ENABLING MOBILE ROBOT PERCEPTION AND SHARED-CONTROL WITH VISION-BASED SENSOR FUSION SYSTEMS

BY

YU CHEN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mechanical Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2023

Urbana, Illinois

ABSTRACT

Mobile robots with various locomotion mechanisms have been developed to adapt to different environmental conditions and mission requirements. The advancement in mobile robot development has raised the requirements for robot perception to ensure the safe and smooth execution of robot missions. In the past decade, the emergence of a novel self-balancing mobile robot system, known as the ballbot, has opened up new avenues of research in the field of mobile robots. While offering enhanced maneuverability, the under-actuated dynamics of the ballbot invalidate most existing ground robot perception solutions. Moreover, when the ballbot is designed as a riding mobility device, it imposes additional demands and constraints on robot perception and human-robot shared motion control. Specifically, the ballbot’s omnidirectional capability requires operators to possess significant proficiency and experience in riding and controlling the platform effectively. To address these challenges, there is a critical need for advanced and innovative robot perception techniques and higher-level control systems that can reduce the operator’s cognitive load and ensure system safety by preventing unwanted collisions. This dissertation aims to investigate robot perception and safeguarding challenges by leveraging vision-based sensor fusion techniques and commonly available commercial sensors. The research specifically focuses on developing robot perception and shared-control systems that can be deployed on a physical ballbot prototype, adhering to strict Size-Weight-and-Power Constraints (SWaP-C), while also being generalizable to mobile robots with simpler or similar dynamic characteristics as ballbots.

To my late grandfathers, whose legacies will continue to resonate and inspire.

ACKNOWLEDGMENTS

In a certain recess of my bookshelf, history resides. A commemorative R-Gator coin, a token from Prof. Norris, shimmers, echoing our shared journey dating back to 2018. A Starbucks gift card, a gesture from Prof. Liz, stands as a proud testament to our victory with the NSF proposal of 2020. Next to it, a champagne cork holds its ground, a memento of the triumph I celebrated with my colleagues, Chenzhang and Yinan, in the Human Dynamics and Controls Lab when the NSF grant was awarded to us. The remnants of academia, a broken acrylic specimen from a tensile test machine and some retired circuit boards from Dan Block's class, speak of my time as a teaching assistant. They sit comfortably next to an orange and blue mug, a symbol of my journey as a graduate teacher. This recess of my bookshelf runs deep, hosting memoirs from Michigan State University. Carefully folded cheat sheets huddle next to my first paycheck, a tribute to my time at Bush Lab, a haven where I met Prof. Bush, Prof. Roccabianca, Dr. Pan, Dr. Josh, Robert, Zac, and Lindsay. Giant dandelion seeds, mementos of road trips near Yellowstone with Qianli and Ruifeng, rest quietly. My late grandfather's Buddha beads, bestowed upon me on the day I left home and China, whisper his protection from far. There lies an amulet sent by my parents, which I was supposed to hang in my vehicle for a safe journey. Finally, a clutter of cranky gifts and souvenirs from my friends Mengchao, Hankun, Jiayue, Yuhan, Qiren, Weilin, and Peiqi, each a unique testament to our shared adventures, occupies a cherished space. They form a medley of the sentimental, the joyous and the heartrending. These are the talismans I seek in times of stress and despair. They echo a simple truth, reminding me that I was never alone in this enduring pilgrimage of academia.

Naturally, there are many cherished friendships and memories that can’t be reduced to physical mementos. I will forever cherish the kindness and support of Hang, Jifei, Jiaming, I-Chen, Grayson, and Marius, who I was fortunate to meet in the Autonomous and Unmanned Vehicle Systems Laboratory. My TA partner Ben, thank you for your friendship and for covering for me during our time teaching ECE470. My other work allies, Ze and Mahshid, Chapter 4 wouldn’t exist without your contributions. Leo, thank you for the bond of brotherhood we share, I wish you Godspeed on your journey back to the U.S. To my undergraduate students, Chirag, Zheyu, Skyler, Yintao, Zhanpeng, whom I was meant to mentor but ended up learning from and receiving help instead. A very special thanks to Coach Adam, Jeannette, Pat, and Prof. Deana. Without you, PURE would not have been a reality, this journey would not have unfolded. And, of course, to my dear parents and grandparents, our 11 years apart has only served to draw us closer together.

