrics recorded during testing. It has also been concluded, that the proposed approach is less computationally efficient than methods using standard search-based algorithms on a 2D costmap. Thus, there is still work to be done for the developed 2.5D-based planner to be usable in real-time with a real robot. For the 2D approach one of the most basic and widely used methods has been used. It compresses the terrain data into a slope map, which results in the least amount of loss of information about the environment. In reality, the slope map could be

extended by additional traversability costs to also account for different surface types and roughness levels. Meanwhile, the proposed method using 2.5D data directly has great potential of combining the cost calculation on a 2.5D surface with other types of map layers to make planning more efficient. As the proposed method is only a proof-of-concept solution, it still requires more work to achieve a better quality outcome. Path executions failed in most cases due to turning on slopes. In future work, a converted slope map could be taken into account by the controller to restrict turning on steep slopes, similar to the paper [14], which would mitigate this issue.

REFERENCES

- [1] "ANYbotics — Autonomous Legged Robots for Industrial Inspection — anybotics.com," <https://www.anybotics.com/>, [Accessed 23-12-2024].
- [2] "Clearpath Robotics: Mobile Robots for Research & Development — clearpathrobotics.com," <https://clearpathrobotics.com/>, [Accessed 02-01-2025].
- [3] Noise Texture Node - Blender 4.3 Manual — docs.blender.org, https://docs.blender.org/manual/en/latest/render/shader_nodes/textures/noise.html, [Accessed 05-01-2025].
- [4] M. C. Blaise Gassend, "dynamic reconfigure - ROS Wiki — wiki.ros.org," http://wiki.ros.org/dynamic_reconfigure, [Accessed 11-01-2025].
- [5] J. H. Dongho Kang, "SimBenchmark — leggedrobotics.github.io," <https://leggedrobotics.github.io/SimBenchmark/>, [Accessed 30-12-2024].
- [6] D. H. Eitan Marder-Eppstein, David V. Lu, "costmap_2d - package summary," http://wiki.ros.org/action/info/costmap_2d?action=info, 2018, [Accessed 22-12-2024].
- [7] T. R. Etherington, "Perlin noise as a hierarchical neutral landscape model," *Web Ecol.*, 22, 1–6, 2022.
- [8] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A.-A. Agha-Mohammadi, "Step: Stochastic traversability evaluation and planning for risk-aware off-road navigation," 2021. [Online]. Available: <https://arxiv.org/abs/2103.02828>
- [9] A. Farley, J. Wang, and J. A. Marshall, "How to pick a mobile robot simulator: A quantitative comparison of coppeliasim, gazebo, morse and webots with a focus on accuracy of motion," *Simulation Modelling Practice and Theory*, vol. 120, p. 102629, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1569190X22001046>
- [10] I. E. Gargano, K. D. von Ellenrieder, and M. Vivolo, "A survey of trajectory planning algorithms for off-road uncrewed ground vehicles," in *Modelling and Simulation for Autonomous Systems*, J. Mazal, A. Fagiolini, P. Vasik, F. Pacillo, A. Bruzzone, S. Pickl, and P. Stodola, Eds. Cham: Springer Nature Switzerland, 2025, pp. 120–148.
- [11] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," vol. 3, 01 2009.
- [12] E. Rohmer, S. P. N. Singh, and M. Freese, "Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework," in *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013, "www.coppeliarobotics.com".
- [13] C. Rösmann, F. Hoffmann, and T. Bertram, "Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control," in *2015 European Control Conference (ECC)*, 2015, pp. 3352–3357.
- [14] L. Sharma, M. Everett, D. Lee, X. Cai, P. Osteen, and J. P. How, "Ramp: A risk-aware mapping and planning pipeline for fast off-road ground robot navigation," 2023. [Online]. Available: <https://arxiv.org/abs/2210.06605>
- [15] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, <https://ompl.kavrakilab.org>.
- [16] R. Takemura and G. Ishigami, "Traversability-based rrt* for planetary rover path planning in rough terrain with lidar point cloud data," *Journal of Robotics and Mechatronics*, vol. 29, pp. 838–846, 10 2017.
- [17] L. Wellhausen and M. Hutter, "Artplanner: Robust legged robot navigation in the field," *Field Robotics*, vol. 3, no. 1, p. 413–434, Jan. 2023. [Online]. Available: <http://dx.doi.org/10.55417/fr.2023013>

LiDAR-Based Ground Segmentation with Structured Point Clouds for Multi-Sensor AMRs*

Hamid Didari¹ and Gerald Steinbauer-Wagner¹

Abstract—LiDAR-based perception is a popular component of autonomous mobile robots (AMRs) for obstacle avoidance and traversable area detection. Traditional ground segmentation approaches, such as ring-based methods, often assume a fixed sensor placement and may struggle in multi-LiDAR or tilted sensor configurations. To overcome these limitations, we propose a novel segmentation approach based on the organized point cloud representation, which preserves the spatial arrangement of LiDAR data in a structured 2D format. Our method first organizes the raw point cloud into a structured array, ensuring direct neighborhood accessibility without additional spatial searches. We then use a rolling window over the array to estimate surface normal vectors. Ground segmentation is performed iteratively by classifying normal vectors based on orientation and height consistency. A likelihood approach is further utilized to segment points by assigning them to their corresponding normal vectors. Furthermore, we evaluate our method through experimental tests on a real-world multi-LiDAR AMR in five different scenarios within unstructured environments, achieving an average accuracy of 0.939.

Index Terms—Mobile Robots, Scene Understanding, Off-Road Navigation

I. INTRODUCTION

The deployment of autonomous mobile robots (AMRs) in logistics has become increasingly prevalent due to their potential to enhance productivity and reduce costs. Research by Keith and La [10] highlights that AMRs improve efficiency by minimizing manual labor in repetitive tasks, leading to lower operational costs and increased throughput. Similarly, a multiple case study by Grover et al. [7] identifies AMRs as key enablers of digital transformation in Industry 4.0 warehouses, where they contribute to cost reduction through efficient material handling and workflow optimization. Economic analyses further indicate that AMRs can lead to substantial long-term savings. A study by Zhang et al. [11] evaluates the return on investment (ROI) of AMR deployment, showing that companies recover their initial investment due to reduced labor expenses and increased productivity.

Despite their benefits, widespread AMR adoption in dynamic and unstructured environments faces several challenges, with local perception being one of the most critical. AMRs rely on LiDAR, cameras, and radar to perceive their surroundings, but sensor noise, occlusions, and environmental variations pose challenges. Among various perception

technologies, LiDAR-based perception is particularly effective in enabling AMRs to navigate complex environments by accurately detecting obstacles and identifying traversable areas. By generating high-resolution 3D point clouds, LiDAR sensors provide a precise spatial representation of the surroundings, allowing robots to differentiate between safe paths and potential hazards. This capability is critical for obstacle avoidance and traversable area detection, especially in outdoor and dynamic environments where other sensors may struggle due to lighting variations.

Ground segmentation is a fundamental task in LiDAR-based perception, facilitating accurate obstacle detection and navigation. Traditional methods often employ geometric approaches, such as plane fitting and elevation thresholding, to distinguish ground from non-ground points. However, these methods may struggle with complex terrains and require manual parameter tuning. To overcome these challenges, modern approaches integrate probabilistic models and machine learning techniques. Markov Random Fields (MRF) has been used to model spatial relationships between points, improving segmentation accuracy in uneven terrains [15]. Additionally, deep learning-based methods, such as Convolutional Neural Networks (CNNs), can learn complex patterns in point cloud data, enabling robust ground segmentation in diverse environments [14].

A more recent and efficient approach is ring-based ground segmentation, such as Patchwork, which leverages the structure of LiDAR point clouds to classify ground and obstacles [12]. Patchwork progressively segments the ground from near to far distances using LiDAR's ring structure, improving computational efficiency and robustness in unstructured outdoor terrains. However, a major limitation of Patchwork and many deep learning-based segmentation methods is the assumption that the LiDAR sensor is mounted horizontally at the robot's center. In reality, AMRs may use multiple LiDARs positioned at different angles or orientations—such as tilted or vertically mounted sensors—to enhance 3D coverage.

To overcome this limitation, we developed a segmentation approach based on the organized point cloud representation instead of partitioning space into rings. An organized point cloud is a structured representation where LiDAR points are stored in a 2D array format, preserving their spatial arrangement as captured by the sensor. This contrasts with an unstructured point cloud, where points are stored in a random order without inherent neighborhood relationships. The advantage of an organized point cloud is that each point's neighboring points are directly accessible using fixed

¹Hamid Didari and Gerald Steinbauer-Wagner are with the Institute of Software Technology, Graz University of Technology, Graz, Austria. {hamid.didari, steinbauer}@ist.tugraz.at

*This work was funded by the Austrian Research Funding Association (FFG) under the scope of the THINK.WOOD.INNOVATION program.

indices.

By leveraging this structure, our method ensures that partitioning is independent of the LiDAR setup, making it more adaptable to different sensor configurations. We compute the normal vector for each partition and fit a likelihood model to the points within it. Since the robot is assumed to be on a traversable surface, partitions with a zero mean height are classified as ground. From these initial ground partitions, we iteratively expand to neighboring partitions that are unlabeled. A partition is labeled as ground if its surface inclination angle is below a predefined threshold and its height difference from a known ground partition is within acceptable limits.

Furthermore, we use the likelihood model of each normal vector to classify points in the point cloud. This structured approach allows us to determine which normal vector a given point belongs to, even when neighboring points in the organized point cloud are not necessarily part of the same surface. This is because neighboring in the organized point cloud is based on the sensor's capturing position rather than the actual 3D spatial arrangement.

The structure of the remaining sections of the paper is as follows: the next section gives an overview of the related work, followed by Section III, which provides details on the developed method. In the consecutive section, the evaluation and results are presented, and lastly, in Section V we conclude the paper with drawn conclusions and future work.

II. RELATED WORK

There are different approaches for point cloud segmentation. One approach works directly on the point cloud, as demonstrated by Diaz et al. [4], who proposed two methods for ground segmentation: Normal Vector-Based Filtering, which utilizes KNN, PCA, and Naïve Bayes classification, followed by RANSAC plane fitting to refine ground points; and Voxel-Based Filtering, which structures the point cloud into 3D voxels, applies height-based filtering, 3D adjacency segmentation, and statistical refinement. While the first method achieved slightly higher accuracy, the voxel-based approach was faster, making it the preferred choice for real-time applications. Another notable approach is the fast segmentation method proposed by Himmelsbach et al. [8], designed for autonomous ground vehicles. Their method splits the segmentation process into two steps: local ground plane estimation and fast 2D connected components labeling. This strategy efficiently processes large, unordered 3D point clouds by first separating ground and non-ground points using local plane fitting and then clustering the remaining points based on spatial connectivity. Golovinskiy and Funkhouser [6] introduced a min-cut-based segmentation method that formulates point cloud segmentation as a graph optimization problem. Their approach constructs a k-nearest neighbors graph, applies a background penalty function, and enforces foreground constraints to achieve robust segmentation. The segmentation is determined by solving a global min-cut optimization, which minimizes the cost of

separating object points from the background. The method supports both automatic and interactive segmentation and is particularly effective in complex urban environments where object-background separation is challenging. More recently, Huang et al. [9] introduced a coarse-to-fine MRF-based approach to improve ground segmentation accuracy while maintaining computational efficiency. Their method first performs coarse segmentation using local feature extraction to classify points into high-confidence obstacle, ground, and unknown points. The MRF model is then constructed using the coarsely segmented data, eliminating the need for prior knowledge. The graph cut algorithm minimizes the MRF model to refine segmentation results. Additionally, deep learning-based approaches have gained traction for efficient and accurate segmentation of LiDAR point clouds. One such method is SalsaNet, introduced by Aksoy et al. [1], which is an encoder-decoder-based deep learning model designed for fast road and vehicle segmentation. SalsaNet processes LiDAR point clouds in a Bird-Eye-View (BEV) projection and utilizes ResNet blocks in the encoder for efficient feature extraction. It also incorporates a class-balanced loss function to address the imbalance between road and vehicle classes in autonomous driving scenarios. Building upon SalsaNet, Cortinhal et al. [3] introduced SalsaNext, an improved network for semantic segmentation of LiDAR point clouds with uncertainty estimation. SalsaNext extends SalsaNet by incorporating a novel context module, a residual dilated convolution stack, and a pixel-shuffle layer in the decoder to improve segmentation accuracy while maintaining efficiency. Additionally, SalsaNext applies Bayesian treatment to estimate epistemic and aleatoric uncertainties, making it a robust choice for safety-critical applications such as autonomous driving.

III. METHOD

To overcome the limitations of ring-based segmentation approaches, such as the assumption that the LiDAR is mounted horizontally at the center of the robot, and to support multi-LiDAR configurations, we propose a method that utilizes the organized point cloud representation instead of partitioning the space into concentric rings. Our approach arranges each LiDAR data into a structured array, $\mathbb{R}^{m \times n \times 3}$, where m and n represent the sensor's vertical and horizontal resolution, respectively. This format preserves the spatial arrangement of points as captured by the sensor, with each cell storing its corresponding (x, y, z) coordinates. By maintaining this structured representation, we eliminate the need for a kd-tree [2] to find neighboring points when computing normal vectors. Traditional kd-tree search has a complexity of $\mathcal{O}(\log N)$ for nearest neighbor queries, where N is the number of points in the cloud. In contrast, our structured representation enables direct access to neighboring points in constant time $\mathcal{O}(1)$, reducing computational overhead. The segmentation pipeline consists of the following steps: (1) finding neighboring points based on their indices in the 2D structured array and removing outliers for each LiDAR, (2) estimating normal vectors, (3) classifying normal vectors, (4)

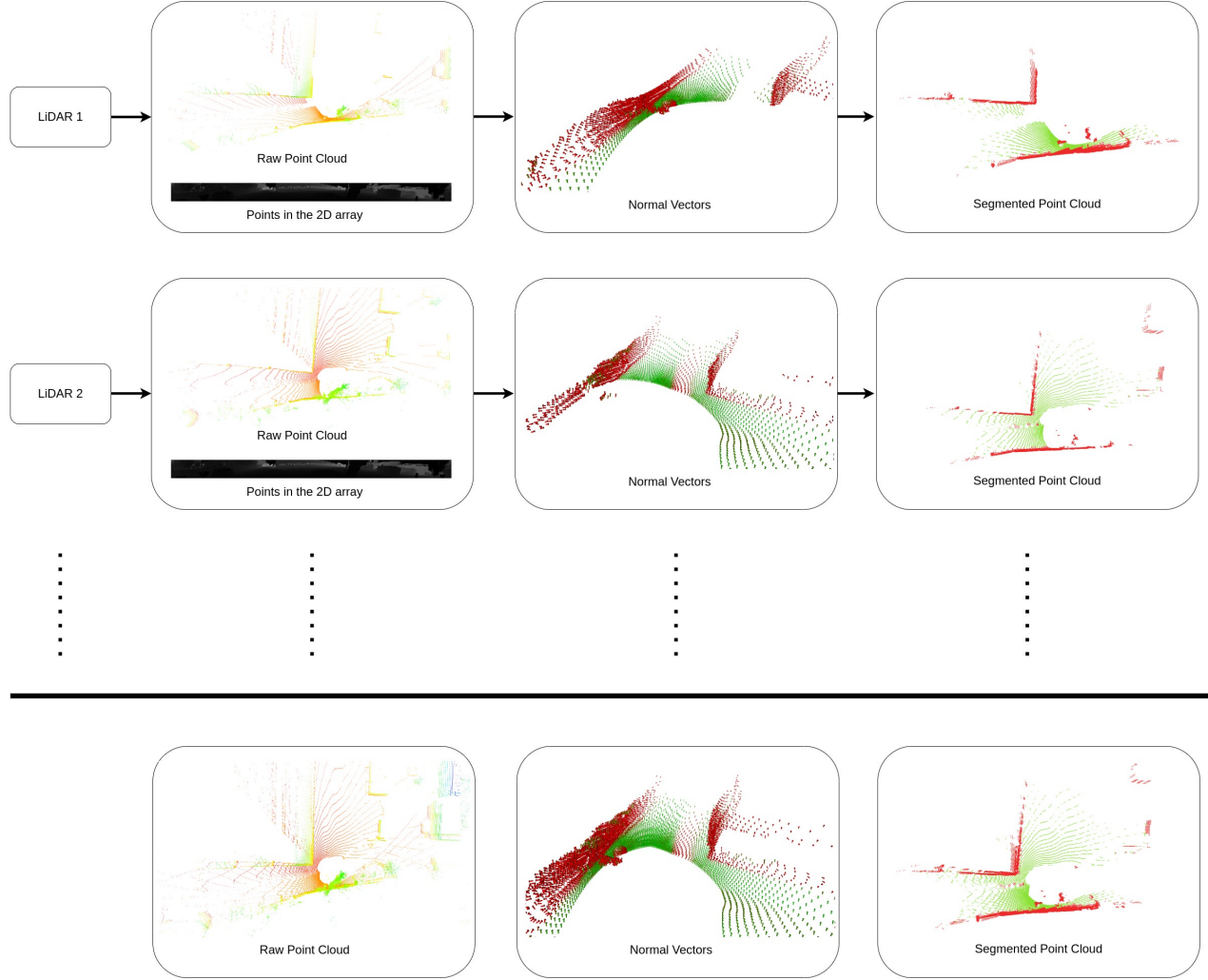


Fig. 1. Segmentation Pipeline: First, for each LiDAR, points are sorted into a structured 2D array. Then, normal vectors are estimated and labeled, followed by assigning points to their corresponding normal vectors.

assigning labels to the points using Likelihood Estimation, and (5) merge the labeled points from different LiDARs into one. A high-level overview of the pipeline is shown in Figure 1.

A. Rolling Window

Given an array where each cell corresponds to a point $P \in \mathbb{R}^{m \times n \times 3}$, we use a rolling window instead of fixed partitioning. This technique ensures that each point's local neighborhood is dynamically considered, leading to better spatial consistency and more accurate normal estimations. A rolling window of size $w \times w$ is defined as:

$$W_{i,j} = \left\{ \mathbf{p}_{u,v} \mid i - \frac{w}{2} \leq u \leq i + \frac{w}{2}, j - \frac{w}{2} \leq v \leq j + \frac{w}{2} \right\}. \quad (1)$$

Since being in the same window does not necessarily imply that all points belong to the same physical surface, we first apply an outlier removal step. Given a point $\mathbf{p} = (x, y, z) \in W_{i,j}$, we define its radial distance as:

$$d_{\mathbf{p}} = \sqrt{x^2 + y^2 + z^2}. \quad (2)$$

A point is considered an outlier and removed if:

$$\frac{|d_{\mathbf{p}} - \bar{d}_{w_{i,j}}|}{\bar{d}_{w_{i,j}}} > \tau_d, \quad (3)$$

where $\bar{d}_{w_{i,j}}$ is the mean distance of all points in $W_{i,j}$, and τ_d is a predefined threshold controlling the allowed deviation.

B. Normal Estimation

For each window $W_{i,j}$, we estimate the surface normal vector $\mathbf{n}_{w_{i,j}}$ by fitting a plane using PCA. Given a local set of k neighboring points $\mathcal{X}_{w_{i,j}} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_k\}$ from the window, the normal vector $\mathbf{n}_{w_{i,j}}$ is computed as:

$$\mathbf{n}_{w_{i,j}} = \arg \min_{\mathbf{n}} \sum_{\mathbf{p} \in \mathcal{X}_{w_{i,j}}} (\mathbf{n} \cdot (\mathbf{p} - \bar{\mathbf{p}}_{w_{i,j}}))^2, \quad (4)$$

where $\bar{\mathbf{p}}_{w_{i,j}}$ is the centroid of the points in $\mathcal{X}_{w_{i,j}}$.

C. Ground Classification

Since the robot operates on a traversable surface, each $w_{i,j}$ with a mean height $\bar{h}_{w_{i,j}}$ close to zero (in the robot's base frame) is initially classified as ground:

$$G_{w_{i,j}} = \begin{cases} 1, & \text{if } |\bar{h}_{w_{i,j}} - h_r| < \varepsilon_h, \\ -1, & \text{otherwise.} \end{cases} \quad (5)$$

Here, $G_{w_{i,j}} = 1$ indicates that the region is labeled as ground, while $G_{w_{i,j}} = -1$ denotes an unknown classification. h_r represents the reference ground height, and ε_h is a predefined height tolerance threshold.

Next, we apply an iterative expansion strategy. For each labeled ground window $w_{i,j}$, we iteratively check its neighboring window $w_{m,n}$ and classify it as ground if:

$$|\bar{h}_{w_{m,n}} - \bar{h}_{w_{i,j}}| < \delta_h \quad \text{and} \quad \theta_{m,n} < \theta_{\text{thresh}}, \quad (6)$$

where $w_{m,n}$ is a neighbor of $w_{i,j}$, δ_h is the allowed height difference between neighboring window, and $\theta_{m,n}$ is the surface inclination angle computed as:

$$\theta_{m,n} = \cos^{-1}(\mathbf{n}_{m,n} \cdot \mathbf{z}), \quad (7)$$

with $\mathbf{z}$ being the global vertical unit vector. This process is repeated iteratively until no new windows are labeled as ground. Finally, any remaining unlabeled windows are classified as non-ground.

D. Point Labeling Using Likelihood

After estimating the normal vectors and labeling the windows, we use a likelihood approach to estimate whether a given point belongs to a particular surface. This is particularly useful for correctly classifying points that are not direct neighbors in the 2D array but belong to the same physical surface due to edge continuity.

To achieve this, we model the likelihood of a point $\mathbf{p}$ belonging to the surface associated with the normal vector $\mathbf{n}_{w_{i,j}}$ as:

$$L(\mathbf{p} | \mathbf{n}_{w_{i,j}}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(d_{\mathbf{p},w_{i,j}})^2}{2\sigma^2}\right), \quad (8)$$

where $d_{\mathbf{p},w_{i,j}}$ is the perpendicular distance of the point $\mathbf{p}$ from the plane defined by the normal vector $\mathbf{n}_{w_{i,j}}$, and $\sigma_{w_{i,j}}$ represents the standard deviation of distances for points within $w_{i,j}$.

The perpendicular distance is computed as:

$$d_{\mathbf{p},w_{i,j}} = (\mathbf{p} - \bar{\mathbf{p}}_{w_{i,j}}) \cdot \mathbf{n}_{w_{i,j}}, \quad (9)$$

where $\bar{\mathbf{p}}_{w_{i,j}}$ is the centroid of $w_{i,j}$.

A point $\mathbf{p}$ is classified as belonging to the surface associated with normal vector $\mathbf{n}_{w_{i,j}}$ if its likelihood $L(\mathbf{p} | \mathbf{n}_{w_{i,j}})$ exceeds a predefined threshold τ_L .

To assign labels to individual points, we compute the likelihood of each point belonging to different windows and assign it the label of the window that maximizes the likelihood. Given a point $\mathbf{p}$, its assigned label is:

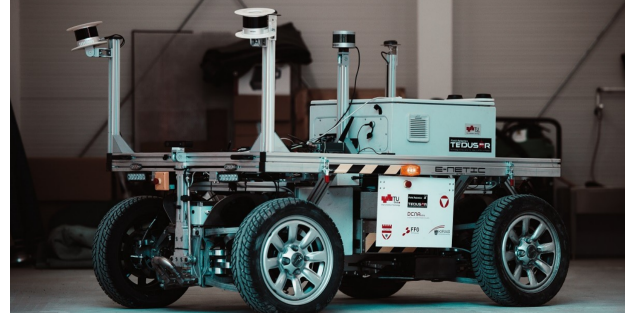


Fig. 2. The experimental robot setup equipped with two 32-layer Hesai LiDARs.

$$\ell(\mathbf{p}) = \arg \max_{w_{i,j}} L(\mathbf{p} | \mathbf{n}_{w_{i,j}}). \quad (10)$$

This approach ensures that each point is assigned to the most probable surface, leading to more consistent and accurate segmentation.

IV. RESULTS

AMRs are deployed in various environments, necessitating different LiDAR configurations based on the specific application. For instance, autonomous vehicles often utilize high-resolution 128-layer LiDARs, like in the KITTI dataset [5], to achieve a comprehensive 360-degree view. In contrast, our application involves operation in unstructured environments, where dense point cloud coverage in front of the robot is crucial for distinguishing traversable and non-traversable slopes. To achieve this without relying on an expensive 128-layer LiDAR, we employ two 32-layer Hesai LiDARs, mounted on the front left and right of the robot. This configuration enhances the density of LiDAR points in the robot's immediate path. The experimental robot setup and LiDAR configuration are illustrated in Figure 2.

To assess the performance of the developed method, we compute error metrics across five different scenarios, including slopes, unstructured environments, and campus areas, as shown in Figure IV, and compare them to Patchwork [12]. Additionally, we use Label Cloud [13] to manually annotate points in the point cloud. In the following section, we introduce the error metrics and evaluate the performance of the developed method across these five scenarios.

A. Error Metrics

To quantitatively evaluate the performance of our method, we employ five metrics: *Precision*, *Recall*, *F1 score*, *Accuracy*, and *Coverage*. These metrics assess the classification performance based on the number of correctly and incorrectly classified points.

Let the number of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) be denoted as N_{TP} , N_{TN} , N_{FP} , and N_{FN} , respectively. The evaluation metrics are defined as follows:

- **Precision** (Positive Predictive Value): measures the proportion of correctly identified positive instances among

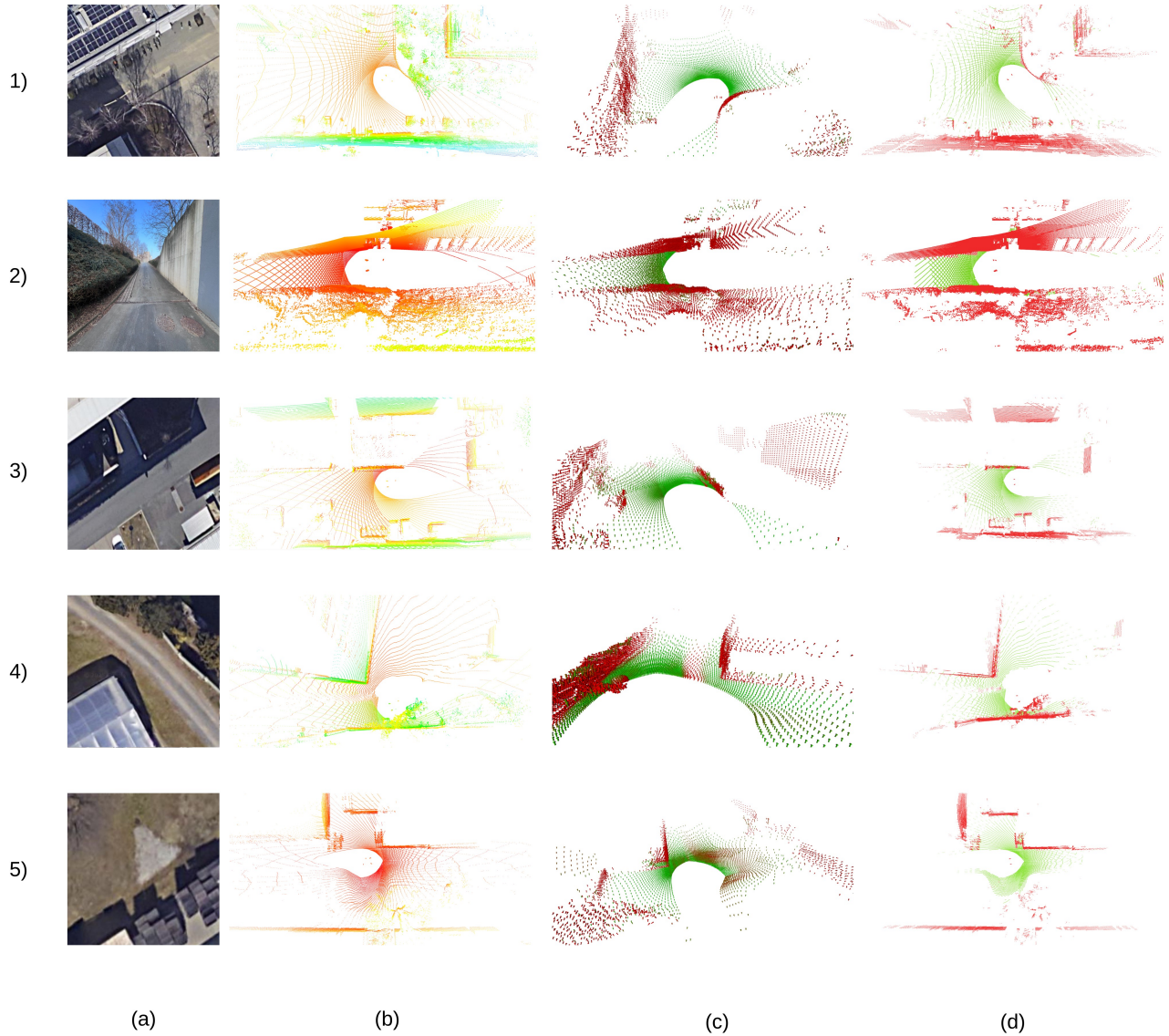


Fig. 3. Illustration of the ground segmentation process. (a) environment of the scenario, (b) raw LiDAR point cloud with colors representing point height. (c) extracted normal vectors and classification results, distinguishing segmented ground (green) and non-ground (red) regions. (d) final segmented point cloud, where green points are labeled as ground and red points as non-ground. Scenario 1 and 3 feature flat ground, scenario 2 includes a slope in front of the robot, scenario 4 depicts a road with a ditch on the right side, and scenario 5 represents an off-road area with varying slopes.

all predicted positive instances.

$$\text{Precision} = \frac{N_{TP}}{N_{TP} + N_{FP}} \quad (11)$$

- **Recall** (Sensitivity or True Positive Rate): measures the proportion of correctly identified positive instances out of all actual positive instances.

$$\text{Recall} = \frac{N_{TP}}{N_{TP} + N_{FN}} \quad (12)$$

- **F1 Score**: the harmonic mean of Precision and Recall, providing a balanced measure of model performance, especially in cases of class imbalance.

$$F_1 = \frac{2 \cdot N_{TP}}{2 \cdot N_{TP} + N_{FP} + N_{FN}} \quad (13)$$

- **Accuracy**: measures the overall proportion of correctly classified instances out of all instances.

$$\text{Accuracy} = \frac{N_{TP} + N_{TN}}{N_{TP} + N_{TN} + N_{FP} + N_{FN}} \quad (14)$$

- **Coverage**: measures the proportion of labeled points among all data points.

$$\text{Coverage} = \frac{N_{TP} + N_{TN} + N_{FP} + N_{FN}}{N_{\text{total}}} \quad (15)$$

A high Precision indicates fewer false positives, while a high Recall suggests fewer false negatives. The F1 Score provides a trade-off between these two metrics, and Accuracy gives an overall measure of classification correctness. Coverage ensures an assessment of how much of the dataset

is labeled. This measure also depends on the sensor setup; for instance, in our setup, since the sensor is installed tilted, as we move farther from the sensor, the density of points in an area becomes lower and lower, making it difficult to calculate normal vectors. These metrics collectively provide a comprehensive evaluation of the segmentation performance.

B. Performance Evaluation

One of the key aspects of evaluating our method is the coverage of labeled points, which directly impacts segmentation accuracy. Labeling a point requires the assignment of a normal vector, but in some cases, this is not feasible. For instance, points that are farther from the LiDAR sensor tend to have larger spatial gaps, making it challenging to compute a reliable normal vector. Additionally, points in high-variance regions, such as those affected by vegetation or irregular surfaces, may lack sufficient neighboring points to form a well-defined surface. In such cases, points remain unlabeled due to insufficient data.

The results indicate that scenarios with a higher presence of trees, such as Scenario 1, tend to have a lower coverage value, as a greater proportion of points do not belong to distinct, continuous surfaces. Conversely, environments with fewer trees lead to higher coverage. On average, across the five evaluated scenarios, our method achieves a coverage value of 0.897, as detailed in Table I. PatchWork shows better coverage since it uses the entire point cloud for labeling rather than processing each LiDAR separately. It also performs better in scenarios with fewer slopes. Overall, while PatchWork achieves higher coverage, our method demonstrates better average performance across the five scenarios.

Scenario	Coverage		Accuracy		Precision		Recall		F1 Score	
	Ours	Patchwork	Ours	Patchwork	Ours	Patchwork	Ours	Patchwork	Ours	Patchwork
1	0.749	0.972	0.84	0.912	0.828	0.898	0.843	0.914	0.834	0.904
2	0.907	0.961	0.971	0.925	0.892	0.777	0.971	0.92	0.930	0.820
3	0.920	0.981	0.952	0.971	0.953	0.972	0.951	0.970	0.952	0.971
4	0.962	0.926	0.970	0.921	0.955	0.907	0.970	0.921	0.962	0.913
5	0.949	0.956	0.961	0.951	0.948	0.938	0.960	0.950	0.955	0.945
Avg	0.897	0.959	0.939	0.936	0.915	0.898	0.939	0.935	0.927	0.911

TABLE I
PERFORMANCE METRICS COMPARISON: OURS VS. PATCHWORK [12]

The segmentation method demonstrates high accuracy across different environments, as indicated by the average accuracy of 0.939 and an F1-score of 0.927. These values suggest that the approach consistently distinguishes ground points from non-ground points with minimal misclassifications. The average precision of 0.915 indicates that false positives were minimized, meaning non-ground points were rarely misclassified as ground, while the Recall (0.939 avg.) confirms that most ground points were correctly identified.

V. CONCLUSION AND FUTURE WORK

In this paper, we presented a LiDAR-based ground segmentation method that efficiently preserves spatial structure by using an organized point cloud, making it adaptable to different sensor configurations. By directly accessing neighboring points, we extract normal vectors and classify them as

ground or obstacles. Furthermore, we assign points to normal vectors using a likelihood-based approach. Experimental evaluations across five diverse scenarios showed an average accuracy of 0.939 and an F1-score of 0.927, demonstrating its reliability in distinguishing ground from non-ground points. The method also adapts well to unstructured terrains and multi-LiDAR configurations, proving useful for real-world robotic navigation.

One limitation of our work is that we process each point cloud separately and do not consider the overlap between point clouds. This overlap can be addressed in future work by incorporating LiDAR transformations relative to each other. By doing so, we can directly access local neighboring points from multiple LiDARs, improving segmentation accuracy and consistency.

REFERENCES

- [1] E. E. Aksoy, S. Baci, and S. Cavdar, "Salsanet: Fast road and vehicle segmentation in lidar point clouds for autonomous driving," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 926–932.
- [2] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, p. 509–517, Sept. 1975. [Online]. Available: <https://doi.org/10.1145/361002.361007>
- [3] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "Salsanet: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving," 2020.
- [4] N. Diaz, O. Gallo, J. Caceres, and H. Porras, "Real-time ground filtering algorithm of cloud points acquired using terrestrial laser scanner (tls)," *International Journal of Applied Earth Observation and Geoinformation*, vol. 105, p. 102629, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0303243421003366>
- [5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.
- [6] A. Golovinskiy and T. Funkhouser, "Min-cut based segmentation of point clouds," in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 2009, pp. 39–46.
- [7] A. Grover and M. H. Ashraf, "Leveraging autonomous mobile robots for industry 4.0 warehouses: A multiple case study analysis," *The International Journal of Logistics Management*, vol. 35, 07 2023.
- [8] M. Himmelsbach, F. v. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3d point clouds for ground vehicles," in *2010 IEEE Intelligent Vehicles Symposium*, 2010, pp. 560–565.
- [9] W. Huang, H. Liang, L. Lin, Z. Wang, S. Wang, B. Yu, and R. Niu, "A fast point cloud ground segmentation approach based on coarse-to-fine markov random field," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7841–7854, 2022.
- [10] R. Keith and H. M. La, "Review of autonomous mobile robots for the warehouse environment," *arXiv*, vol. 2406.08333, 2024. [Online]. Available: <https://arxiv.org/abs/2406.08333>
- [11] I. Kubasáková, J. Kubáňová, D. Benčo, and N. Fábryová, "Application of autonomous mobile robot as a substitute for human factor in order to increase efficiency and safety in a company," *Applied Sciences*, vol. 14, no. 13, 2024. [Online]. Available: <https://www.mdpi.com/2076-3417/14/13/5859>
- [12] H. Lim and J. Kim, "Patchwork: Robust ground segmentation in point clouds for autonomous vehicles," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 932–939. [Online]. Available: <https://doi.org/10.1109/ICRA2021.9561584>
- [13] C. Sager, P. Zschech, and N. Kühl, "labelcloud: A lightweight domain-independent labeling tool for 3d object detection in point clouds," 2021.
- [14] D. Zermas, I. Izzat, and N. Papanikolopoulos, "Cnn for very fast ground segmentation in velodyne lidar data," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2931–2938, 2018.
- [15] X. Zhao and C. Wu, "Markov random field for ground segmentation in 3d lidar data," *Remote Sensing*, vol. 2, no. 3, pp. 833–852, 2010. [Online]. Available: <https://www.mdpi.com/2072-4292/2/3/833>

Multi-Waypoint Path Planning and Motion Control for Non-holonomic Mobile Robots in Agricultural Applications

Mahmoud Ghorab and Matthias Lorenzen

Abstract—There is a growing demand for autonomous mobile robots capable of navigating unstructured agricultural environments. Tasks such as weed control in meadows require efficient path planning through an unordered set of coordinates while minimizing travel distance and adhering to curvature constraints to prevent soil damage and protect vegetation. This paper presents an integrated navigation framework combining a global path planner based on the Dubins Traveling Salesman Problem (DTSP) with a Nonlinear Model Predictive Control (NMPC) strategy for local path planning and control. The DTSP generates a minimum-length, curvature-constrained path that efficiently visits all targets, while the NMPC leverages this path to compute control signals to accurately reach each waypoint. The system's performance was validated through comparative simulation analysis on real-world field datasets, demonstrating that the coupled DTSP-based planner produced smoother and shorter paths, with a reduction of about 16% in the provided scenario, compared to decoupled methods. Based thereon, the NMPC controller effectively steered the robot to the desired waypoints, while locally optimizing the trajectory and ensuring adherence to constraints. These findings demonstrate the potential of the proposed framework for efficient autonomous navigation in agricultural environments.

Index Terms—Motion Planning and Control, Agricultural Robots, Dubins Traveling Salesman Problem, Model Predictive Control.

I. INTRODUCTION

Autonomous navigation in unstructured agricultural environments, such as meadows, poses significant challenges due to unpredictable terrain, the non-holonomic system dynamics of many mobile robots, and the possible presence of both static and dynamic obstacles [15]. An ecological weed control system is a prime application where efficient navigation is crucial, enabling the reduction of herbicide use and minimizing human intervention. In the considered application, the process begins by selecting a geo-fence that defines the field's safety boundaries, ensuring the robot operates within a designated area. Next, the target weeds are autonomously detected and mapped during a scanning phase. Once the scanning and mapping are complete, the robot is tasked with navigating to the identified weeds and eliminating them using a mechanical weed removal tool, avoiding the use of chemical herbicides. This last phase is the primary focus of this work, where the proposed DTSP-based global path planner, as well as the NMPC local path

planner and waypoint-following controller are integrated to efficiently guide the robot to each detected weed, while considering the different robot and environmental constraints.

However, the order in which the targets should be visited is not determined a priori. Hence, the objective of the global path planner is to generate a feasible path of minimum length that efficiently visits all the targets. The problem of determining the order of waypoints to minimize travel distance is typically formulated as an Euclidean Traveling Salesman Problem (ETSP). However, solving the ETSP alone does not consider the vehicle's non-holonomic constraints or environmental constraints, such as avoiding damage to soil and healthy grass by preventing arbitrarily sharp turns in the path. Therefore, the planner has to consider curvature constraints, by generating a minimum length path making use of Dubins curves instead of straight line segments. This formulation, known as the Dubins Traveling Salesman Problem (DTSP), extends the classical ETSP to non-holonomic vehicles with a minimum turning radius constraint [20].

While the DTSP based planner provides a feasible global path to guide the robot towards each waypoint, a local planner and controller is essential for ensuring safe and adaptive navigation in dynamic environments and computing the necessary control input. The presence of static obstacles and dynamic agents, such as animals, human workers or other robots operating in the field, requires real-time local path replanning. To this end, Nonlinear Model Predictive Control (NMPC) is employed as both the local path planner and waypoint following controller within the same framework.

A. Related Work

Various formulations and extensions of the DTSP and NMPC have been presented in the literature, each with its own advantages and trade-offs. Selecting the right combination of DTSP and NMPC formulations is crucial for achieving efficient and reliable navigation in agricultural environments. The choice directly impacts the optimality of the generated paths, the overall motion control objectives, and the ability to meet specific task requirements while adhering to overall system's constraints.

Approaches to solving DTSP primarily differ in how they determine the ordering of waypoints and compute the associated orientations. These differences influence the accuracy of the near optimal solution, and the computational effort. Similarly, NMPC formulations vary in terms of cost function design, constraints handling, and real-time performance, making the selection process highly application-dependent.

This work was supported by the BMBF, Deutsche Agentur für Transfer und Innovation within the program DATipilot.

Mahmoud Ghorab and Matthias Lorenzen are with Institute for Applied Artificial Intelligence and Robotics (IKR), Kempten University of Applied Sciences, Bahnhofstraße 61, 87435 Kempten (Allgäu), Germany {mahmoud.ghorab, matthias.lorenzen}@hs-kempten.de

This work emphasizes the importance of choosing the most suitable DTSP and NMPC formulations tailored to agricultural applications, balancing global path feasibility, motion planning adaptability, and considering real-world operational constraints.

1) *DTSP-based Global Path Planning*: In [3] Dubins introduced a method for determining the shortest path in a 2D space, given curvature constraints as well as the entry and exit orientations between two points as input. The resulting path consists of a combination of straight line segments and arcs with radii that adhere to the vehicle's curvature constraints.

The DTSP was first introduced by [20]. In this extension of the classical TSP, the path connecting any two points must be a Dubins curve and two curves that meet at the same point must share the same orientation.

The core distinctions between methods addressing the DTSP lie in how they determine the ordering of the waypoints and calculate the orientations associated with the points. Interested readers are referred to the comprehensive survey [13] for a detailed review of the various routing methods.