As I approach the final stretch of this journey, I must say, I could not have done this without you all, and indeed, I would not have wanted to do this without you!

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

A. Mobile Robot Perception

A “mobile robot” can be defined as a robot that can actively actuate itself in its operating environment [1]. Mobile robots can be classified by their means of locomotion. For example, iRobot’s Roomba [2] and the Segway [3] can be classified as wheeled robots; Boston Dynamic’s Spot [4] can be categorized as a legged robot that traverses the ground using articulated limbs. Depending on the complexity of the robot’s locomotion mechanism and the expected operating environment, mobile robots are required to have different levels of perception capability (collecting and processing information about the surroundings and the robot itself) such that they can interact with the environment properly and safely [5]. For instance, as a conventional wheeled robot, the Roomba is limited to 2D planar motions on structured and flat surfaces; hence it only needs wheel

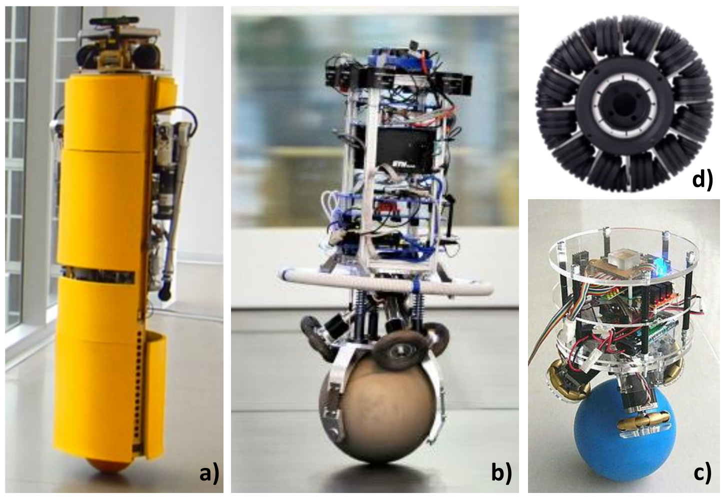


Fig. 1.1. Examples of modern ballbot designs a) CMU ballbot [7], b) ETH Rezero [9], and c) BallIP [8]. d) The omni-wheel used in this research study (OW125, SeCure Inc., China).

encoders, proximity sensors, and a compass system to navigate in its mission [2]. Whereas a legged robot such as Spot, which can perform dynamic motions like running and jumping in an uncontrolled environment, requires numerous sensors to measure the robot's surrounding and internal states. In other words, the more dynamic and complex movements that a mobile robot is capable of making, the higher environmental perception capability required for its locomotion [6].

B. Ball-based Self-balancing Robots

A ballbot [7]–[9] is a self-balancing mobile robot that rides on top of a ball instead of two coaxial wheels like the two-wheeled self-balancing riding device by Segway. Most ballbot drivetrain designs are made possible using three or more omni-wheels and a ball [8] (Fig.1.1a-c). The omni-wheel can be driven forward like any normal wheel, but its perpendicular mounted rollers (Fig.1.1d) allow it to slide sideways with minimal friction. By carefully arranging the contact angles between the omni-wheels and the ball, the drivetrain can move in any direction and spin in place. As a result, a ballbot has 5 Degrees of Freedom (DoF), which includes 2 DoF for horizontal planar translations and 3 DoF for rotations in each Euclidean axis [10]. This unique design gives the ballbot the ability to travel "omnidirectionally", and its self-balancing characteristic allows the ballbot to carry heavy payloads with a high center of mass while keeping a small footprint [7]. Leveraging its high maneuverability and minimum footprint, the ballbot can gain access to existing infrastructures that are designed for the able-bodied. Therefore, the ballbot is an ideal candidate for becoming the next-generation smart assistive/human transportation device.