Existing literature mostly adopted a decoupled approach for route generation [20], [12], [18], [14]. Thereby, first, the visiting sequence is determined solving the ETSP. Then, the vehicle's orientation at each point is defined, for example, using the Alternating Algorithm (AA) [20]. Finally, the waypoints are connected with Dubins curves. However, relying solely on the Euclidean distance metric to define the visit order does not necessarily yield efficient results when using Dubins curves for path generation. This approach can lead to excessive circular maneuvers, especially in dense waypoint configurations typical of autonomous weed control applications. Since the optimization of waypoint coordinates and headings is inherently coupled, decoupling them compromises optimality [23]. As a result, a tour based solely on the ETSP ordering cannot achieve an approximation ratio better than $O(n)$ (i.e., the best solution is within a factor of n of the optimal solution) see [17].

In the coupled approach, the sequence is determined by directly using the lengths of the Dubins curves between pairs of points. However, the main challenge here is to find the right mechanism to determine the entry and exit orientations without even having a predefined sequence of points. In [10], the orientations of all points are initially set to zero (or a fixed random value), and all interconnecting curves are calculated and connected to form a complete graph. An instance of the Asymmetric TSP (ATSP) is then solved to find the shortest path in this graph. This method was later extended to include a complete heading discretization [9]. The technique involves selecting a finite set of k possible headings at each waypoint. A graph is created with n clusters, each representing a waypoint and containing k nodes that correspond to different headings. Subsequently, the Dubins distance between configurations of node pairs from different clusters is computed. Finally, a tour through all clusters, containing exactly one point per cluster, is then determined. A

logarithmic approximation ratio $O(\log(n))$ for this ATSP can be achieved by directly solving the problem using available algorithms implementations, such as those described in [7], [5], [8].

In both DTSP formulations and most global planners in general, solutions are computed under tight time constraints, often resulting in suboptimal paths based on simplified models. Consequently, there is considerable room for improvement by integrating appropriate motion planning and control systems to further locally optimize the global path.

2) *NMPC-based Motion Planning*: The fundamental principle of MPC is to use the system's model to forecast its future behavior and optimally adjust control actions by solving a constrained optimization problem over a receding horizon at each sampling time [19], [6]. By minimizing a cost function that incorporates possible nonlinear multi-input multi-output (MIMO) system dynamics along with state and input constraints, NMPC has proven to be a promising approach for various applications, including stabilization, tracking, and motion planning of mobile robots in unstructured and dynamic environments [16], [2], [22], [11].

In automated weed control applications, the primary objective is for the robot to reach and stop at each designated waypoint. This is ensured by making the state corresponding to the desired pose a stable attractor of the feedback control loop. A conventional approach to ensure this with NMPC involves enforcing terminal costs and/or terminal region constraints near the desired set-point. However, when the set-point is located at relatively long distance from the robot, the prediction horizon required becomes prohibitively long for practical applications. An alternative strategy is to reformulate the problem as one of path following by generating a global path that connects all waypoints and then following this path piece-wise [4], [25], [16].

In the considered application, as in many other applications, the goal is to reach the target while satisfying constraints rather than strictly following a specific path. As noted in Section I-A.1, global planners often yield suboptimal paths when computed in finite time, particularly under kinematic and dynamic constraints. Therefore, exactly following these paths can complicate motion control and make it impossible when real-time obstacle avoidance is required. Instead, a flexible approach that allows the motion planner to dynamically optimize the global path and find shortcuts is preferred.

A novel NMPC formulation, proposed in [11], guarantees convergence to a desired target while ensuring closed-loop stability, adherence to system constraints, and collision avoidance with obstacles. The method optimally selects an artificially generated reference set-point, dynamically adjusted along the global reference path, which guides the robot without requiring strict path following. This artificial reference is used to define feasible stabilizing terminal constraints.

This work adapts and integrates the coupled DTSP formulation from [9] with the NMPC-based motion planner from [11] to enable an automated, robot-based weed control

application. The resulting integrated framework addresses a critical gap in applied research by combining a multi-waypoint, curvature-constrained DTSP-based global planner with an advanced NMPC-based local motion planner and controller tailored for agricultural robots.

The remainder of the paper is organized as follows. Section II details the proposed system, explaining the integration of the DTSP-based global path planner with the NMPC-based local path planner and waypoint follower. Section III describes the simulation setup and presents a comparative analysis of the results. Finally, Section IV concludes the paper and outlines directions for future work.

II. PROPOSED NAVIGATION AND CONTROL SYSTEM

A. System Overview

The proposed system integrates a two-layer architecture for autonomous navigation. The global path planner, based on the coupled DTSP formulation, processes unordered multi-waypoint coordinates to compute an optimal sequence of curvature-constrained Dubins paths connecting these waypoints. These paths minimize travel distance while adhering to curvature constraints tailored specifically for agricultural applications, where sharp turns can damage the soil and grass. The NMPC-based local path planning and waypoint following algorithm utilizes the resulting global Dubins path to ensure precise convergence to each waypoint while respecting different system constraints.

B. DTSP Algorithm

Given W waypoints in a 2D space, the DTSP aims to determine the shortest path that connects all points while adhering to curvature constraints. Consequently, the path between any two points should be a Dubins curve, and the curves meeting at the same point must share the same orientation.

The following steps present the DTSP routing problem based on [9]:

- 1) For each of the W target points, select K candidate headings (e.g., $k\frac{2\pi}{K}$ for $k \in \{0, 1, \dots, K-1\}$).
- 2) Represent each target as a cluster of K nodes, where each node corresponds to a configuration $q_i = (p_i, \theta_i)$ with position p and a candidate heading θ . The total number of nodes is nK .
- 3) For each pair of nodes q_i and q_j that belong to different clusters (i.e., different targets), compute the Dubins curve with minimum distance $\mathcal{D}_p(q_i, q_j)$. This curve is parameterized by the minimum turning radius ρ , defines the cost for traveling from a specific configuration at target i to a different one at target j .
- 4) Arrange the computed Dubins distances into a cost matrix M of size $N \times N$, where $N = nK$.

From the matrix M , one can construct an ordered sequence $\mathbf{Q}_\Sigma = (q_{\Sigma(0)}, q_{\Sigma(1)}, \dots, q_{\Sigma(N-1)})$ which represent some permutation Σ of configurations $q_{\Sigma(i)} = (p_{\Sigma(i)}, \theta_{\Sigma(i)})$ of a complete tour of the mobile robot, after excluding transitions between configurations within the same target.

Based on this representation, the corresponding objective function can be formulated as follows:

$$\underset{\theta, \Sigma}{\text{minimize}} \quad \mathcal{L}_p(\mathbf{Q}_\Sigma) \quad (1)$$

Where the cost function is defined as:

$$\mathcal{L}_p(\mathbf{Q}_\Sigma) = \mathcal{D}_p(q_{\Sigma(N-1)}, q_{\Sigma(0)}) + \sum_{i=0}^{N-2} \mathcal{D}_p(q_{\Sigma(i)}, q_{\Sigma(i+1)}) \quad (2)$$

C. NMPC Algorithm

The robot's motion is governed by a discrete-time, nonlinear dynamic system, described by the following difference equation:

$$\mathbf{x}(n+1) = f(\mathbf{x}(n), \mathbf{u}(n)), \quad (3)$$

where $f: \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$ is a continuous function that models the system dynamics. Here, $\mathbf{x}(n) \in \mathbb{R}^{n_x}$ represents the system state, while $\mathbf{u}(n) \in \mathbb{R}^{n_u}$ denotes the control input at the sampling time t_n , where $n = 0, 1, 2, \dots$.

The global path $\mathcal{P}_d$ generated from the DTSP-based planner can be represented as a sequence of path segments connecting each pair of consecutive waypoint poses as follows:

$$\mathcal{P}_d = (p_0, p_1, \dots, p_{W-1}), \quad (4)$$

where W is the total number of waypoints. Each path segment p_w is described as a continuous function:

$$p_w: [0, 1] \mapsto \mathbb{R}^{n_x}, \quad (5)$$

where $p_w(0)$ represents the initial configuration of the path segment, while $p_w(1)$ represents the target configuration.

The following NMPC formulation used in this work was originally proposed in [11]. This approach ensures that both constraint satisfaction and convergence to a desired target can be guaranteed. Unlike traditional path-following approaches, this method does not require the robot to strictly follow the reference path p_w . Instead, the path only serves as a guidance mechanism to identify a suitable terminal constraint, which guarantees that at each control step, the local solution computed by the NMPC algorithm can be suitably extended to reach the target pose. This is achieved by introducing an artificial reference, which serves as an intermediate target configuration and is optimized within the NMPC optimization problem.

In the following, the predicted state and control input trajectories over the finite prediction horizon N are denoted as $\bar{\mathbf{x}}(\cdot) \in X$ and $\bar{\mathbf{u}}(\cdot) \in U$, where X and U represent the set of admissible states and inputs respectively. These trajectories are defined as

$$\bar{\mathbf{x}}(\cdot) = (\bar{\mathbf{x}}(1), \bar{\mathbf{x}}(2), \dots, \bar{\mathbf{x}}(N)), \quad (6)$$

$$\bar{\mathbf{u}}(\cdot) = (\bar{\mathbf{u}}(0), \bar{\mathbf{u}}(1), \dots, \bar{\mathbf{u}}(N-1)). \quad (7)$$

The artificial reference is chosen along the current path segment p_w . With the additional optimization variable $\bar{s} \in [0, 1]$ and the path p_w , this artificial reference is given by $p_w(\bar{s})$.

The MPC cost function is defined by

$$J_N(\mathbf{x}_0, \bar{\mathbf{x}}(\cdot), \bar{\mathbf{u}}(\cdot), \bar{s}) = \sum_{k=0}^{N-1} \ell(\bar{\mathbf{x}}(k), \bar{\mathbf{u}}(k)) + V_o(\bar{s}), \quad (8)$$

where the stage cost $\ell : \mathbb{R}^{n_x+n_u} \rightarrow \mathbb{R}_{\leq 0}$ and offset cost $V_o : [0, 1] \rightarrow \mathbb{R}_{\geq 0}$ are positive definite functions. We define the stage cost

$$\ell(\bar{\mathbf{x}}(k), \bar{\mathbf{u}}(k)) = \|\bar{\mathbf{x}}(k) - p(\bar{s})\|_Q^4 + \|\bar{\mathbf{u}}(k)\|_R^4, \quad (9)$$

where Q and R are positive definite weighting matrices that penalize the deviation of the predicted states from the intermediate artificial reference pose and penalize excessive control effort, respectively.

The offset cost $V_o(\bar{s})$ ensures that the artificial reference progresses forward toward the final target pose $p_w(1)$ as it penalizes the distance along the path between the current artificial reference and the target pose. Is defined by

$$V_o(\bar{s}) = q_s(1 - \bar{s})^2, \quad (10)$$

where q_s is a positive weighting scalar that penalizes the deviation between the final reference index 1 and the current optimal intermediate artificial reference $\bar{s}$.

Finally, the NMPC algorithm at each sampling time t_n , $n = 0, 1, 2, \dots$, can be described as follows:

- 1) Measure the state $\mathbf{x}(n) \in X$ of the robot.
- 2) Set $\mathbf{x}_0 = \mathbf{x}(n)$, solve the optimal control problem (OCP) defined by:

$$\text{minimize}_{\bar{\mathbf{u}}(\cdot), \bar{s}} J_N(\mathbf{x}_0, \bar{\mathbf{x}}(\cdot), \bar{\mathbf{u}}(\cdot), \bar{s}) \quad (11a)$$

$$\text{s.t. } \bar{\mathbf{x}}(0) = \mathbf{x}_0 \quad (11b)$$

$$\bar{\mathbf{x}}(k+1) = f(\bar{\mathbf{x}}(k), \bar{\mathbf{u}}(k)), \quad k \in [0, N-1] \quad (11c)$$

$$\bar{\mathbf{x}}(k) \in X, \quad k \in [1, N] \quad (11d)$$

$$\bar{\mathbf{u}}(k) \in U, \quad k \in [0, N-1] \quad (11e)$$

$$\bar{\mathbf{x}}(N) = p(\bar{s}) \quad (11f)$$

$$\bar{s} \in [0, 1] \quad (11g)$$

$$\mathcal{B}(\bar{\mathbf{x}}(k)) \cap \mathcal{O}_i = \emptyset, \quad k \in [1, N], i \in [1, N_o] \quad (11h)$$

- 3) Denote the obtained optimal solution $\mathbf{u}^*(\cdot)$, $\mathbf{x}^*(\cdot)$, s^* .
- 4) Apply the control input $\mathbf{u}(n) = \mathbf{u}^*(0)$ to the system.
- 5) Repeat until the robot reaches the final waypoint, then start over using the next path segment.

General constraints on states and control inputs for nonlinear systems are incorporated into the OCP in the form of set membership conditions, as defined in (11d) and (11e), respectively. Furthermore, static obstacle avoidance can be also considered in the optimization problem by considering constraints (11h). Where $\mathcal{B}$ represents the robot's footprint, and $\mathcal{O}_i$ denotes the i -th obstacle in the environment.

III. RESULTS

The proposed system is evaluated in a simulated agricultural scenario, where a mobile robot navigates to a set of target weeds. The results are presented in terms of path planning and waypoint-following performance metrics, including path

length, target reaching, smoothness, and curvature constraints adherence. A comparative analysis is conducted between the proposed DTSP planner with angle discretization and the decoupled approach based on the Alternating Algorithm (AA), see Section I-A and [20]. The results demonstrate the effectiveness of the integrated global planner and NMPC methods adapted in this work.

A. Simulation Setup

The simulation scenario consists of a 2D field with a set of target weeds distributed across the area. In this phase, the global path planner generates a Dubins path that connects all target weeds in the field, while the NMPC controller optimizes the robot's trajectory to reach each detected weed accurately while adhering to constraints from the robot's kinematics and the environment.

After formulating the DTSP and transforming it into an ATSP, the problem was solved using the LKH optimizer, which is an effective implementation of the Lin-Kernighan traveling salesman heuristic [7].

The NMPC problem is symbolically formulated in MATLAB using the CasADi framework [1]. To ensure a smooth and continuously differentiable path function, the global Dubins reference path is first sampled at 5 cm intervals and then converted into a CasADi function, $p(s)$, using CasADi's linear interpolation utilities. This function is parameterized over the normalized domain $s \in [0, 1]$.

In this agricultural application a differential-driven mobile robot model as described in [21] is utilized:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v \cos(\theta) \\ v \sin(\theta) \\ \omega \end{bmatrix} \quad (12)$$

The robot's control inputs are defined as $\mathbf{u} = [v \ \omega]^T$, where v and ω represent the linear and angular velocity respectively. The output states of the robot are given by $\mathbf{x} = [x \ y \ \theta]^T$, which represent the 2D pose of the robot, including its position (x, y) and orientation θ . This mathematical model is employed for both the simulation and prediction models, without taking into account possible process or measurement noise.

The prediction model is integrated using the fourth-order Runge-Kutta (RK4) method to compute the state evolution over each discretization interval. The continuous-time OCP is discretized via direct multiple shooting, which converts it into a nonlinear programming (NLP) problem that is then solved with the Interior Point Optimizer (IPOPT) [24].

The NMPC problem is parameterized by a sampling time of $\Delta t = 0.1$ seconds and a prediction horizon of $N = 20$. The weight matrices are defined as

$$Q = \text{diag}(0.1, 0.1, 0.01),$$

$$R = \text{diag}(0.1, 1.0),$$

$$q_s = 10^4.$$

The minimum turning radius constraint, required in this application, is enforced by the inequality constraint

$$\bar{v}(k) \geq r_{\min} |\bar{\omega}(k)|$$

which is added to the optimal control problem. Furthermore, control inputs box constraints

$$\mathbf{u}_{min} \leq \bar{\mathbf{u}}(k) \leq \mathbf{u}_{max}$$

are taken into account to limit the robot's linear and angular velocity. Finally, to ensure smooth motion, in such agricultural application it is convenient to also consider constraints on the rate of change of control inputs (i.e., acceleration of the robot)

$$\Delta \mathbf{u}_{min} \leq \bar{\mathbf{u}}(k) - \bar{\mathbf{u}}(k-1) \leq \Delta \mathbf{u}_{max}.$$

The robot considered in this work has maximum linear velocity of 0.5 m/s and a maximum angular velocity of 1.9 rad/s. The rate of change constraints are defined as a fraction of the maximum control values, allowing adaptation based on operational requirements (e.g., $\mathbf{u}_{max}/n$), where $n \in [1, N_o]$

B. Simulation Results

The test scenario illustrated in Fig. 1 evaluates the performance of the proposed DTSP global path planner (Fig. 1a) against a DTSP planner from the decoupled category (Fig. 1b), as discussed in Section I-A. This planner utilizes the Alternating Algorithm (AA) to determine the waypoints orientations, whereas the DTSP method applied in this work incorporates 10 angle discretization levels for each waypoint. Both planners were tested on the same dataset, consisting of 150 target weeds distributed across approximately 20×60 square meters field, with a vehicle turning radius constraint of 0.5 meters.

In both cases, the proposed NMPC algorithm was able to optimize the reference paths and accurately reach each waypoint, while still respecting the turning radius constraints required to protect the soil and grass from damage. A steady-state error of no more than 0.05 meters was achieved at each target pose.

The proposed DTSP planner presented in Fig. 1a, achieved a total path length of 323.49 meters, outperforming the decoupled approach shown in Fig. 1b, which resulted in a path length of 384.58 meters, i.e. nearly 19% longer. For reference, the shortest possible path computed by only solving the ETSP without considering curvature constraints was 314.20 meters.

As observed in Fig. 1b, the path generated by the DTSP planner using the alternating algorithm is suboptimal, characterized by numerous loops that are necessary to reach the next waypoint given the curvature constraints. In contrast, the proposed DTSP planner with 10 angle discretization levels, as shown in Fig. 1a, leads to a different order of the waypoints, allowing for a significantly smoother path. This path connects all targets with hardly any redundant loops, which can effectively guide the NMPC towards the targets. Experiments with angle discretization, starting from three orientations per waypoint and incrementally increasing, showed that higher discretization levels generally reduced path cost but also increased computational time. This trade-off depends on factors such as the density of targets and the turning-radius constraints. Furthermore, the benefits of using

a coupled approach quickly grow with a higher target density and a larger minimum turning radius.

As depicted in Fig. 1, the robot successfully navigates all target weeds accurately while adhering to curvature constraints. Thereby the proposed NMPC does not strictly follow the reference path but locally optimizes the trajectory based on the NMPC cost function. E.g., to protect the soil, tight turns are discouraged, leading to wider turns to smooth out tight turns from the global planner, as long as this does not significantly increase the path length. On the other hand, it takes shortcuts by making tighter turns when this helps to significantly reduce the travel distance. This local planning behavior of the NMPC can be tuned by adjusting the prediction horizon length, the cost function weights, and the allowable turning radius.

IV. SUMMARY AND OUTLOOK

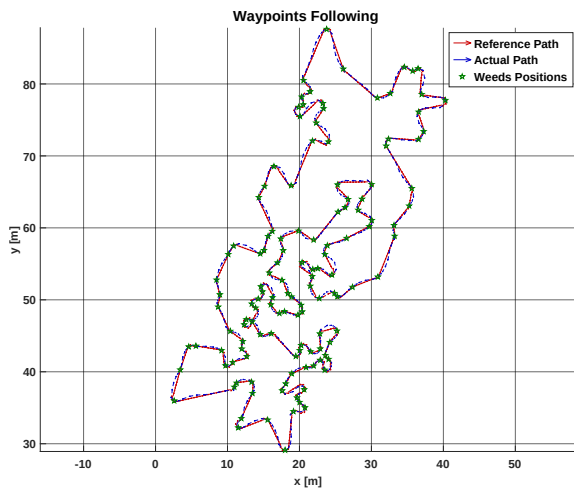
This paper has presented a practical autonomous navigation framework for non-holonomic mobile robots in agricultural applications. Given target coordinates, the proposed framework integrates a global path planner based on a coupled DTSP formulation with an NMPC-based motion planning and control strategy to generate feasible reference paths and compute optimal control inputs that satisfy both the robotic system constraints and the operational demands of the agricultural environment.

The system's performance was validated through a comparative analysis with a reference path generated by a global planner based on a decoupled DTSP formulation, demonstrating the advantages of the applied DTSP approach and its effectiveness as a reference input for the local motion planner and controller. By optimally selecting a feasible artificial reference and corresponding terminal constraint along the planned path, the NMPC methodology smooths out sharp turns, identifies efficient shortcuts, and ensures precise waypoint navigation while maintaining overall system stability under various constraints.

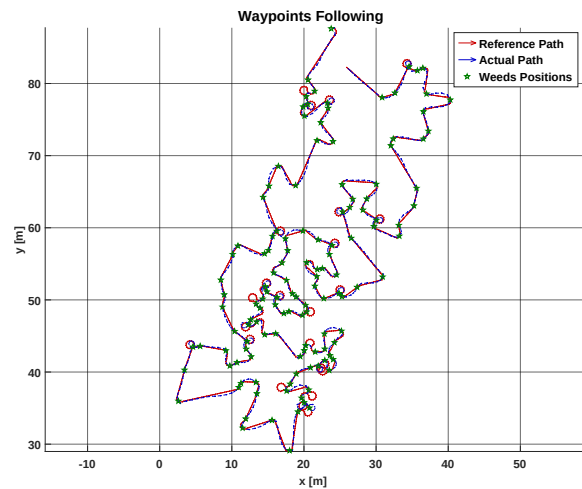
Future research will focus on enhancing local motion planning by considering complex obstacle scenarios, including moving humans, animals, other robots and machinery into the NMPC's OCP formulation for safe, real-time adaptation to moving agents. Experimental field validation is planned under varying terrain conditions to address challenges arising from process and measurements noise, bridging the gap between simulation and practical agricultural robotics.

REFERENCES

- [1] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, Mar. 2019.
- [2] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model Predictive Contouring Control for Collision Avoidance in Unstructured Dynamic Environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4459–4466, Oct. 2019.
- [3] L. E. Dubins, "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.



(a) Reference path generated using the proposed coupled formulation of the DTSP-based planner with 10 angle discretization levels. This method eliminates loops and provides a more efficient path.



(b) Reference path generated using a decoupled formulation of the DTSP-based planner with the Alternating Algorithm for angle selection. This approach results in suboptimal paths with redundant circular loops.

Fig. 1: Comparison of Dubins tours (red arrows) for approximately $20 \text{ m} \times 60 \text{ m}$ field containing 150 target weeds (green stars), with a vehicle turning radius of 0.5 m. The proposed coupled DTSP-based planner (a) chooses a different order of the waypoints, thereby allowing for a smoother path, whereas the decoupled DTSP-based planner (b) results in a less optimal path with redundant loops. In both cases, the NMPC closed-loop state trajectory (blue arrows) successfully reaches all waypoints while locally optimizing motion by smoothing sharp turns and taking efficient shortcuts when beneficial.

- [4] T. Faulwasser, B. Kern, and R. Findeisen, "Model predictive path-following for constrained nonlinear systems," in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference*, Dec. 2009, pp. 8642–8647.
- [5] A. M. Frieze, G. Galbati, and F. Maffioli, "On the worst-case performance of some algorithms for the asymmetric traveling salesman problem," *Networks*, vol. 12, no. 1, pp. 23–39, 1982.
- [6] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control*, ser. Communications and Control Engineering. Cham: Springer International Publishing, 2017.
- [7] K. Helsgaun, "An effective implementation of the Lin–Kernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106–130, Oct. 2000.
- [8] H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko, "Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs," in *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, Oct. 2003, pp. 56–65.
- [9] J. Le Ny, "Performance optimization for unmanned vehicle systems," Thesis, Massachusetts Institute of Technology, 2008.
- [10] J. Le Ny and E. Feron, "An Approximation Algorithm for the Curvature-Constrained Traveling Salesman Problem," Oct. 2010.
- [11] M. Lorenzen, T. Alamo, M. Mammarella, and F. Dabbene, "MPC-based motion planning for non-holonomic systems in non-convex domains," in *European Control Conferences*, Thessaloniki, Greece, June 2025.
- [12] X. Ma and D. A. Castanon, "Receding Horizon Planning for Dubins Traveling Salesman Problems," in *Proceedings of the 45th IEEE Conference on Decision and Control*, Dec. 2006, pp. 5453–5458.
- [13] D. G. Macharet and M. F. M. Campos, "A survey on routing problems and robotic systems," *Robotica*, vol. 36, no. 12, pp. 1781–1803, Dec. 2018.
- [14] D. G. Macharet, A. A. Neto, V. F. da Camara Neto, and M. F. M. Campos, "Data gathering tour optimization for Dubins' vehicles," in *2012 IEEE Congress on Evolutionary Computation*, June 2012.
- [15] M. Mammarella, L. Comba, A. Biglia, F. Dabbene, and P. Gay, "Cooperation of unmanned systems for agricultural applications: A theoretical framework," *Biosystems Engineering*, vol. 223, pp. 61–80, Nov. 2022.
- [16] M. W. Mehrez, K. Worthmann, G. K. Mann, R. G. Gosine, and T. Faulwasser, "Predictive Path Following of Mobile Robots without Terminal Stabilizing Constraints," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9852–9857, July 2017.
- [17] J. Ny, E. Feron, and E. Frazzoli, "On the Dubins Traveling Salesman Problem," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 265–270, Jan. 2012.
- [18] S. Rathinam, R. Sengupta, and S. Darbha, "A Resource Allocation Algorithm for Multivehicle Systems With Nonholonomic Constraints," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 98–104, Jan. 2007.
- [19] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Madison, Wisconsin: Nob Hill Publishing, 2017.
- [20] K. Savla, E. Frazzoli, and F. Bullo, "On the point-to-point and traveling salesperson problems for Dubins' vehicle," in *Proceedings of the 2005, American Control Conference, 2005*. Portland, OR, USA: IEEE, 2005, pp. 786–791.
- [21] R. Siegwart, *Introduction to Autonomous Mobile Robots*, ser. Intelligent Robots and Autonomous Agents. Cambridge, MA: MIT Press, 2004.
- [22] R. Soloperto, J. Köhler, and F. Allgöwer, "A Nonlinear MPC Scheme for Output Tracking Without Terminal Ingredients," *IEEE Transactions on Automatic Control*, vol. 68, no. 4, pp. 2368–2375, Apr. 2023.
- [23] P. Váňa and J. Faigl, "Optimal solution of the Generalized Dubins Interval Problem: Finding the shortest curvature-constrained path through a set of regions," *Autonomous Robots*, vol. 44, no. 7, pp. 1359–1376, Sept. 2020.
- [24] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar. 2006.
- [25] S. Yu, X. Li, H. Chen, and F. Allgöwer, "Nonlinear model predictive control for path following problems," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 8, pp. 1168–1182, 2015.

Comparison of neural networks road detection in off-road environments

Jakob Oberpertinger¹, Matthias Eder¹ and Gerald Steinbauer-Wagner¹

Abstract—As unmanned ground vehicles (UGVs) are more frequently deployed in unstructured environments, there is a growing need for robust road and terrain detection systems. The ability to navigate autonomously in challenging terrains depends on the effectiveness of computer vision models.

Off-road environments encompass rugged terrain, forest roads, agricultural fields, and more, characterized by dynamic changes and unpredictable obstacles. UGVs must discern drivable ground to enable effective navigation while identifying and circumventing obstacles in real-time.

This paper investigates different sensor-based and neural network-driven approaches to address these challenges, focusing on the critical task of identifying forest roads in off-road environments. Using different sensors, we assess their effectiveness in different environmental conditions through a comprehensive comparative analysis of three neural network architectures. Our results highlight the strengths and limitations of different sensor modalities and neural network models. They provide insight into their performance under adverse conditions such as overexposed images, complex shadows, and dense vegetation on forest roads. This research provides valuable insights into developing robust off-road navigation systems essential for advancing autonomous ground vehicle technology.

I. INTRODUCTION

New application areas for unmanned ground vehicles (UGV), such as disaster response or forestry, have led to the need for safe navigation both on and off the road. Effective navigation is essential; it requires not only the ability to identify clear and accessible routes but also the foresight to avoid potential obstacles. Mastering this skill enhances safety and ensures a smoother journey every time. However, research in unstructured environments still lags behind that in structured environments [9]. Off-road environments for unmanned ground vehicles (UGVs) present unique challenges compared to traditional on-road environments. These environments vary widely, including rugged terrain, forest roads, agricultural areas, etc. Off-road environments can experience rapid changes, such as the appearance of lighting and weather conditions, temporary obstacles, or changes in terrain conditions. UGVs must be able to adapt in real-time to meet these dynamic challenges. The absence of clearly defined routes makes it challenging for an unmanned ground vehicle (UGV) to navigate to its destination. For a UGV to find its destination, it must make two important decisions. Firstly, it needs to detect accessible routes that it can safely traverse. Secondly, it must detect obstacles so that it can

safely avoid them. There are many different approaches to solving these two challenges, using different sensors or neural network architectures.

Our research focuses on identifying navigable terrain in off-road environments, essential for safe and efficient navigation in unknown terrain. To address this challenge, we are undertaking a comprehensive comparison of three different neural network architectures using a variety of sensors, including RGB and depth images from stereo cameras and point clouds from lidar sensors. By exploring the effectiveness of different sensors, we aim to identify their respective strengths and limitations. This investigation goes beyond pure theoretical analysis, as we are carefully testing the limits of these networks under harsh environmental conditions. These conditions are characterized by significant challenges such as fluctuating sunlight, complex shadow patterns, and dense vegetation. The interplay of sunlight and shadows poses a significant hurdle for camera sensors and neural networks, especially if not adequately trained. In addition, vegetation poses challenges. Forest roads exhibit patches of grass in the center, which complicates the identification of navigable pathways.

The remainder of this paper is structured as follows: Section II discusses current research topics in path detection in off-road environments. Section III presents the three different neural networks evaluated in Section IV. Section V concludes the paper.

II. RELATED RESEARCH

A. Methods

Unmanned Ground Vehicles (UGVs) operating in off-road environments require robust road detection systems for safe and efficient navigation. Recent advancements in neural networks have significantly improved off-road path detection capabilities. However, developing a reliable and stable network for this purpose and selecting the appropriate sensors poses notable challenges. Ilas [8] outlines the key sensor technologies UGVs use to make real-time decisions while monitoring their surroundings. The study explores the various sensors employed across different environments and vehicle prototypes, evaluating the advancements in sensor technology.

Another important technology in off-road road detection is Convolutional Neural Networks (CNNs). CNNs excel in capturing spatial hierarchies of features, making them well-suited for image-based tasks. Researchers have explored various CNN architectures tailored for off-road scenarios. The work of Holder et al. [7] focuses on transfer learning, taking a pre-trained CNN designed for urban road scenes

¹Jakob Oberpertinger, Matthias Eder, and Gerald Steinbauer-Wagner are with the Institute of Software Technology, Graz University of Technology, Graz, Austria. {jakob.oberpertinger, matthias.eder, steinbauer}@tugraz.at

and retraining it to classify off-road scenes. The analysis involves assessing the network performance during various stages of training and exploring different levels of prior training on subsets of off-road data. The study compares the CNN approach with a traditional feature-driven Support Vector Machine (SVM) classifier, demonstrating state-of-the-art results in the challenging problem of off-road scene understanding.

Neural Networks using Lidar data have become a significant advancement in off-road road detection, offering depth information that allows for a more nuanced understanding of the environment. Zhong et al. [14] present a method known as LRTI, designed for identifying drivable areas in challenging off-road scenes. The complexity of this task arises from unstructured class boundaries, irregular features, and noise. By leveraging three-dimensional LiDAR data and a bird's eye view (BEV) perspective, LRTI utilizes texture information derived from LiDAR reflection data. The method incorporates an instance segmentation network to effectively learn this texture information, facilitating the identification of drivable areas. A multi-frame fusion strategy is employed to improve reliability. LRTI successfully achieves real-time processing on unmanned ground vehicles (UGVs).

Nate Haddad [5] discusses the challenges of training large deep learning algorithms due to the need for a substantial training dataset and computing power. Transfer learning, a method of transferring knowledge from one domain to another, is introduced as a solution to avoid training from scratch. The focus is on applying transfer learning to large encoder-decoder-style deep neural networks, specifically examining its impact on semantic segmentation tasks. DeepLabv3+, a state-of-the-art architecture from 2018, is highlighted for its efficiency in incorporating techniques from the 2016 Xception model [4].

B. Datasets

Chen Min et al. introduce the first off-road freespace detection dataset, called the ORFD dataset. Recognizing the importance of free space detection in autonomous driving technology, the authors highlight the limitations of existing deep learning methods, which primarily focus on urban road environments. To address this gap, they present the ORFD dataset, comprising 12,198 LiDAR point clouds and RGB image pairs collected in various off-road scenes, weather conditions, and light conditions. The authors propose a novel neural network, OFF-Net, which utilizes a transformer architecture to integrate local and global information, catering to the needs of a large receptive field for free space detection.

Peng et al. [10] address the significance of semantic scene understanding for robust autonomous navigation, particularly in off-road environments. Acknowledging the reliance of recent 3D semantic segmentation advancements on extensive training data, the authors identify a gap in existing datasets, which are either urban-focused or lack multimodal off-road data. The authors introduce RELLIS-3D, a multimodal dataset collected in an off-road setting to bridge this gap. The paper evaluates state-of-the-art deep learning semantic seg-

mentation models on RELLIS-3D, revealing that the dataset introduces challenges distinct from urban environments.

The RUGD dataset [13] provides semantic annotations for unstructured outdoor environments, supporting off-road autonomous navigation. The dataset from a mobile robot platform includes video sequences with dense pixel-wise annotations for terrain classification and obstacle detection. It features 24 semantic categories, including eight terrain types, to enhance path planning and localization in environments lacking structured cues.

III. EVALUATED ARCHITECTURES

In this chapter, we evaluate three previously published neural network architectures, selected for their diverse input modalities and relevance to understanding the off-road scene. Our aim is not to propose new architectures, but to assess how well existing state-of-the-art segmentation methods generalize to off-road environments, particularly in challenging conditions such as forest roads, uneven terrain, and under-exposed regions. The motivation behind the selection of these three models is based on their complementary input representations and processing strategies:

- **OFF-Net:** Chosen for using surface normal maps and a transformer-based architecture, offering a high-level representation of terrain structure. It is designed to leverage geometric cues from RGB-D input for improved scene segmentation.
- **DeepLabV3+:** A well-established CNN-based model known for its high segmentation accuracy and strong performance across various domains. It is particularly beneficial when working with limited or domain-specific training data.
- **SalsaNext:** A LiDAR-based semantic segmentation model operating directly on 3D point clouds. Its selection allows us to evaluate how pure LiDAR-based perception compares to image-based methods in unstructured off-road scenes.

This comparative evaluation's significance lies in understanding these architectures' behavior under real-world deployment constraints. By testing on our dataset, comprising RGB imagery, stereo-derived depth, and LiDAR scans collected in diverse environments, we aim to provide practical insight into each network's robustness and adaptability. This evaluation not only identifies the performance boundaries of each modality but also informs future design decisions for autonomous navigation systems in GNSS-denied and visually ambiguous terrain.

In the following section, we will present the concept of the comparison between the three neural networks, which are:

- Off-Road-Freespace-Detection (ORFD)
- DeepLabv3+
- SalsaNext using RELLIS-3D dataset

The three architectures and their design are presented in this chapter in detail.

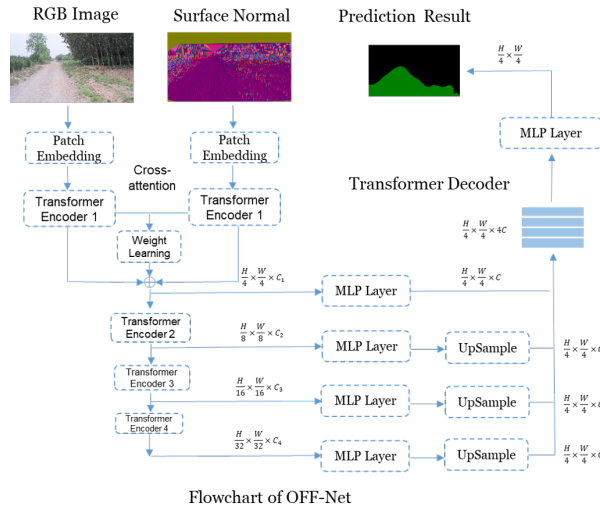


Fig. 1: The architecture of the OFF-Net [12].

A. Off-Road-Freespace-Detection (ORFD)

This architecture was presented by Chen Min et al. [12] 2022, which addresses the critical aspect of free space detection in off-road environments for autonomous driving. The paper presents a novel neural network, OFF-Net, which uses a transformer architecture to integrate local and global information, addressing the need for expansive receptive fields in free-space detection tasks, which are critical for accurate detection. Figure 1 shows an overview of the presented OFF-Net. As can be seen in the figure, the network combines two pieces of information: the RGB image and the corresponding surface normal. The paper's authors use LiDAR point cloud information to calculate the surface normal for each image. In our case, we calculate the surface normal from a dense depth image provided by the ZED2 stereo camera¹. The transformer encoder can extract the features from these two pieces of information, and the transformer decoder predicts the free space. The paper also presents the dataset they have created for off-road freespace detection, called the ORFD dataset. The dataset includes off-road environments such as forests, farmland, and countryside with different weather conditions. The results demonstrate that SNE-RoadSeg, utilizing surface normals instead of depth information, outperforms FuseNet in free space detection. Furthermore, the newly proposed OFF-Net achieves even higher accuracy, surpassing FuseNet by 10.8% in F-score and 16.3% in mIOU. OFF-Net, employing the Transformer framework, efficiently captures local and global information while maintaining real-time processing capabilities, 7 times smaller and 2.7 times faster than SNE-RoadSeg [12].

B. DeepLabV3+

The second architecture, DeepLabv3+, is a simple but effective decoder module to improve segmentation results.

¹<https://www.stereolabs.com/docs>

Chen et. al. [1] describes this architecture as follows: Multiple downsampling of CNN results in a smaller feature map resolution, which leads to lower prediction accuracy and loss of boundary information in semantic segmentation. Similarly, aggregating the context around a feature helps to better segment it, which is achieved with sparse convolutions. DeepLabv3+ helps to solve these problems. The architecture can be seen in Figure 2. To save time and in the absence of a large dataset, we used a pre-trained model from the paper by Nate Haddad [5], who proposes to extend the application of a pre-trained DeepLabv3+ model to the challenging domain of off-road perception. The authors successfully employ transfer learning techniques using the Yamaha-CMU Off-Road Dataset for semantic segmentation of off-road images, showcasing the model's adaptability and effectiveness in a different domain. The Yamaha-CMU Off-Road Dataset [11] consists of 1076 images collected in different environments using three different sensors. It was labeled using eight classes (sky, rough trail, smooth trail, traversable grass, high vegetation, non-traversable low vegetation, and obstacle). The model takes an image as an input parameter, which is provided by the ZED2 stereo camera mounted on the front of the robot.

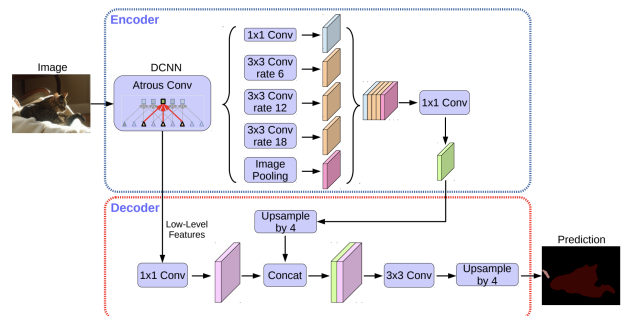


Fig. 2: The architecture of the Deeplabv3+ [1].

C. SalsaNext

Last, we used a model using LiDAR data as the input parameter. Peng et al. [10] introduced in their paper SalsaNext, an advanced model designed for real-time uncertainty-aware semantic segmentation of full 3D LiDAR point clouds. The authors made some major improvements to the already existing model SalsaNet. Some improvements are as follows: they replaced the ResNet encoder blocks with a new residual dilated convolution stack with gradually increasing receptive fields and added the pixel-shuffle layer in the decoder. Finally, we implemented a model that utilizes LiDAR data as its input parameter. In their paper, Peng et al. [10] introduced SalsaNext, an advanced framework designed for real-time, uncertainty-aware semantic segmentation of complete 3D LiDAR point clouds. The authors made significant enhancements to the existing SalsaNet model. Notable improvements include the substitution of the ResNet encoder blocks with a novel residual dilated convolution stack that features progressively increasing receptive fields

and incorporates a pixel-shuffle layer in the decoder. They also switch from stride convolution to average pooling and apply central dropout treatment. To directly optimize the Jaccard index, they combine the weighted cross-entropy loss with Lovasz-Softmax loss and inject a Bayesian treatment to compute the epistemic and aleatoric uncertainties for each point in the cloud [2]. The improved architecture can be seen in Figure 3. The authors of the paper [10] present the dataset RELLIS-3D, a collection of off-road environments captured at the Rellis Campus of Texas A&M University. The RELLIS-3D dataset comprises a large set of raw sensor data, including color camera images, laser scans, high-precision global positioning measurements, inertial measurements, and depth images from a 3D stereo camera, and is labeled in 20 classes. The results show that SalsaNext achieves a higher mIoU of 43.07% compared to KPConv’s 19.07%, which is significantly lower than their performance on the SemanticKITTI dataset, which was 59.5% mIoU and 58.8%, respectively. The imbalance in the point cloud dataset poses a significant challenge for both algorithms, with KPConv showing a more pronounced degradation. Despite attempts to mitigate the imbalance through sampling strategies during training, such efforts only marginally improved the results by 0.6% mIoU [10].

As the classes did not include forest roads, we selected a subset of the 20 available classes, focusing only on those relevant to detecting passable ground. This subset includes dirt, grass, puddles, asphalt, and mud.

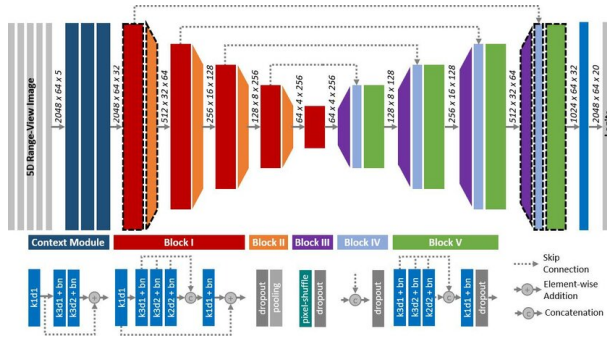


Fig. 3: The architecture of the SalsaNext [2].