C. PURE

Since the introduction of the modern powered wheelchair in the 1970s, it has played a crucial role as a personal mobility aid for individuals with various injuries and disabilities [11]. However, commercial power wheelchairs have largely remained rooted in the concept of motorizing a manual wheelchair. The traditional differential-drive kinematics used in these wheelchairs restricts their ability to perform lateral movements. Additionally, in order to maintain static stability and minimize tipping risks, most power wheelchairs have large footprints, which makes navigation in domestic environments challenging. Furthermore, these wheelchairs typically require a joystick interface for control, occupying the operator's hands during navigation. We are working on a ballbot that is designed to assist humans with transportation needs, i.e., PURE (Personalized Unique Rolling Experience, Fig.1.2). PURE is modularized such that the top section will be easily convertible between human-riding mode (Fig.1.2b-c) and payload-carrying mode (Fig.1.2a). The drivetrain and chair frame of PURE are constructed using aluminum and titanium alloy materials. Additionally, a titanium protection ring is installed near the bottom of the drivetrain to support the system and offer protection in case of a power outage or unexpected events. The ball used in the system has a diameter of 23 cm. It is coated with 3D-printed TPU (thermoplastic polyurethane) patches and has a polyester core. This combination of materials provides durability and traction. PURE allows for two forms of control: direct physical interaction (leaning of the torso while riding or pushing/pulling the ballbot), and remote command (via a joystick controller). PURE is

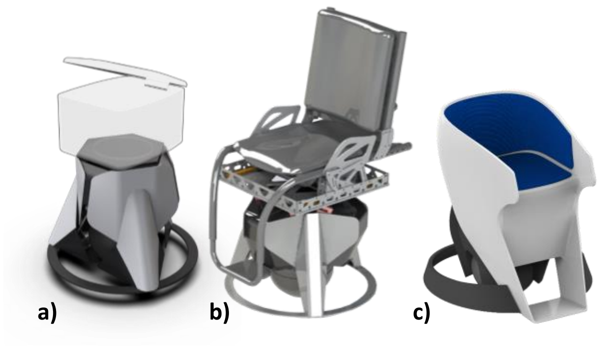


Fig. 1.2. a) PURE in payload carrying mode, b) PURE Gen2 prototype in human riding mode, and c) a concept drawing for a future generation PURE in human riding

proposed to adapt to the weight and movement of the rider or payload and different riding terrains/environments.

However. the omnidirectional motion capability, minimum form factor, and under-actuated dynamics [12] pose unique robot perception challenges that cannot be easily solved with existing methods. These difficulties mainly arise from four aspects. First, PURE’s ability to spin in place and travel in both longitudinal (anterior-posterior) and lateral (left-right) directions requires the system to have a wide Field-Of-View (FOV) to prevent potential collisions. Second, when PURE accelerates, it will tilt towards the direction of acceleration due to the under-actuated dynamics [13], which invalidate the common assumptions that the ground plane is always at a fixed distance and parallel to the robot frame [14]. Third, PURE’s small form factor restricts the battery, computation power, and sensor systems that can be carried. Thus, size, weight, and power (a.k.a. “SWaP-C” [15]) are important constraints to consider. And fourth, having only one point of contact with the floor, PURE is more susceptible to instability while driving over small ground objects and rough surfaces compared to conventional statically-balanced robots. This design requires the perception system to pay close attention to the surface’s abnormalities and traversability. On the other hand, controlling and operating a 5 DoF system can be mentally demanding as it may require a certain level of user proficiency, not to mention compensating for additional complexity imposed by the environment.

D. Vision-based Sensor Fusion and Shared-Control

In the field of assistive robotics [16], previous efforts have helped realize applications such as smart wheelchairs [17]–[34], smart walkers [35], [36], intelligent canes [37], and vision-based navigation vests for the visually impaired [38]. In these examples, multiple sensor sources were used to detect and understand the robots’ surroundings and states for enabling intelligent behaviors and assistive features. Using multiple sensor sources comes with numerous advantages, including learning diverse aspects of an environment, improving measurement accuracy or speed, expanding the sensing limits of the sensors' hardware, and gaining a level of robustness against the risk of failure [39]. These benefits give rise to the study of sensor (data) fusion. And the key is to find efficient ways to combine sensor data and contribute positively to the task of perception. If properly designed, a sensor fusion system can achieve reasonably high performance and robustness without the need for expensive, heavy, bulky, and power-intensive lab-grade sensors and computers that violate the SWaP-C constraints.