After implementing, we conducted rigorous testing for various environments and sensors. The following chapter describes the results and evaluations of these tests in detail.

IV. EVALUATION

Autonomous navigation in off-road scenarios presents unique challenges that demand robust and accurate perception systems.

A. Data Generation

To evaluate the three networks and generate test data, we are utilizing the robots, Mercator [6], developed by Graz University of Technology, and Husky², developed by Clearpath.

²<https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/>

Mercator is a universal off-road platform developed for autonomous navigation in disaster response scenarios. It is a four-wheeled mobile platform with double Ackermann steering, an onboard computer, and a mounting frame for various sensor setups. Husky is a medium-sized robotic development platform with a large payload capacity. It is a customizable robot with the ability to add multiple sensors. The assessment spanned diverse environments, ranging from optimal visibility forest roads to challenging off-road terrains covered with grass.

To collect and record data for analysis, we equipped the two unmanned robots mentioned above with a ZED2 stereo camera³ and 3D LiDAR scanners. Our data collection spanned a variety of environments and locations, including mountainous areas, rural landscapes, and forest roads in Styria, Austria, capturing different weather and terrain conditions. We selected challenging scenarios from the collected data for network testing, including varying light conditions, narrow forest roads, off-road paths with grass tracks, and grass-covered terrain, as shown in Figure 4. The ground truth annotation of the data was conducted manually.

B. Network Performance Metrics:

To evaluate the three different models, we have used the widely used mean intersection-over-union (mIoU) metric [3], which is given by

$$mIoU = \frac{1}{C} \sum_{c=1}^C \frac{TP_c}{TP_c + FP_c + FN_c} \quad (1)$$

where C is the number of classes, and TP (=true positive), FP (=false positive) and FN (=false negative) are the predictions for class c. The analysis focused on two classes: traversable and non-traversable areas.

C. Quantitative Results

We chose 250 images from our generated data for a quantitative analysis, described in IV-A. This ensures a well-distributed selection that captures key challenges such as lighting conditions, vegetation, and shadows.

Table I outlines the mIoU rates for each network for each image shown in Figure 4 and the mIoU (=mean IOU). Notably, on the mIoU, the DeepLabV3+ model outperformed OFF-Net by 9.64%, despite OFF-Net utilizing the surface normal as additional information. The SalsaNext network achieved a mIoU rate of 27.86%, emphasizing its ability to distinguish between traversable and non-traversable areas.

	Reference	Green Strip	Shadow	Underexposed	mIoU
DeepLabV3+	95.11%	51.55%	53.29%	1.53%	76.91%
OFF-Net	73.60%	29.25%	50.60%	0.12%	67.27%
SalsaNext	30.82%	28.28%	35.53%	19.87	27.86%

TABLE I: mIoU of the three neural networks.

³<https://www.stereolabs.com/docs>

D. Comparative Analysis:

Despite OFF-Net incorporating additional information, the DeepLabV3+ model outperformed it. This raises questions about the effectiveness of the extra data and underscores the importance of careful feature selection and integration. Figure 4 shows the difference between the two networks using camera information, in which the second column shows the ground truth in light green, the third column shows the prediction of the DeepLabV3+ model in blue, and finally, the last column shows the prediction of the OFF-Net network in dark green.

The four different scenarios visualize the main problems and limitations of the two networks. The first scenario (reference) shows a well-visible, clear, and wide forest road, which both networks can predict quite well, with both mIOU values higher than 70%, as shown in Table I. The next scenario (green strip) shows an off-road divided by a grass strip. Here, both networks have difficulty accurately delineating the entire road and only manage to identify segments without grass. Again, the DeepLabV3+ scores a higher mIOU value compared to the OFF-Net.

The third scenario (shadow) shows a narrow forest path in a partially shaded wooded area. The OFF-Net has difficulty distinguishing between shaded and sunlit areas. However, DeepLabV3+ shows superior performance in this respect, suggesting that the OFF-Net model could be improved by refining the training dataset. DeepLabV3+ detects areas at the side of the path, which can lead to difficult or impassable paths. If we look at the mIOU values from Table I, we can see that DeepLabV3+ has a slightly higher mIOU value, but if we look at the images, OFF-Net is more accurate on the path. Last but not least, a road is completely covered with grass, which neither network can predict. Both networks have an mIOU value lower than 2%. It shows the networks are not trained for this type of off-road.

E. Insights into SalsaNext Network:

While SalsaNext demonstrated its ability to distinguish between drivable surfaces such as grass, dirt, and bush, ... its limitation lies in its lack of specificity in identifying true off-road. As a result, it is not a good choice for off-road detection and, therefore, scores the worst mIOU values. Future improvements could focus on refining the training data to include a wider range of off-road surfaces, thereby improving its ability to make nuanced distinctions. Figure 5 shows the predicted point cloud for different environments. The first environment is a wide forest road; the second is a narrow forest path.

F. Challenges and Solutions for OFF-Net:

OFF-Net faced challenges related to sun reflection and shadows, impacting its predictions. Bright reflections and rapid changes in brightness, especially transitioning from shadows to sunlight, were identified as major concerns. Moreover, the network can be improved by adding more difficult scenarios to the training data, such as underexposure,

forest roads divided by grass strips, or fully covered roads with grass.

V. CONCLUSION

This paper evaluated three neural networks—DeepLabV3+, OFF-Net, and SalsaNext—for autonomous navigation in off-road environments using the Mercator robot. Tests covered forest paths, narrow trails, and grass-covered terrain, highlighting each model's strengths and limitations.

Future work should improve SalsaNext's training data and improve OFF-Net through adaptive mechanisms or filtering. These insights support further optimization of network robustness for real-world off-road navigation.

ACKNOWLEDGMENT

This work was funded by the Austrian Research Funding Association (FFG) under the scope of the THINK.WOOD.INNOVATION program.

REFERENCES

- [1] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," arXiv, 2018.
- [2] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving," arXiv, 2020.
- [3] M. Everingham, S. Eslami, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, pp. 98–136, Jan. 2015.
- [4] R. Garg, A. Kumar, N. Bansal, M. Prateek, and S. Singh, "Semantic segmentation of polsar image data using advanced deep learning model," *Scientific Reports*, vol. 11, pp. 15365:1–18, 07 2021.
- [5] N. Haddad, "Semantic segmentation of off-road images using transfer learning and deeplabv3+," <https://github.com/nmhaddad/semantic-segmentation/tree/master>, 2022.
- [6] R. Halatschek, K. Ramanna, W. Url, and G. Steinbauer-Wagner, "Universal offroad robot platform for disaster response," in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 11 2020, p. 6.
- [7] C. J. Holder, T. P. Breckon, and X. Wei, "From on-road to off: Transfer learning within a deep convolutional neural network for segmentation and classification of off-road scenes," in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds. Cham: Springer International Publishing, 2016, pp. 149–162.
- [8] C. Ilaş, "Electronic sensing technologies for autonomous ground vehicles: A review," in *2013 8TH INTERNATIONAL SYMPOSIUM ON ADVANCED TOPICS IN ELECTRICAL ENGINEERING (ATEE)*, 2013, pp. 1–6.
- [9] F. Islam, M. M. Nabi, and J. E. Ball, "Off-road detection analysis for autonomous ground vehicles: A review," *Sensors*, vol. 22, no. 21, 2022. [Online]. Available: <https://www.mdpi.com/1424-8220/22/21/8463>
- [10] P. Jiang, P. R. Osteen, M. B. Wigness, and S. Saripalli, "RELLIS-3D dataset: Data, benchmarks and analysis," *CoRR*, vol. abs/2011.12954, 2020. [Online]. Available: <https://arxiv.org/abs/2011.12954>
- [11] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, "Real-time semantic mapping for autonomous off-road navigation," in *Field and Service Robotics*. Springer, 2018, pp. 335–350.
- [12] C. Min, W. Jiang, D. Zhao, J. Xu, L. Xiao, Y. Nie, and B. Dai, "Orfd: A dataset and benchmark for off-road freespace detection," 2022.
- [13] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, "A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments," in *International Conference on Intelligent Robots and Systems (IROS)*, 2019.
- [14] C. Zhong, B. Li, and T. Wu, "Off-road drivable area detection: A learning-based approach exploiting lidar reflection texture information," *Remote Sensing*, vol. 15, no. 1, 2023. [Online]. Available: <https://www.mdpi.com/2072-4292/15/1/27>

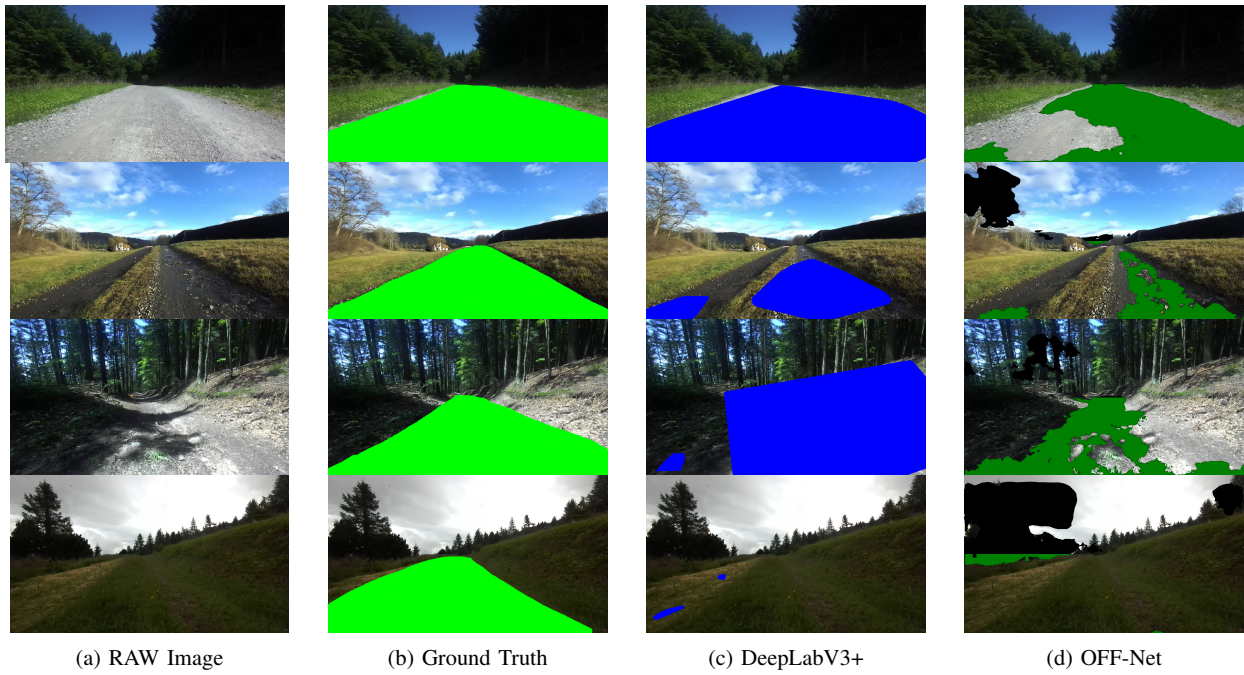


Fig. 4: Predictions of the image based networks DeepLabV3+ and OFF-Net.

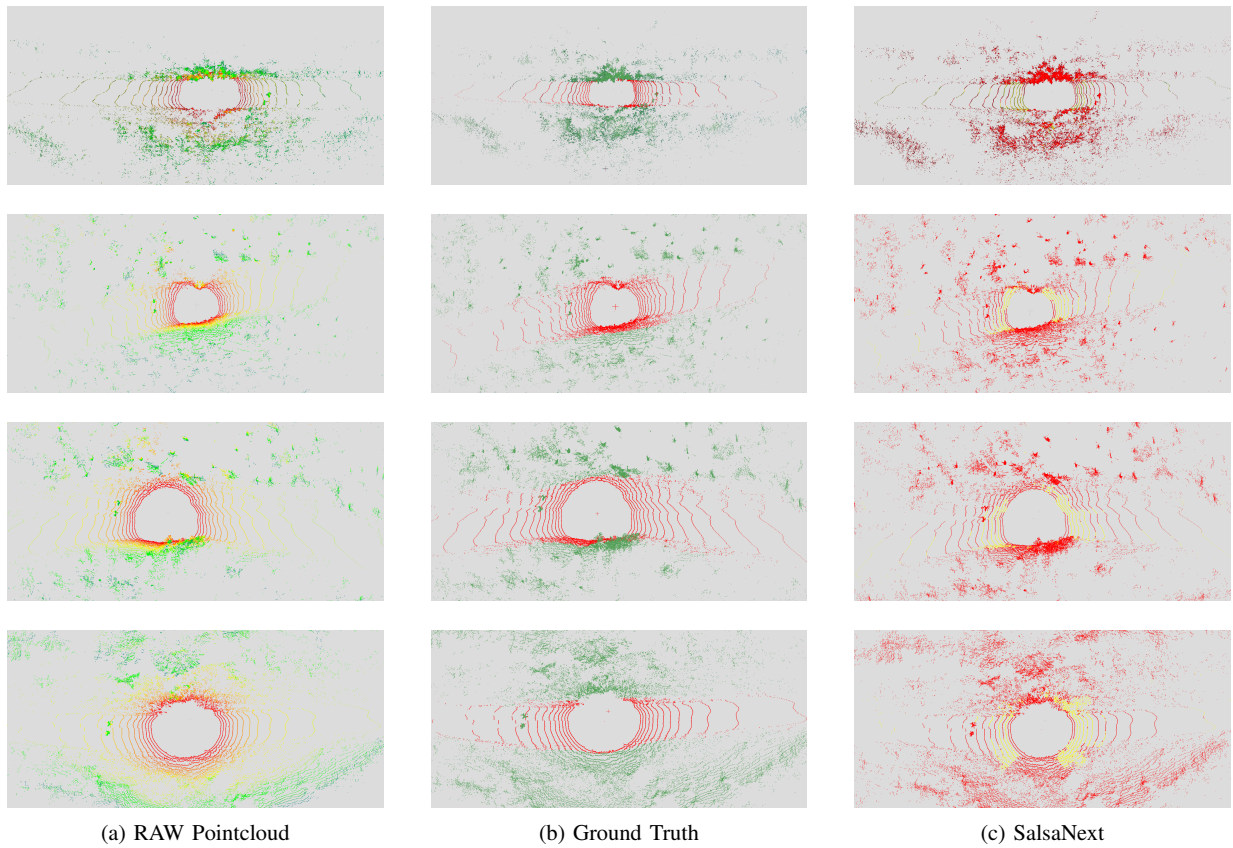


Fig. 5: SalsaNext Network prediction

Multi Robot Route Planning for ROS2

Matthias Reicher¹ and Markus Bader¹

Abstract—This work presents the implementation of a multi robot route planner based on the prioritized planning approach as well as its integration into ROS2 and the well-known Nav2 stack. Further, a method to increase the resilience towards uncertainty and unpredictability in timing during the execution of found routes is introduced. These so-called routing preconditions are shown to be effective on a subset of routing scenarios and offer significant opportunity for further exploration.

Index Terms—multi robot system, path planning, ROS2, Nav2

I. INTRODUCTION

To leverage the advantages of a multi-robot system (MRS), large fleets of mobile robots must be able to effectively compute routes from one point in the environment to another without risking collision. This makes multi-robot-route-planning a fundamental problem for MRS, as it lays the groundwork for more complex behavior [3]. Many approaches to solving this problem have been discussed in the literature, with so-called “prioritized planning” appearing in a significant number of publications [2]. However, up to current knowledge, no publicly available ROS2-compatible software packages provides an easy integration of such functionality. This work aims to close the identified gap, similar to the previous work of [1] on ROS, but by taking advantage of the advanced capabilities offered by the well-known Nav2 stack. Results are presented by using a simulated environment as shown in Fig. 1.

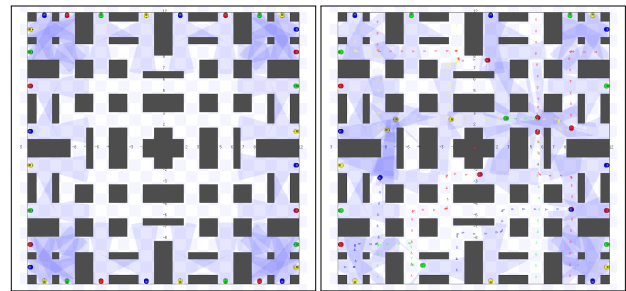
II. PRIORITIZED PLANNING

Prioritized Planning refers to the practice of decomposing the multi-robot-route-planning problem into a series of single-robot-route-planning (SRRP) problems. Each of the SRRP-problems concerns itself with finding a collision-free route for an individual robot and must take static obstacles as well as robots for which a route has already been found into consideration. Since routes are planned in descending order according to some priority metric, higher-priority robots represent dynamic obstacles in the planning space of low priority robots.

III. IMPLEMENTED PLANNING ALGORITHM

To realize this specification of a Prioritized Planner, some considerations need to be made: First, a planning algorithm which is able to handle dynamic obstacles is required to solve the individual SRRP-problems. Second, the routes generated by the prioritized planner need to be suited for execution by a real MRS.

¹The authors are with Faculty Informatics at TU Wien, Vienna, Austria.
firstname.lastname@tuwien.ac.at



(a) Initial position (b) During navigation with Nav2

Fig. 1: Stage-simulation of a 32-robot MRS.

A. Sequential Planner

The chosen planning algorithm can be described as a variant of the spatio-temporal A*-Algorithm introduced in [4] operating on a graph-based abstraction of the environment. This abstraction is able to emulate 4/8-connected grid maps, as well as higher level concepts such as voronoi graphs with multi-edges. The key difference to the well-known A*-Algorithm is given by additional occupancy checks whenever a graph vertex is explored and added to the frontier: should it be occupied by another robot at the point in time in which the planning robot expects to enter, time must be spent waiting earlier along the currently considered route. If it is impossible to insert this waiting time at some point along the path without risking collisions, the proposed node is not marked for further exploration. These iterative planning processes result in a detailed record describing at which points in time any particular graph vertex is expected to be occupied by a robot if no unexpected delays occur.

B. Route Representation

After planning an ideal path for a robot in the system, post-processing is done to create a route suited for execution by a real MRS. Routes consist of a series of indexed route segments, each describing a move from one vertex of the graph to one of its neighbors. In addition to the timestamps during which this move is expected to take place, a set of preconditions for the segment is generated by considering all other robots scheduled to pass the destination of the move before it occurs. A precondition is considered to be satisfied as soon as the robot it is referencing has completed the noted segment of its own route (i.e. it has passed through the vertex at which both routes cross). This creates clear precedence relations, which serve to improve the systems resilience towards neglected or unexpected delays during navigation.

IV. ROS2 INTEGRATION

The ROS2 integration of the implemented planner is split between multiple communicating system components, pictured in Fig. 2.

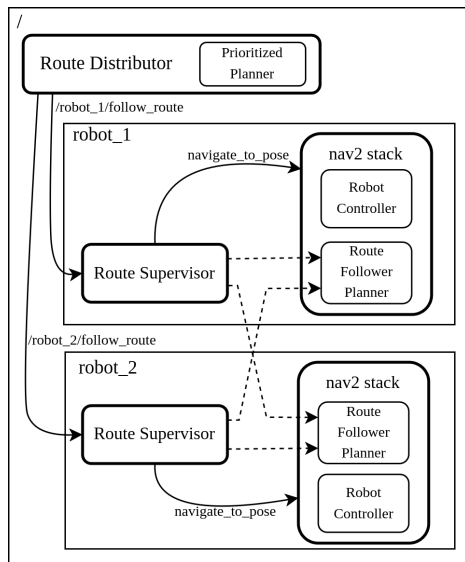


Fig. 2: Architecture of the ROS2 integration.

A. Route Distributor

The Route Distributor node acts as the central coordinator of the MRS. It is responsible for initializing navigation by generating each robots route using the implemented prioritized planning algorithm and distributing them among the MRS using ROS actions. During route execution, it monitors the received feedback and aborts navigation should unexpected issues arise.

B. Route Supervisor

The communication between robots and the Route Distributor is handled by an individual Route Supervisor node for every robot. Each of these nodes also monitors the robots progress along its own route and publishes this information for consumption by all the Route Followers in the system. This enables robots to wait on unsatisfied preconditions in order to avoid situations not considered during planning.

C. Route Follower

To enable the use of the wide variety of localization strategies, local planners and other software components available within Nav2, the system integrates with a Nav2-planner-plugin known as the Route Follower.

V. EVALUATION

The implemented planning algorithm was tested on randomly generated routing problems featuring 8-32 robots concurrently attempting to find a route through a heavily restricted warehouse-like environment. Through varying the order in which routes are planned, a solution to each of these routing problems was found. The systems capability

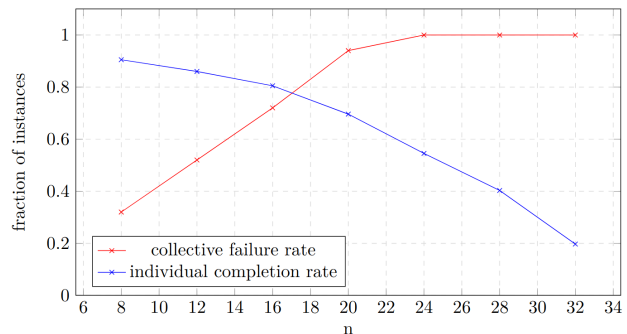


Fig. 3: Routing success in a highly constrained environment.

of executing these found routes was then evaluated by simulating navigation using the Stage simulator.

Fig. 3 depicts the ratio of individual robots which were able to reach their goals as well as the chance of any robot failing to finish its route due to an emergency stop, a collision or similar reasons. Both metrics behave in a roughly linear fashion, resulting in sharply degrading reliability as more concurrently navigating robots are added to the system.

Two central causes for these failures were identified:

- 1) Off-the-shelf Nav2 local planner solutions navigating based on a generic path representation deviating from the strictly defined pre-planned routes.
- 2) Endless waiting on an unsatisfied precondition referring to a stuck robot causing cascading failure in the system.

VI. SUMMARY AND OUTLOOK

Collision-free routes for members of a multi-robot systems can be found by the implemented algorithm, but it is evident that this does not guarantee that these routes can be executed without issue in realistic conditions. While routing preconditions were introduced to counteract timing-related failures, they have proven insufficient to avoid them entirely without addressing flaws in the systems architecture and implementation. Introducing additional mechanisms to increase robustness such as on-line re-planning in case of a detected deadlock represents another avenue for future work.

VII. ACKNOWLEDGMENT

This research is supported by the Austrian Science Fund (FWF) under project No. 923138, GreenFDT.

REFERENCES

- [1] B. Binder, F. Beck, F. König, and M. Bader, "Multi Robot Route Planning (MRRP): Extended Spatial-Temporal Prioritized Planning," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019, pp. 4133–4139.
- [2] J. Heselden and G. Das, "Heuristics and rescheduling in prioritised multi-robot path planning: A literature review," *Machines*, vol. 11, no. 11, p. 1033, 2023.
- [3] G. Kyprianou, L. Doitsidis, and S. A. Chatzichristofis, "Towards the achievement of path planning with multi-robot systems in dynamic environments," *J. Intell. Robot. Syst.*, vol. 104, no. 1, 2022.
- [4] W. Wang and W.-B. Goh, "Multi-robot path planning with the spatio-temporal A* algorithm and its variants," in *Advanced Agent Technology*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 313–329.

ROS with LEGO Spike

Jakob Buchsteiner¹, Daniel Marth², Moritz Taferner¹ and Markus Bader¹

Abstract—Teaching mobile robotics algorithms through hands-on hardware exercises can be both costly and resource-intensive. This work addresses this challenge by introducing an affordable differential drive vehicle constructed from LEGO components. An onboard Raspberry Pi, equipped with a camera and a Build HAT, provides standard ROS2 interfaces. An outstanding feature of the design is the calculation of laser ranger data from camera images, which enables the investigation of sensor and motion models, as well as probabilistic approaches for self-localization and mapping. The paper presents a prototype together with statistical results on the motion and sensor models within the real and simulated environment.

Index Terms—ROS2, Mobile Robot, Self-Localization

I. INTRODUCTION

Robot Operating System (ROS) plays a major role in the growing field of robotics, especially in education, such as teaching mobile robotics. Integrating affordable hardware with a software platform like ROS enables undergraduate students to gain hands-on experience in robotics.

This paper explores the possibilities of integrating the Lego Spike PRIME robotics kit into the latest version of ROS2 [1], utilizing a Raspberry Pi 4 single-board computer. For tight integration with the ROS2 ecosystem we employ pre-existing components such as *ros_control* [2], the default ROS2 implementation of AMCL (Adaptive Monte Carlo Localization) and the simulation tool Gazebo [3]. The experimental evaluation shows the viability of the presented approach as a base platform for simple localization tasks.

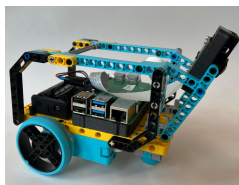


Fig. 1: The assembled robot used for the evaluations of this paper.

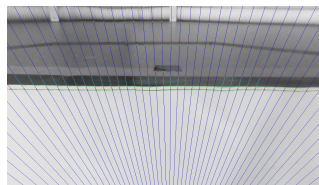


Fig. 2: Generating laser range data for localization using low-cost camera images.

II. RELATED WORK

Commercial platforms such as the *Turtlebot* [4], which was specifically developed for ROS, are also frequently used in education to practice hands-on mobile robotics.

*This work was not supported by any organization

¹Jakob Buchsteiner, Moritz Taferner and Dr. techn. Markus Bader (first.last@tuwien.ac.at) are with Faculty of Informatics, Vienna University of Technology.

²Daniel Marth (daniel.marth@tum.de) is with the Department of Computer Science, Technical University of Munich.

However, the acquisition costs play a very large role and therefore make it a less accessible option for institutions with limited resources. [5] also looks at using Lego Spike based robots in combination with ROS2 and Gazebo, however does not address localization. Our design fills this gap by extracting laser ranger data from camera images, allowing direct application of textbook algorithms like [6].

III. IMPLEMENTATION

To provide a base platform for teaching, we developed three primary components: hardware support, localization system using a camera and a simulation environment. The components draw their information from a shared robot description in the Unified Robot Description Format (URDF).

A. Hardware Support

One challenge of hardware integration is retaining reusability for different robot designs. Thus, we leverage existing motion controllers of the *ros2_control* framework, by providing the robot description and a plugin serving as a hardware abstraction layer for the Lego hardware. This layer controls the actuated wheels connected to the Raspberry Pi Build HAT, communicating using the documented serial protocol.

For evaluation, we use a differential drive platform with two independently driven wheels on each side of the robot and a caster wheel, allowing the robot to move in both linear and angular directions (Fig. 1). The differential drive controller included in *ros2_control* then translates motion commands to wheel velocities and performs odometry with the data from the wheel encoders.

B. Localization

Classic implementations of AMCL and SLAM (Simultaneous Localization and Mapping) operate on laser range data [6], however such sensors are expensive. We solve this issue by developing an intermediate layer, which extracts distance measurements from camera images using line markings on the floor.

Using the cameras intrinsic and extrinsic calibration parameters we construct a projective transform $H \in \mathbb{R}^{3 \times 3}$, mapping points on the ground plane to points on the camera plane. We determine lines in the image corresponding to radial lines around some ray center point on the ground plane. Along each ray, we apply a simple edge detection kernel, and estimate the width of the line using two pairs of line entry and exit points. After checking against the width threshold to isolate line markings from other line features, entry points are reported as the distance measurement in the respective direction (Fig. 2).

C. Simulation

Since many educational robotic tasks can be prototyped and evaluated in simulation, we set up a simulation environment to support development and preliminary testing. The same robot description used for hardware support is enriched by physics parameters specific to the simulator Gazebo. Most noteworthy are mass, rotational inertia and friction. We approximated inertia by dividing the robot into subcomponents, each of which were approximated as cuboids and cylinders with evenly distributed mass. The inertia calculation is then automatically performed by Gazebo. Friction parameters were manually tuned to prevent the robot from slipping in the simulation ($\mu_1 = \mu_2 = 1.0$).

IV. RESULTS

We evaluated and compared the vehicle's pose estimation using only the implemented motion model (odometry) and using AMCL with our emulated laser ranger data. Ground truth data was acquired from an OptiTrack motion tracking system.

A. Experimental Setup

The map used for evaluation is roughly a square with a side length of one meter and black, 5 cm thick tape markings used for localization. The robot is instructed to follow a figure-eight trajectory using open-loop control, and the trajectories estimated using odometry and AMCL are then compared against ground truth. Performance is analyzed in both simulation and real world environments according to [7].

B. Analysis

For a qualitative analysis of self-localization accuracy, we plot the trajectories for both simulation and real-world environments (Fig. 3). The convergence of the estimated AMCL trajectory towards the ground truth can be observed in Figs. 3a, 3b and 3d. We can also observe reasonably accurate odometry in Fig. 3c.

In simulation and without an offset in the initial pose estimate, the estimation using AMCL shows an absolute trajectory error of 9.6 mm, while the accuracy of the odometry trajectory has absolute trajectory errors above 50 mm.

Since [7] disregard offsets in the trajectory's starting pose, the numeric evaluation suggests that the estimated AMCL trajectories in the real world are worse than the raw odometry. However, Fig. 3d shows the improvement the localization system achieves: While the odometry can never match the ground truth trajectory, the AMCL particle filter relatively quickly converges to the correct position.

V. CONCLUSION

The robot's performance under real-world conditions demonstrated its potential as a suitable platform for teaching fundamental robotics concepts, such as navigation and localization. Robot control was proved to be precise in both simulation and real-world environments. Although self-localization solely derived from the wheel encoders deteriorates over time due to drifting, the AMCL particle filter did

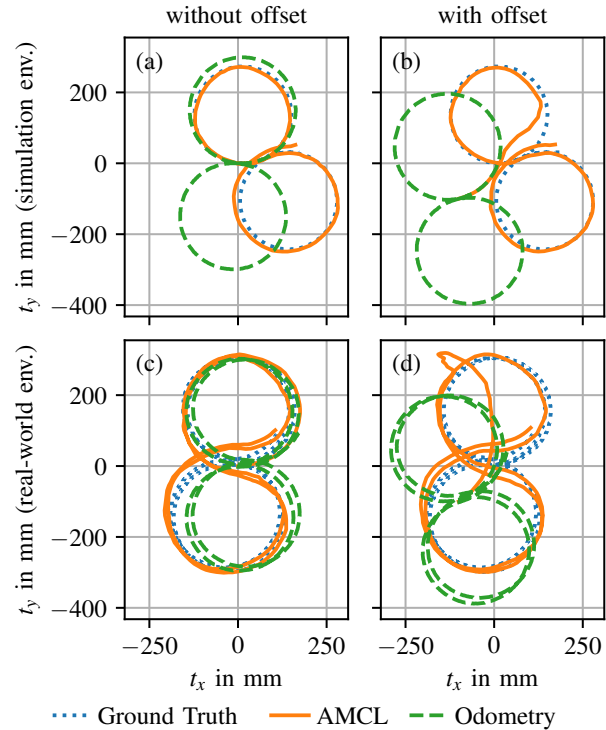


Fig. 3: Ground truth trajectories vs. the trajectories estimated by odometry and AMCL in simulation and real world environments. The right column shows that AMCL with emulated LIDAR data is able to recover from an initially incorrect pose estimate ($\Delta x = -100$ mm, $\Delta y = -100$ mm, $\Delta \theta = 0.3$).

not only achieve smaller pose errors but could also recover from incorrect initial pose estimates.

REFERENCES

- [1] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022.
- [2] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. Rodríguez Tsouroukdissian, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke, and E. Fernández Perdomo, "ros.control: A generic and simple control framework for ROS," *The Journal of Open Source Software*, 2017.
- [3] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, pp. 2149–2154 vol.3.
- [4] R. Amsters and P. Slaets, "Turtlebot 3 as a robotics education platform," in *Robotics in Education*, M. Merdan, W. Lepuschitz, G. Koppensteiner, R. Balogh, and D. Obdržálek, Eds. Springer International Publishing, 2020, pp. 170–181.
- [5] O. Gervais and T. Patrosio, "Developing an Introduction to ROS and Gazebo Through the LEGO SPIKE Prime," in *Robotics in Education*, M. Merdan, W. Lepuschitz, G. Koppensteiner, R. Balogh, and D. Obdržálek, Eds. Cham: Springer International Publishing, 2022, pp. 201–209.
- [6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, ser. Intelligent robotics and autonomous agents series. MIT Press, 2006.
- [7] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 573–580.

Elastic Structure Preserving Control for a Structurally Elastic Robot

Alexander Kitzinger¹, Hubert Gatringer¹ and Andreas Müller¹

Abstract—Elastic lightweight manipulators offer multiple benefits but come at the cost of increased structural flexibility, making the system more susceptible to vibrations. These circumstances require control concepts with a special focus on vibration suppression. Based on an lumped element model formulation, a control method called elastic structure preserving control is used for additional damping injection, while using standard motor PD control, to ensure low tracking error of the flexible link robot's end effector. As a first proof of concept for the used structural elastic robot the method is only applied for the first degree of freedom. The results obtained are further compared to a flatness-based control approach utilizing exact feed forward linearization and full state feedback control. Both methods are tested using cost-effective IMU measurements for feedback control, in addition to the motor measurements. The outcome demonstrates that, based on the evaluated angular accelerations, both methods achieve comparatively effective vibration damping relative to standard motor PD control.

Keywords: elastic structure preserving control, flatness-based control, elastic robot, lumped element model

I. INTRODUCTION

Elastic lightweight robots, such as the one shown in Fig. 1, are characterized by an improved payload-to-manipulator weight ratio, resulting in advantages like lower manufacturing costs, reduced energy consumption, and space-efficient usability. Additionally, their advantageous dynamic properties enable high speed manipulations, which are crucial for industrial applications. Nevertheless, high jerk inputs and external disturbances lead to non-desirable TCP oscillations, resulting in intolerable position errors and settling times. To tackle this challenges [1] presents a flatness-based trajectory control method, emphasizing the use of IMU sensors for vibration suppression. The aim of this work is to evaluate the feasibility and performance of elastic structure preserving (ESP) control introduced in [2] and benefit from its advantages, potentially also for structurally elastic robots. In doing so, an easily comprehensible controller parameterization is expected to enhance the damping characteristics of the considered flexible link manipulator. However, due to the limiting factors of the robot setup, a positive result is not guaranteed. Crucial aspects include the distinctive multiple oscillatory modes of the flexible links, bus delay times caused by the centralized ESP control scheme and noise and uncertainties introduced by the low-cost accelerometer and gyroscope measurements.

¹ Alexander Kitzinger, Hubert Gatringer, Andreas Müller are with Institute of Robotics, Johannes Kepler University Linz, 4040 Linz, Austria {alexander.kitzinger, hubert.gatringer, a.mueller}@jku.at

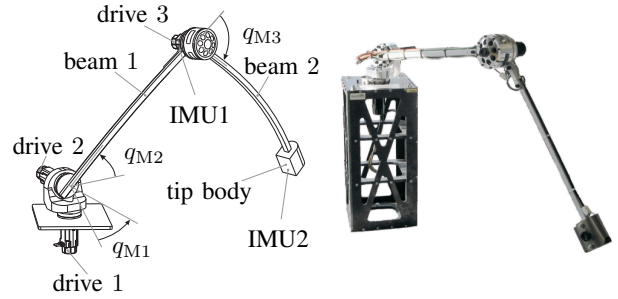


Fig. 1: Sketch and photo of considered elastic robot

II. MODELING

The foundation of the model-based control builds a lumped element model using virtual springs to represent the three elastic harmonic drive gears and two flexible beams. The formulation of the equations of motion (EoM) for the underactuated mechanical system is based on [3] and given by

$$\mathbf{M}_M \ddot{\mathbf{q}}_M + \mathbf{Q}_R(\dot{\mathbf{q}}_M) + \mathbf{K}(\mathbf{q}_M - \mathbf{q}_A) = \mathbf{Q}_M \quad (1)$$

$$\mathbf{M}_A(\mathbf{q}_A) \ddot{\mathbf{q}}_A + \mathbf{g}_A(\mathbf{q}_A, \dot{\mathbf{q}}_A) + \mathbf{K}(\mathbf{q}_A - \mathbf{q}_M) = \mathbf{0} \quad (2)$$

using the minimal coordinates of the three motor $\mathbf{q}_M$ and their corresponding arm angles $\mathbf{q}_A$. The positive definite, symmetric mass matrices $\mathbf{M}_M$ and $\mathbf{M}_A$ include the motor and arm inertia, whereas vector $\mathbf{g}_A$ describes the nonlinear gravitational, Coriolis and centrifugal forces of the links. Coupling between the actuated motor and under-actuated arm equation is represented by the diagonal and positive definite linear stiffness matrix $\mathbf{K}$. The vector $\mathbf{Q}_R$ contains considered viscous and Coulomb friction forces, while $\mathbf{Q}_M$ is the vector of the generalized motor driving torques.

III. CONTROL

According to [2] the control goal for the elastic robot is to derive a structure preserving state transformation that transforms the under-actuated system (1)–(2) into the quasi-full actuated closed loop form

$$\mathbf{M}_M \ddot{\tilde{\mathbf{q}}}_M + \mathbf{K}(\tilde{\mathbf{q}}_M - \tilde{\mathbf{q}}_A) = \tilde{\mathbf{Q}}_M \quad (3)$$

$$\mathbf{M}_A(\tilde{\mathbf{q}}_A) \ddot{\tilde{\mathbf{q}}}_A + \tilde{\mathbf{g}}_A(\tilde{\mathbf{q}}_A, \dot{\tilde{\mathbf{q}}}_A) + \mathbf{K}(\tilde{\mathbf{q}}_A - \tilde{\mathbf{q}}_M) = -\mathbf{D}\dot{\tilde{\mathbf{q}}}_A \quad (4)$$

where the adjustable positive definite diagonal-matrix $\mathbf{D}$ injects damping according to the new coordinates $\tilde{\mathbf{q}}^T = (\tilde{\mathbf{q}}_M^T, \tilde{\mathbf{q}}_A^T)$ and input $\tilde{\mathbf{Q}}_M$. The new arm coordinates correspond to the motion error of the arm angles $\tilde{\mathbf{q}}_A^T = \mathbf{q}_A - \mathbf{q}_{A,d}$ and the new motor coordinates $\tilde{\mathbf{q}}_M^T$ reflect the desired damping and tracking behavior. The transformation to the

closed loop form (3)–(4) does not cause dynamical shaping of the inertial properties and is preserving the initial stiffness $\mathbf{K}$ of the links. The gravitational and friction terms are compensated, while the Coriolis terms remain.

For a proof of concept, only the first degree of freedom $q_A = q_{A,1}$ will be considered, using stationary angles of $q_{A,2} = q_{A,3} = 0$ for the remaining arm and corresponding motor coordinates. Therefore, the control law simplifies drastically as the gravitation, centrifugal and Coriolis terms vanish. Equating (2) and (4) yields the state transformation for the motor coordinate

$$\tilde{q}_M = q_M - \underbrace{(q_{A,d} - K^{-1}D\dot{q}_A + K^{-1}M_A\ddot{q}_{A,d})}_{q_{M,d}}. \quad (5)$$

The corresponding input transformation, obtained by equating (1) and (3), characterizes the control law without friction compensation for the applied motor torque

$$Q_M = \underbrace{\tilde{Q}_M - D\dot{\tilde{q}}_M - M_M K^{-1}D\ddot{\tilde{q}}_M^{(3)}}_{Q_{da}} + \underbrace{(M_M + M_A)\ddot{q}_{A,d} + M_M K^{-1}M_A q_{A,d}^{(4)}}_{Q_{ff}} \quad (6)$$

and using cascaded motor PD control (servo drive) in the new coordinates

$$\ddot{\tilde{q}}_M = -K_D(K_P\tilde{q}_M + \dot{\tilde{q}}_M) \quad (7)$$

The i -th time derivative is denoted by $\tilde{q}_A^{(i)}$. The adjustable control parameters are K_P , K_D and the link-side damping factor D . Based on the desired motor position $q_{M,d}$, the feed forward Q_{ff} and damping torque Q_{da} the control law is implemented on the elastic robot using a cycle time of $400 \mu s$ and the setup shown in Fig. 2.

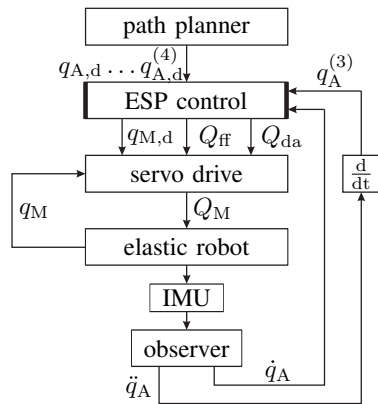


Fig. 2: Control Scheme

IV. RESULTS

The control method is tested using a required fourfold continuously differentiable $\sin^2$ trajectory with angular motion from -45° to 45° . The joint limitations are: maximum velocity 1.25 rad/s , maximum acceleration 15.6 rad/s^2 and maximum jerk 195.3 rad/s^3 .

The result in Fig. 3 shows that ESP control achieves significantly better tracking performance than simple PD motor joint control (same servo drive parameters), preventing the robot arm from overshooting oscillations as indicated by the angular accelerations $\ddot{q}_A$. After the trajectory, residual vibrations remain which result from model uncertainties, static friction and coupled in vibrations in other directions of motion that are not actively controlled. The vibration suppression and motor torque Q_M is comparable to the results obtained using the flatness-based approach from [1]. However, ESP control has the advantage that TCP damping can be easily varied and adjusted intuitively, making it particularly interesting for further investigations.