Traditional smart wheelchairs [17]–[34] are statically balanced devices, meaning they do not require active balancing maneuvers to remain upright. This is because their supporting structures, such as wheels or legs, maintain at least three points of contact with the ground. As a result, the relative orientation between the wheelchair's sensor frame and the ground surface can be assumed to be fixed. When operating in structured and flat environments, the transformation between the

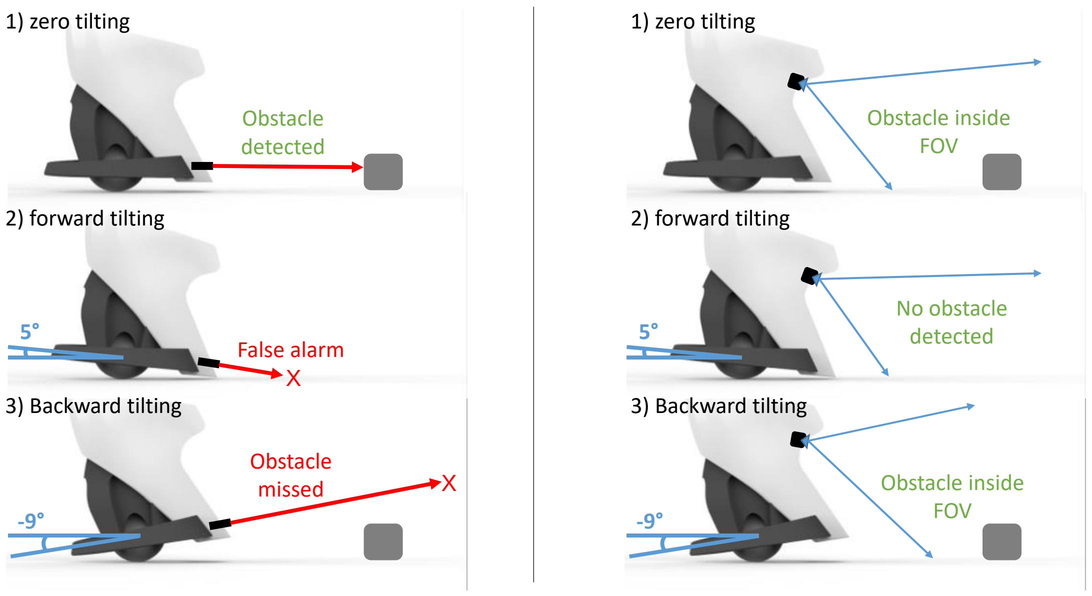


Fig. 1.3. A comparison between low-mounted proximity sensor (left) and side-mounted RGB-D camera (right) under 1) zero tilting; 2) forward tilting; and 3) backward tilting.

sensor frame and the ground plane is known, simplifying obstacle detection. Low-mounted proximity sensors with low clearance to the floor can be used to detect small obstacles without the risk of misclassifying the floor as an obstacle.

As mentioned, PURE’s unique omnidirectional maneuverability and dynamic stability pose challenges. It is difficult to anticipate the direction or tilt angle at which PURE will encounter obstacles. A low-mounted proximity sensor may generate false alarms by mistaking the floor as an obstacle or may fail to detect an obstacle if its projector is facing upward, depending on the tilting conditions (Fig.1.3 (left)).

In response to the challenges posed by PURE’s unique omnidirectional maneuverability and dynamic stability, a perception system with a wide FOV and the ability to differentiate between the ground plane and obstacles is necessary. There are designs as in [19], [28]–[30], [32] which add top-mounted cameras or LiDAR to gain a wide or 360-degree view of the surroundings. However, these designs are more suited for research and development purposes instead of real-world product deployment. These designs often include tall mounting poles that are socially unacceptable and increase the risk of overhead collisions, limiting user mobility.

To address these issues, this dissertation focused on the development of a low-profile, side-mounted sensor array for PURE. Sensor placement locations were carefully chosen to maximize the FOV while minimizing interference with the user’s body and chair-transfer motion. Vision-based sensor fusion systems, renowned for their wide FOV, ability to capture dense visual information, and measure 3D structures [40], offer promising solutions for ground robots like PURE [7]–[9], [41]–[43]. This dissertation aimed to explore the potential of a vision-based sensor fusion system as a versatile robot perception solution and to develop shared-control prototypes enabled by this system. In this context, motion control of a semi-autonomous device that is shared between a human operator and the device will be referred to as “shared-control”. Studies [44], [45] have highlighted the importance of utilizing the user’s residual abilities for promoting independence, suggesting that complete autonomy might not be desirable. Alternatively, shared-control is a software system that offloads low-level motion control, such as obstacle avoidance tasks, to a robot controller. This allows the human operator to focus on high-level navigation objectives, reducing cognitive load [11], [46]–[48]. This collaborative approach combines the strengths of both the user and the robotic system, enhancing overall performance and promoting independence.