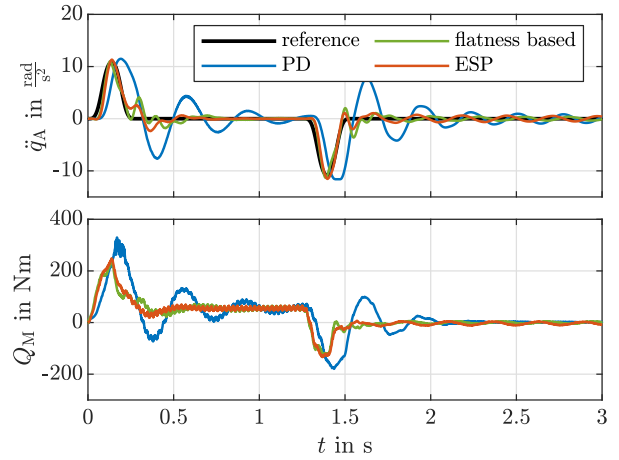


Fig. 3: Comparison of the tested control methods

V. CONCLUSION

This initial test demonstrates that elastic structure-preserving control can also be beneficial for structural elastic robots using IMU measurements. Nevertheless, the next step should involve extended research implementing the control method in combination with a suitable real-time observer including all three DOF of the elastic robot. Furthermore, a time-optimal application as outlined in [4] is desirable.

ACKNOWLEDGMENT

This work has been supported by the “LCM – K2 Center for Symbiotic Mechatronics” within the framework of the Austrian COMET-K2 program.

REFERENCES

- [1] P. Staufer, H. Gatringer, and H. Bremer, “Vibration Suppression for a Flexible Link Robot using Acceleration and/or Angular Rate Measurements and a Flatness Based Trajectory Control,” in *35th Mechanisms and Robotics Conference*, 2011.
- [2] M. Keppler, D. Lakatos, C. Ott, and A. Albu-Schäffer, “Elastic Structure Preserving (ESP) Control for Compliantly Actuated Robots,” *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 317–335, 2018.
- [3] H. Bremer, *Elastic Multibody Dynamics - A Direct Ritz Approach*, ser. Intelligent Systems, Control and Automation: Science and Engineering. 69121 Heidelberg, Tiergartenstraße 17: Springer Verlag, 6 2008, vol. 35.
- [4] K. Springer, H. Gatringer, and P. Staufer, “On time-optimal trajectory planning for a flexible link robot,” *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Eng.*, vol. 227, no. 10, pp. 751–762, 11 2013.

A Modular and Configurable Architecture for ROS 2 Hardware Integration with micro-ROS

Jakob Friedl¹ and Markus Bader¹

Abstract—In general, a vehicle cannot follow a given trajectory if the control commands for the motor controllers are not delivered to the hardware in time. This issue arises when a standard computer running ROS 2 is used for control without a real-time extension. This paper presents an architecture that leverages micro-ROS on an ESP32-C6 with a RISC-V CPU running a Real-Time Operating System (RTOS). The goal is to demonstrate that drift compensation, based on odometry and IMU data, can be performed in real-time directly on the microcontroller. As a first step, we show how micro-ROS handles robot kinematics (Ackermann steering) within the firmware, configured via a persistent parameter server. We demonstrate that this design improves integration simplicity, adaptability, separation of concerns and evaluate real-time compliance.

Index Terms—micro-ROS, mobile robotics, embedded systems, ROS 2

I. INTRODUCTION

This paper proposes a microcontroller-based ROS 2 integration architecture aimed specifically at modular, configurable robotics hardware. Leveraging the micro-ROS framework, it provides a plug-and-play solution that simplifies interfacing embedded hardware with higher-level ROS 2 ecosystems shown in Figure 1. Traditional designs often use onboard computers that interact directly with hardware components (as can be seen in [7]), a strategy that can lead to redundant software development, a mixing of low-level hardware interactions with higher-level control concerns, and challenges in ensuring real-time performance. Micro-ROS addresses these limitations by extending ROS 2 functionalities directly to resource-constrained microcontrollers.

In this approach, the micro-ROS agent on the ROS 2 host translates the lightweight eXtremely Resource-Constrained Environments-Data Distribution Service (XRCE-DDS) middleware used by micro-ROS into standard DDS messages [1], [2]. It supports transports such as UDP and USART by default, and can be extended with custom implementations. However, hardware variations often force firmware rebuilds or manual tweaks; our design instead embeds an NVS-backed parameter server, allowing kinematic and hardware settings to be adjusted live via ROS 2 parameters.

Incorporating an RTOS into the firmware permits local prioritization of time-critical tasks, ensuring reliable operation under strict deadlines. Prior work in underwater vehicles demonstrates a micro-ROS RTOS setup [6], but offers no built-in mechanism for runtime customization or a clear task breakdown. In contrast, we leverage advanced

¹The authors are with Faculty Informatics at TU Wien, Vienna, Austria. firstname.lastname@tuwien.ac.at

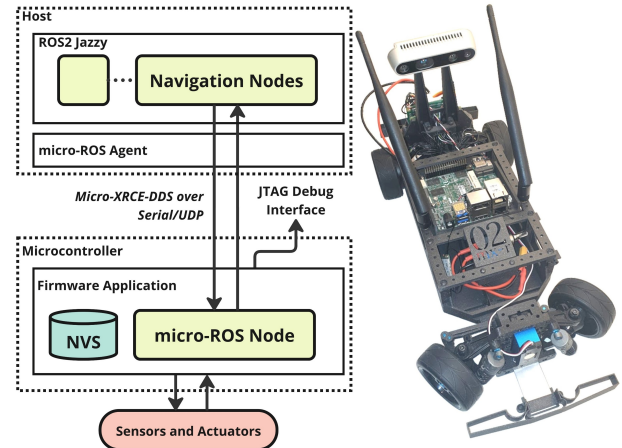


Fig. 1. System overview and test platform on the right

RTOS features alongside our persistent parameter server and detail a modular FreeRTOS task architecture (see II-B). While demonstrated on an Ackermann-steering robot, our modular FreeRTOS task breakdown and NVS parameter server generalize directly to other ROS 2-integrated hardware and form the basis for more advanced microcontroller-based trajectory following.

II. PROPOSED ARCHITECTURE

A. Hardware Components and System Overview

The initial question was how best to structure motor control for an Ackermann robot integrated with ROS 2. As highlighted in the introduction, conventional approaches often conflate low-level hardware interfacing with high-level control, impede real-time performance, and lack a common framework for comparing firmware designs. The goal of the specific design in this paper is to accept twist commands via the `/cmd_vel` topic, compute the necessary kinematics for motor speeds and steering in the firmware, while publishing the resulting odometry, thus simplifying the interface between ROS 2 system and hardware. Kinematic and control variables are live-adjustable through the parameter server. This approach is demonstrated on the MX-Car [4], a mobile robot developed at TU Vienna with an Ackermann drivetrain featuring two non-steering rear wheels (driven by BLDC hub motors) and a front steering servo.

Figure 1 shows the overall system with the hardware platform at the right. The top segment depicts the onboard or

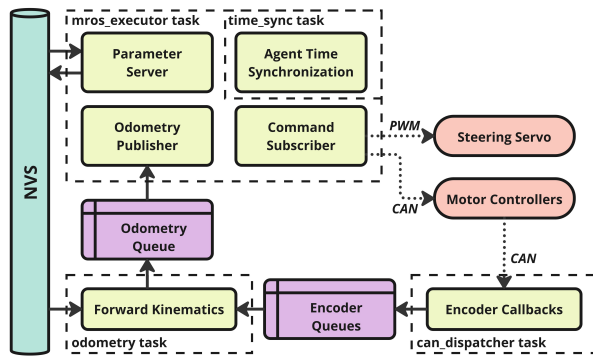


Fig. 2. Firmware application architecture overview

network-connected computer running ROS 2 (with a Dockerized micro-ROS agent), the middle segment shows the ESP32 running FreeRTOS with a dedicated micro-ROS node using Non-Volatile Storage (NVS) for persistent configuration and the respective interaction with the sensors and actuators. A more detailed look at the firmware is provided in the next section.

The single-core ESP32-C6-DevKitM-1 is used as primary controller due to its processing capabilities, peripherals and open RISC-V architecture. Micro-ROS integrates into its build system via an ESP-IDF component [3]. Motor control is provided by two daisy-chained ODrive-Micro controllers, with a Control Area Network (CAN) transceiver interfacing the microcontroller to exchange commands (e.g., position, speed, effort) and feedback (e.g., velocity, current) [5].

B. Firmware and Communication

Figure 2 shows a simplified view of the firmware architecture. We enforce strict timing by splitting work across four FreeRTOS tasks with distinct priorities. The `can_dispatcher` and `mros_executor` tasks run at highest priority, ensuring no inbound messages are lost. The `odometry` task runs at medium priority, and the low-priority `time_sync` task handles agent clock alignment via micro-ROS mechanisms (obtaining the host timestamp) without interfering with real-time deadlines.

The `can_dispatcher` manages all CAN traffic to and from the motor drivers, and the `mros_executor` drives the micro-ROS executor, handling subscribers, timers, and an NVS-backed parameter server. Configuration parameters and kinematic properties (e.g., wheel base, track width) are stored persistently in non-volatile storage and can be updated via standard ROS 2 param calls.

Incoming twist messages on `/cmd_vel` are processed in the `mros_executor`: the callback computes motor and steering setpoints using the Ackermann kinematic model as described by [8] and forwards them to the drives, so no command queuing is needed. In the `odometry` task, execution blocks until fresh encoder estimates arrive from both drives; it then performs forward kinematics to update an internal pose. A timer-driven publisher retrieves this pose through a single-

element FreeRTOS queue (buffering only the latest state) and publishes on `/odom`, timestamping the message with the arrival time of the latest encoder sample. An onboard or remote ROS 2 computer (depending on the transport) can then integrate higher-level features such as path planning or trajectory control.

III. EVALUATION

To assess the real-time performance of the integration of this architecture, we measured timing at 10Hz over the 115200 baud serial transport during full system operation. For odometry, 1200 messages were received by the host. The reception intervals exhibited a mean of 99.998ms, a standard deviation of 5.63ms, and a peak-to-peak jitter of 52.0ms. We then measured the round-trip delay from publishing a stamped twist command message on the host to applying motor setpoints on the microcontroller by again logging 1200 messages. This delay averaged 28.684ms, with a standard deviation of 4.73ms, and spanned 27.82ms peak-to-peak. In both cases, no messages were lost.

Compiled with space optimization, the complete firmware occupies 342.9kB of flash (4.09%) and 118.2kB DIRAM (26.15%), confirming a compact footprint.

IV. SUMMARY AND OUTLOOK

This paper presents an architecture that integrates resource-constrained microcontrollers with ROS 2 using micro-ROS on an RTOS. It exposes hardware interfaces as standard topics and services and simplifies configuration via a persistent parameter server. We evaluated its real-time performance at 10Hz over serial transport, observing latency and jitter levels acceptable for typical mobile robotics applications, while still leaving room for optimization.

Future work will build on this architecture to realize complex trajectory control on the microcontroller. The aim is to calculate collision-free trajectories in the ROS 2 framework on the host and then pass them on to the micro-ROS controller for execution.

V. ACKNOWLEDGMENT

This research is supported by the Austrian Science Fund (FWF), under project No. 923138, GreenFDT.

REFERENCES

- [1] "Features and architecture of micro-ros," <https://micro.ros.org/docs/overview/features/>, 2025, [Online; 22 April 2025].
- [2] "Micro xrcs-dds documentation," <https://micro-xrcs-dds.docs.eprosima.com/en/latest/>, 2025, [Online; 22 April 2025].
- [3] "Micro-ros esp-idf component," https://micro.ros.org/docs/tutorials/core/first_application_rtos/esp32/, 2025, [Online; 22 April 2025].
- [4] "Mx-car github repository," <https://github.com/tuwien/mx-car>, 2025, [Online; 22 April 2025].
- [5] "Odrive-micro hardware datasheet," <https://odriverobotics.com/docs/micro-datasheet>, 2025, [Online; 22 April 2025].
- [6] P. A. Gutiérrez-Flores and R. Bachmayer, "Concept development of a modular system for marine applications using ros2 and micro-ros," in *2022 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*. IEEE, 2022, pp. 1–6.
- [7] W. Jo, J. Kim, R. Wang, J. Pan, R. K. Senthikumar, and B.-C. Min, "Smartbot: A ros2-based low-cost and open-source mobile robot platform," *arXiv preprint arXiv:2203.08903*, 2022.
- [8] A. Kaltenegger, "Physical and graphical simulation of an ackermann steered vehicle," *Bachelor's Thesis, TU Wien*, 2016.

Automating 3D printing for mass production

(b) Felix Daunert¹, Prof. Dr. Tobias Weiser², Prof. Dr. Dirk Jacob, Maximilian Besler, Florian Schmolke*

Abstract—This paper presents the development and evaluation of an automation concept for high-performance 3D printers in an industrial environment. The paper's special characteristic is the multi-domain approach, which combines design and development of the cell with parallel simulation studies. The 3D printing robot cell was complimented with an individual gripping system and a magazine for printing plates. The final production performance of the concept was evaluated with simulation studies of the robot cycle time and overall performance.

Index Terms—Automation, 3D printing, Simulation

I. INTRODUCTION

This paper presents the development of an integrated automation concept for an industrial 3D printing farm with a production capacity in excess of one million parts per annum. The focus was on the integration of automated loading and unloading of a Masked Stereolithography (MSLA) high-speed 3D printer (Solidator 8K [6]) by a 6-axis robot (KUKA KR10 R1100). This project integrated offline simulations and time-valued Petri nets in the design of the automation concept to enhance the overall performance of the concept. The analysis is conducted using KUKA.Sim and Siemens Plant Simulation, where cycle times, material flows and path planing is layed out and optimised. The final concept demonstrates high production capacity and scalability, making it a promising alternative to existing industrial solutions.

II. STATE OF THE ART

Additive manufacturing (AM) is currently experiencing an increased integration of automated processes with the objective of enhancing production efficiency and improving profitability like the Figure 4 Production by 3D Systems [3]. This system provides a comprehensive solution that integrates printing, cleaning and UV curing in a single module. However, this solution is costly and exhibits limited flexibility with regard to scalability. Other manufacturers offer retrofit options that allow partial automation, such as part removal (see Form Auto, Formlabs [4]). However, AM technology requires good automation and scalability for series production [1]. Modular automation concepts, in contrast, are distinguished by a deliberate selection of printing technologies, optimised cell layouts and process optimisation based on simulation. To ensure increased automation and

scalability, it is necessary to employ simulation software such as KUKA.Sim and Plant Simulation. This approach facilitates the determination of realistic operating parameters and consequently enables the evaluation of the developed concept against the requirements.

III. REQUIREMENTS ANALYSIS

In order to achieve an annual production of at least one million parts with dimensions of 50 mm x 30 mm x 20 mm (W x D x H), a calculation of the required number of printers and print cycles is necessary. In this context, the Solidator 8K was identified as a suitable solution, given that its printing time is solely dependent on the part height and it possesses a substantial build plate with dimensions of 330 mm x 185 mm. For automation purposes, an actuator is requisite. This actuator must meet certain requirements, namely the capacity to securely grip the printing plates, a minimum load capacity of 2.5 kg, and six degrees of freedom to facilitate the dexterous manipulation of the printing plates. Furthermore the workspace must accommodate six printers.

IV. GRIPPER AND MAGAZINE DESIGN

In the course of the conception a two-finger gripper was developed for the handling of the printing plates by the robot. Its an asymmetrical construction with the objective of avoiding collisions with the printing bed. The gripper construction, see Fig. 1, was designed with consideration for mechanical and dynamic aspects in order to ensure the secure grasping of the printing plates.



Fig. 1. gripper (with print bed on the left)

A modular magazine functions as a print plate buffer between the outsourced reprocessing and the loading of the print plates. The magazine and the gripper are both crucial for a comprehensive simulation of the system's processing time and, by extension, its productivity. The magazine is employed as a station for the taking of new plates, while the gripper ensures the precise path planing during grasping and handling of these plates.

V. ROBOT SELECTION

A 6-axis kinematic system is evidently the optimal solution, as the requirements clearly indicate. The 6 Solidators

*This work was not supported by any organization

¹ Felix Daunert, Student of Automation and Robotics, Faculty Electrical Engineering, University of applied science Kempten, 87435 Kempten, Germany felix.daunert@stud.hs-kempten.de

²Prof. Dr. Tobias Weiser is with Institute for Applied AI and Robotics, University of applied science Kempten, 87435 Kempten, Germany tobias.weiser@hs-kempten.de

are arranged in a U-shape around the kinematics. A reach analysis in KUKA.Sim revealed that the KUKA KR10 R1100 meets all requirements with a reach of 1100 mm, a payload capacity of 10 kg and a repeatability of ± 0.02 mm. These parameters ensure that the printers can be processed with the requisite precision and without collisions in a centrally arranged cell.

VI. ROBOT SIMULATION

The movement patterns of the integrated robot system were modelled and simulated in KUKA.Sim, with CAD models of the U-arrangement of the Solidators around the kinematics, the print plate magazine and the gripper being imported (see Fig. 2). These must be integrated into the simulation to ensure precise programming of collision free movement patterns of the kinematics. A black box is utilised as a depository for the printed plates, since there was no interface to subsequent stations defined.

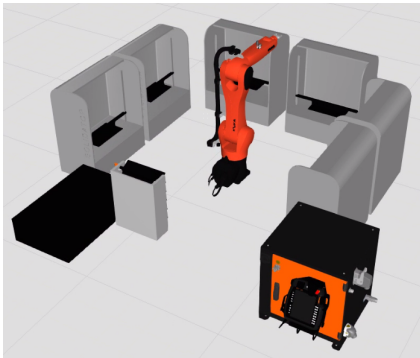


Fig. 2. KUKA.Sim Layout with black box and magazine on the left

The outcomes of the simulations demonstrate that the KR10 R1100 is capable of achieving an average cycle time of 15.5 seconds per printer. These results take into account critical movement segments, such as the reduction in acceleration during the transportation of printed plates. The reorientation required for each printer has small effect as the maximum deviation of the mean is 1 second. This is due to the tool center point (TCP) speed of $2 \frac{m}{s}$ while reorienting. The majority of the mean cycle time is spent picking up and setting down the printing plates at a reduced TCP speed of $0.1 \frac{m}{s}$. Utilizing these simulations enabled the programming of collision-free motion sequences and the determination of real cycle times for the robot. This data is imperative for simulating the overall task.

VII. TOTAL CYCLE SIMULATION

The basis for the overall process simulation in Plant Simulation is a time-evaluated Petrinetwork, in which a shift calendar has been introduced for the simulation of a calendar year. This is due to the fact that maintenance, potential repairs and the replacement of the resin reservoirs are manual tasks. The calendar is modeled on a working week from Monday to Friday 8 hours daily, and takes into account conditions such as availability. The educated guess is made

that the printers are available at 95% of the time and the kinematics at 100%, which is regarded as ideal. The duration of the printing process is either 20 minutes for 80 standing parts or 6 minutes for 24 lying parts with a constant setup time of 2 minutes and a processing time of 15.5 seconds by the kinematics. The simulation examined the standing and lying arrangements of parts on the plates, with the upright configuration resulting in a higher part production due to a reduced proportion of setup time in the total operating hours. For the calendar year, a standing arrangement of parts on the plate yielded a theoretical annual output of 2.3 million parts. This calculation is based on the utilisation of 6 Solidator 8Ks and a KUKA KR10 R1100 kinematic system. The scalability of the system is evident in its ability to accommodate the placement of the kinematic structure on a 300 mm-high platform, enabling the construction and processing of an additional second storey comprising 6 solidators. Furthermore, the robot demonstrates the capacity to efficiently handle twelve solidators, resulting in an average cycle time of 16.5 seconds.

VIII. CONCLUSION

In this project, an automation concept was developed that uses 6 Solidator 8K printer in combination with a precisely matched 6-axis robot (KUKA KR10 R1100) to realise an efficient and scalable 3D printing farm. The optimisation of production was achieved through the targeted use of offline simulation and time-weighted petri nets, resulting in an average kinematic processing time of 15.5 seconds per printer. The developed solution has been shown to exceed the production target by a factor of 2.3, thus representing a commercially viable alternative to existing industrial systems. The scientific significance is high for the robot cycle time but reduced by the educated guesses concerning the availability and setup time of the printers. Future research should focus on implementing real printer characteristics for further optimizing. There is the potential through artificial intelligence in the domain of path planning [5] or in Petrinetworks [2] to enhance flexibility and efficiency of the cell.

REFERENCES

- [1] S. Bindl and C. Csiky-Strauss, "Developing applications for additive manufacturing series production," in *2017 IEEE European Technology and Engineering Management Summit (E-TEMS)*, 2017, pp. 1–7.
- [2] S. Hammedi, J. Elmelliani, L. Nabli, A. Namoun, M. H. Alanazi, N. Aljohani, M. Shili, and S. Alshmrany, "Optimizing production in reconfigurable manufacturing systems with artificial intelligence and petri nets," *International Journal of Advanced Computer Science & Applications*, vol. 15, no. 10, 2024.
- [3] (2025) 3d systmes: Figure 4 production. [Online]. Available: www.3dsystems.com/3d-printers/figure-4-production
- [4] (2025) Formlabs form auto. [Online]. Available: formlabs.com/3d-printers/form-auto/?srsltid=AfmBOoqw74MJg4-jKlu6dgDY7wwmBpm786qT5olcFnwQ71_uFFJd3gAV
- [5] M. Tamizi, M. Yaghoubi, and H. Najjaran, "A review of recent trend in motion planning of industrial robots," in *International Journal of Intelligent Robotics and Applications*, Nagoya, Japan, Feb. 2023, p. 253–274.
- [6] tangible engineering GmbH. (2025) Solidator 8k: Resin 3d printer with 8k msla technology for industry. [Online]. Available: solidator.com/en/

A Cost-Effective Testbed for Measuring the Performance of Reference Switches*

Tobias Hofer¹, Thomas Schmid¹, and Olaf Saßnick²

Abstract—Reference switches are widely used in automation and robotics for positioning tasks. However, typically data sheets do not provide specifications regarding their repeatability performance. This work presents a cost-effective testbed to measure the repeatability of different switch types. It uses a high-resolution linear scale and an ATmega microcontroller to detect small deviations in switch trigger points. The firmware is optimized for low latency and openly available. Finally, the proposed testbed is used to assess the performance of different reference switch types.

Index Terms—Repeatability, Reference Switches, Automation

I. INTRODUCTION

In robotics and automation, accurate positioning is required to ensure results being consistent and of high-quality. This is typically achieved using integrated measurement systems. Measurement systems based on incremental encoders are widely used due to their cost-effectiveness, however they require switches for homing the system to a known reference position. While accurate detection of the reference position is critical for overall repeatability, data sheets for switches typically do not specify their repeatability performance. This work therefore proposes a testbed for evaluating the repeatability of various reference switches experimentally, addressing two research questions: How can a cost-effective testbed be designed to evaluate the repeatability performance of reference switches? How repeatable is the performance of commonly used reference switches? While this topic certainly is studied for commercial applications, to the authors' knowledge, no publication on this topic exists so far.

The further content is structured as follows: Section II introduces the terms accuracy and repeatability, along with different types of reference switches. Section III then describes the testbed design, covering the mechanical and electrical setup as well as the firmware implementation. Next, Section IV presents selected results. Finally, Section V summarizes the work and outlines future tasks.

II. BACKGROUND

Accuracy describes how closely a target position can be reached. Any deviation from the target position can be considered an accuracy error [1].

Repeatability refers to how closely the same position can

*This work received no organizational support, authors contributed equally.

¹Tobias Hofer and Thomas Schmid are bachelor students in Information Technologies at the Salzburg University of Applied Sciences, {thofer,tschmid}.itsb-b2022@fh-salzburg.ac.at

²Olaf Saßnick is with Salzburg University of Applied Sciences, olaf.sassnick@fh-salzburg.ac.at

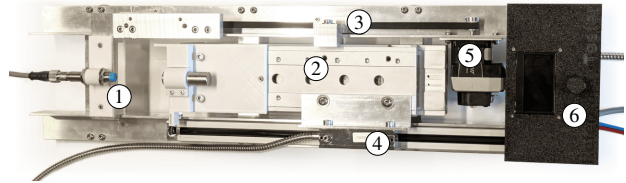


Fig. 1. Top view of the assembled testbed, with ① indicating the reference switch under test, ② the linear slide, ③ the belt drive, ④ the linear scale, ⑤ the stepper motor, and ⑥ the user interface, consisting of a display and input encoder knob.

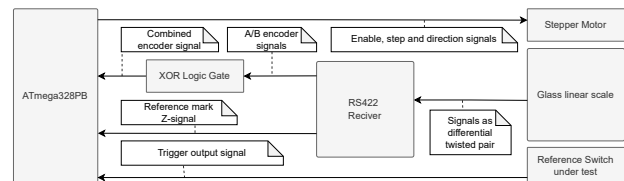


Fig. 2. The electrical communication diagram of the proposed testbed.

be reached again under identical conditions [1]. A high repeatability is possible even with low accuracy, as long as consistently the same accuracy error occurs.

Different types of reference switches are used in automation and robotics. They can be categorized as mechanical or electrical, and as contact or contactless. The most common types are capacitive, inductive, photoelectric, and mechanical switches.

III. EXPERIMENT SETUP

The hardware is selected with cost-effectiveness in mind. Fig. 1 shows the assembled testbed. A pneumatic slide (SMC EMXS20-125) is used as a linear axis, with its cross-roller bearings providing low friction and adjustable mechanical play. Its pneumatic cylinders are decoupled from the motion axis and instead actuated by a NEMA17-sized MDrive stepper motor via an MXL belt. The slide's mass helps dampen stepper motor pulses, while the movement is measured using an incremental Aikron MDI series linear scale (1 μ m resolution with quadrature encoding). The system is controlled by an ATmega328PB evaluation board. To allow for different sensors with varying physical dimensions and mounting options, the sensor is attached to the testbed via a swappable adapter plate. The total hardware cost is approximately 400 €, with the linear scale being the most expensive component at around 200 €.

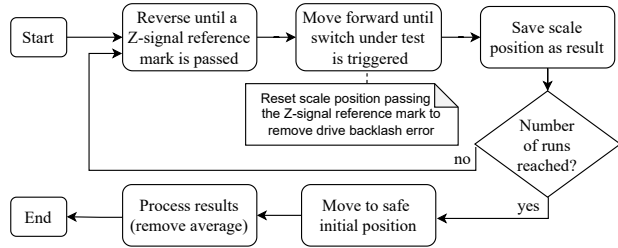


Fig. 3. The experiment flow implemented in the firmware.

A. Electrical Setup

Fig. 2 gives an overview of the testbed’s electrical communication setup. The central control unit is an ATmega328PB microcontroller and all signals, inputs or outputs, are processed or generated by it. The output signals for the stepper motor and input signal of the reference switch are directly interfaced to the microcontroller. The scale provides three separate signals – A, B, and Z – as differential RS-422 signals to increase noise immunity. A separate RS-422 receiver is used to read the differential signals. The Z-signal is the reference signal of the scale and occurs every 50 mm. The A/B signals indicate the incremental position change of the scale. They are 90° phase-shifted relative to each other, and allow to determine the direction of movement. In this setup, the A/B signals are combined using an XOR logic gate to increase the effective resolution by counting every edge on both signals.

B. Firmware Implementation

Fig. 3 shows the implementation for a test cycle in the firmware¹. First the slide is reversed until a reference mark (Z-signal) on the linear scale is passed. Next the slide is moved forward until the reference switch under test is triggered and the position of the linear scale is saved. The linear scale position is tracked using XOR-ed A/B signals as a clock source for two 16-bit hardware timers. As the ATmega328PB supports only one clock edge per timer, two timers are used to count on both edges. An overflow counter is added for both timers to track distances beyond the 16-Bit counting range. The resulting distance is calculated by combining both timer values and the overflow counter.

The Z-signal is handled as an external interrupt, resetting both timer values and the overflow counter on each reference mark. This way, after reversing until the reference mark and moving towards the switch again, a second reset occurs, removing the need to account for mechanical belt drive backlash. To ensure that the testbed firmware has a minimal reaction time, the main loop only needs to check the switch output signal while taking a measurement. The resulting measurement values with 1 μ m resolution can be retrieved using the UART interface of the microcontroller.

¹The source code is openly available under: <https://github.com/sas-o/2025-arw-refswitch-tester>

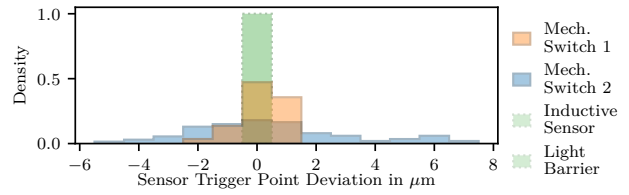


Fig. 4. Histogram of sensor trigger points deviation across $n = 200$ runs.

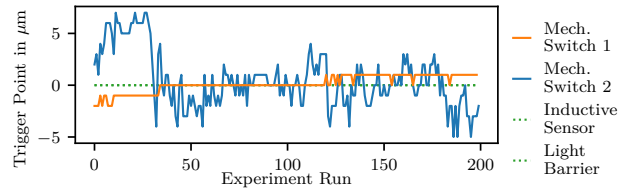


Fig. 5. Deviation of the sensor trigger points per experiment run.

IV. RESULTS

The results for different types of end switches are briefly presented in Fig. 4 and Fig. 5. For each sensor, the experiment was repeated $n = 200$ times. Based on the tested sensors and results, both the selected optical light barrier and the inductive sensor perform better than the provided measurement accuracy of 1 μ m. Therefore, we conclude that their repeatability is smaller than 1 μ m, exceeding the testbed’s measurement resolution. At the same time, it also demonstrates the repeatability provided by the testbed itself. For mechanical switches, a larger deviation is expected, as they require physical contact. Two different models were tested, with model 1 clearly outperforming model 2. In Fig. 5, the deviation of the sensor trigger point is shown per run. No linear error is visible, which could have indicated an issue with the testbed itself.

V. CONCLUSION

In robotics and automation, switches are widely used for reference positioning. However, most data sheets for switches do not specify a repeatability performance for such a task. This work presents a cost-effective testbed for measuring the repeatability of different types of reference switches. Measurement results with a repeatability of 1 μ m are achievable. Next, the testbed will be used to evaluate a broader range of reference switches under varying conditions, such as different movement speeds and trigger directions. Further improvements to the testbed could include replacing the stepper motor with a hollow cup DC motor to reduce induced vibrations, installing limit switches for improved safety, and adding temperature and humidity logging to monitor the test environment.

REFERENCES

- [1] K. C. P.S. Shiakolas and T. Yih, “On the accuracy, repeatability, and degree of influence of kinematics parameters for industrial robots,” *International Journal of Modelling and Simulation*, vol. 22, no. 4, 2002. [Online]. Available: <https://doi.org/10.1080/02286203.2002.11442246>

A Trajectory Consistency Metric for GNSS Anomaly Detection with LiDAR Odometry

Hans-Peter Wipfler¹ and Gerald Steinbauer-Wagner¹

Abstract— While Global Navigation Satellite System (GNSS)-based robot localization is successful in open scenarios, it quickly becomes unreliable in GNSS-degraded environments such as forests. With the increasing interest in using autonomous robots in forestry, it becomes more important to have reliable localization in forest environments, which are among the most challenging areas for GNSS-based localization. Having an estimate for the quality of the localization can help achieve this. While GNSS receivers provide uncertainty estimates based on signal characteristics and the satellites' constellation, practical experience shows that these values are less meaningful in forests. This paper presents an error metric that exploits the properties of commonly used robot localization setups to assess the quality of the localization. This assessment is based on a comparison between a LiDAR odometry-based local trajectory estimate and a GNSS-based global trajectory estimate in their respective coordinate systems. A qualitative analysis shows that the metric enables meaningful statements about the quality of position estimates derived from GNSS measurements in the global coordinate system.

Index Terms— anomaly detection, GNSS, LiDAR odometry

I. INTRODUCTION

State estimation architectures of mobile robots often separate global and local state estimation for localization [6],[3]. This is done by using two world-fixed coordinate systems, a local coordinate frame that is locally consistent but suffers from long-term drift, and a global coordinate frame that is globally consistent but suffers from transient errors in GNSS-based position information. In forest environments, GNSS-based position estimation is heavily influenced by the surrounding environment due to signal shading and reflections caused by objects like trees or rock walls [2]. Even when the GNSS data is fused with IMU (Inertial Measurement Unit) data, practical experience has shown that these phenomena still have a large impact on the global position estimate [6]. However, many robots today are equipped with a LiDAR sensor, which can be used for local motion estimation and provides low-drift, locally consistent position estimates [5]. This work exploits the properties of local and global trajectories to detect patterns in the global trajectory that are not backed by the local trajectory. Based on this we developed a metric for assessing global localization quality, which allows monitoring of localization quality in real-time, making it usable for anomaly detection, adaptive sensor fusion, or GNSS rejection strategies.

^{*}This work was funded by the Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology of Austria and the Austrian Research Promotion Agency (FFG) with the project RoboAlm.

¹Hans-Peter Wipfler and Gerald Steinbauer-Wagner are with the Institute of Software Engineering and Artificial Intelligence, Technical University of Graz, Austria {hans-peter.wipfler, gerald.steinbauer-wagner}@tugraz.at

II. RELATED WORK

LiDAR-based odometry estimates the motion of a robot by aligning successive point clouds. Direct LiDAR Odometry (DLO) [1] is a computationally efficient approach that enables real-time LiDAR odometry on resource-constrained robotic platforms. One of the key features of the method is a submapping strategy that aims to keep the position estimate locally consistent. This makes it a reasonable choice for the use in local state estimation in forests.

In [7], the authors propose the use of trajectory similarity metrics for comparing a reliable short-term trajectory from motion estimation with an IMU with a trajectory obtained from GNSS measurements. These metrics compare only the similarity of the point sets. In contrast, the proposed approach computes its error value based on full transformations $\in \text{SE}(3)$. This allows the application of orientation- and translation-based error metrics.

III. METHODOLOGY

A. Localization Consistency Evaluation

To assess the reliability of GNSS-based localization in forest environments, we introduce a trajectory error metric that evaluates the consistency between local trajectories and global trajectories. Since the LiDAR odometry trajectory is locally consistent, it serves as a short-range reference. To achieve this, a relative pose error estimate between pose pairs from the global and local trajectories is used. To fully exploit the information contained in poses in $\text{SE}(3)$, an alignment of the trajectories is necessary to ensure that both position and orientation are compared meaningfully. In addition, the resulting error estimate should show a high sensitivity to the consistency of the most recent pose. In order to achieve this objective, each transformation used is related to this pose which is illustrated in Figure 1. For each evaluated pose, a subtrajectory is selected using a fixed spatial window defined by the parameters Δs_a and Δs_b where Δs_a defines the minimum look-back distance, ensuring that only sufficiently separated past poses are included, and Δs_b defines the maximum look-back distance, limiting the subtrajectory length to prevent excessive drift influence. Given a trajectory parameterized by the cumulative distance traveled s from the LiDAR odometry, the sub-trajectory consists of poses selected within the interval $[s - \Delta s_b, s - \Delta s_a]$. By choosing poses within this range, we ensure that the sub-trajectory captures the recent motion history while maintaining a stable reference for error computation. This results in three necessary steps that must be performed for each pose of interest: 1) subtrajectory selection: collect past poses within the interval $[s - \Delta s_b, s -$

Δs_a], 2) alignment: align local and global subtrajectories to ensure a meaningful comparison of transformations, 3) error computation: compute the consistency error between both subtrajectories as defined below.

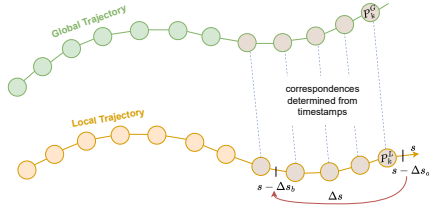


Fig. 1. Illustration of the trajectories used to compute the consistency error. The green poses are the poses of the trajectory of the robot in the global coordinate system. The orange poses are the poses of the robot in the local coordinate system. The global poses are related to the local ones based on time, meaning every P_k^L maps to a P_k^G .

B. Error Metric Formulation

The transformation from the current pose to the pose at $s - \Delta s$ in the aligned local trajectory is given by:

$$T_{s-\Delta s, s}^{L*} = (T_{\text{align}}^L \cdot P_{s-\Delta s}^L)^{-1} \cdot (T_{\text{align}}^L \cdot P_s^L), \quad (1)$$

where T_{align}^L is the transformation obtained in the alignment step, and P_s^L is the pose of the local trajectory at traveled distance s . The global trajectory is related to the local trajectory by the time $t(s)$, with poses defined as $P_{t(s)}^G$, and the corresponding transformation:

$$T_{t(s-\Delta s), t(s)}^G = (P_{t(s-\Delta s)}^G)^{-1} \cdot P_{t(s)}^G. \quad (2)$$

The relative transformation error is computed as:

$$E(\Delta s, s) = (T_{t(s-\Delta s), t(s)}^G)^{-1} \cdot T_{s-\Delta s, s}^{L*} \in \text{SE}(3), \quad (3)$$

where any error metrics for $\text{SE}(3)$, such as rotational or translational error, can be applied. To demonstrate the approach we employ the translational error according to [4]:

$$e_T(\Delta s, s) = \|\text{trans}(E(\Delta s, s))\| \in \mathbb{R}^+. \quad (4)$$

Finally, the consistency error is computed as:

$$\overline{e_T(s)} = \frac{1}{(\Delta s_b - \Delta s_a)} \int_{\Delta s_a}^{\Delta s_b} e_T(\sigma, s) d\sigma \in \mathbb{R}^+. \quad (5)$$

$\overline{e_T(s)}$ represents a metric for the consistency of the local and global subtrajectory and consequently for the current quality of localization.

IV. RESULTS

In the implementation, the integral for the consistency error from Equation 5 is approximated using the trapezoidal rule, performed on poses sampled over s for $\Delta s_a = 0$ and $\Delta s_b = 15m$. To evaluate the metric, we used data collected in a forest setting where the global trajectory was estimated using a *geo-konzept geo-kombi* INS/GNSS system, while the local trajectory was derived from DLO using data from a *Livox MID-360* LiDAR sensor. Figure 2 shows a part of a trajectory estimated by the GNSS system, where the value

of $\overline{e_T(s)}$ is color coded. The robot moved along the middle of a forest road. The road shown in the underlying map can be used as a qualitative reference. It is clearly visible that the estimate of $\overline{e_T(s)}$ is high for obvious anomalies, while it is low for regions where the estimate is likely to be correct. This observation was further confirmed by analyzing the consistency error over a trajectory of more than 6km, showing a strong correlation between high error estimates and significant GNSS inconsistencies.

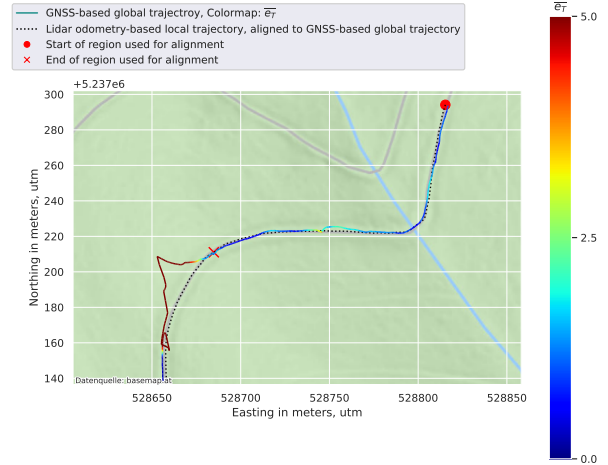


Fig. 2. Global GNSS-based trajectory estimate on a forest road with the corresponding consistency errors $e_T(s)$ color coded. For visualization purposes, the local LiDAR odometry-based trajectory estimate is additionally shown, aligned to the global trajectory estimate using a selected region.

V. CONCLUSION

This work introduces a metric to assess the reliability of GNSS-based localization in forest environments. It compares local LiDAR odometry and global GNSS trajectories to enable real-time anomaly detection and localization quality monitoring. The metric provides a measure of localization data consistency that can be used for adaptive sensor fusion or GNSS rejection strategies. Future work will focus on evaluating this method in diverse environments and integrating it into state estimation frameworks for improved robustness.

REFERENCES

- [1] K. Chen, B. T. Lopez, A. Akbar Agha-mohammadi, and A. Mehta, "Direct lidar odometry: Fast localization with dense point clouds," *CoRR*, vol. abs/2110.00605, 2021.
- [2] T. Feng, S. Chen, Z. Feng, C. Shen, and Y. Tian, "Effects of canopy and multi-epoch observations on single-point positioning errors of a gnss in coniferous and broadleaved forests," *Remote Sensing*, vol. 13, 2021.
- [3] T. Foote, "Rep 105: Coordinate frames for mobile platforms," *Open Robotics*, Tech. Rep., 2013, accessed: Mar. 4, 2025.
- [4] M. Grupp, "evo: Python package for the evaluation of odometry and slam," <https://github.com/MichaelGrupp/evo>, 2017.
- [5] D. Lee, M. Jung, W. Yang, and A. Kim, "Lidar odometry survey: recent advancements and remaining challenges," *Intelligent Service Robotics*, vol. 17, pp. 1–24, 02 2024.
- [6] D. C. Moore, A. S. Huang, M. Walter, E. Olson, L. Fletcher, J. Leonard, and S. Teller, "Simultaneous local and global state estimation for robotic navigation," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3794–3799.
- [7] P. Peltola, J. Xiao, T. Moore, A. R. Jiménez, and F. Seco, "Gnss trajectory anomaly detection using similarity comparison methods for pedestrian navigation," *Sensors*, vol. 18, no. 9, 2018.

Impressum

Eigentümer, Herausgeber und Verleger

Gesellschaft für Messtechnik, Automatisierung und Robotik – GMAR und
Automatisierungs- und Regelungstechnik Institut der TU Wien
Vertreten durch Prof. Dr. Markus Vincze
Gusshausstr. 27/376, 1040 Wien, Tel. +43/1/58801-376611
E-Mail: markus.vincze@tuwien.ac.at
Web: <https://www.acin.tuwien.ac.at/en/>

Redaktion und inhaltliche Verantwortung dieser Ausgabe

Wilfried Kubinger (kubinger@technikum-wien.at)
Simon Kranzer (simon.kranzer@fh-salzburg.ac.at)
Markus Vincze (markus.vincze@tuwien.ac.at)

Lizenz

©2025. This work is openly licensed via CC BY 4.0.

ISSN (Print): 3061-0729
ISSN (Online): 3061-0710
DOI: 10.34749/3061-0710.2025