Prior powered wheelchair shared-control approaches can be broadly categorized into deliberative and reactive methods. Deliberative shared-control follows a “Sense-Plan-Act” formulation, utilizing environment modeling and planning techniques such as Rapidly Exploring Random Trees (RRT) [49], Model Predictive Control (MPC) [50], [51], Reinforcement Learning (RL) [52], or Bayesian methods [53]. These approaches consider both current and future states of the environment to plan efficient and risk-averse paths. In contrast, reactive shared-control directly maps the current state of the environment and the robot into the robot’s control signal. Examples of reactive shared-control methods include Vector Field Histograms (VFH) [54], Dynamic Window Approach (DWA) [55], and Artificial Potential Fields (APF) [47], [56]–[59]. Reactive shared-control is known for its speed, simplicity, and robustness but lacks the ability to reason about complex navigation tasks and find optimal paths. While these existing methods have demonstrated success in enhancing user abilities to operate or remotely control powered walkers and wheelchairs, there is a lack of shared-control research targeting dynamically balanced robots like PURE.

1.2 Objective and Research Significance

The main focus of this dissertation research was to enable robot perception and intelligent behaviors (e.g., shared-control) on mobile robots, by leveraging vision-based sensor fusion. Previous literature addressing Robotic Terrain Classification (RTC) for navigation and surveying has tended to focus solely on either exteroceptive sensors (i.e., sensors that perceive the environment external to the system) or proprioceptive sensors (i.e., sensors that perceive the status and conditions internal to the system) [60]–[66]. However, these approaches lacked complementary sensor modalities and redundancy, making them susceptible to influences from the environment or vehicle motion [67]–[69]. Moreover, existing neural network models for semantic/contextual understanding, while effective and powerful, are often computationally and power expensive [70]. This renders them unsuitable for deployment on robots with SWaP-C constraints, which categorize most indoor robots as they require compact form factors to work in clustered and confined spaces. Traditional shared-control techniques used on smart wheelchairs were designed to fuse operator (i.e., joystick signals) and robot commands on statically-balanced conventional robot platforms [36], [71]–[73]. However, when it comes to dynamically-balanced robots like the ballbot, the operator might be able to overpower the robot agent’s commands by leaning in the opposite direction (as the ballbot needs to track the center of gravity to maintain balance). This implies that these existing techniques cannot be directly transitioned onto PURE.

This dissertation presents three key studies designed to answer the following research questions:

1. How to build a vision-based sensor fusion method that enables robust robot terrain classification on a miniature ground vehicle?

2. How to address the perception challenges of a riding ballbot using a vision-based sensor fusion system?

3. How to transform and transition the developed vision-based sensor fusion system and shared-control onto the riding ballbot system and effectively safeguard the riding operator?

Study 1 - Investigation of a Vision-Proprioception Fusion Method for Robust Robot Terrain Classification (RTC)

The first study used a commercial Unmanned Ground Vehicle (UGV) mobile platform to build a Robotic Terrain Classification solution based on common onboard UGV sensors. The goal was to reduce operation restrictions on environmental illumination and vehicle maneuverability while developing a compact and motion-and-illumination-robust RTC solution capable of accurate and reliable classification under various lighting conditions and motion maneuvers. This work[*] has been presented at the 2021 IEEE IROS conference and published in [74].

Study 2 - Development of a Neural Network Augmented RGB-Depth Perception System for Indoor Robot Navigation

The goal of the second study was to develop a robot perception solution that addresses the defined perception requirements and constraints of a ballbot system. The outcome of this study was a perception system that generates a multi-layer grid map representation summarizing the information on surrounding obstacles and traversable surfaces. The system had several design requirements: 1) it should remain unaffected by PURE’s tilting motion during operations; 2) it should be capable of detecting both large and small objects on the floor that may not be easily detected by depth sensors alone; 3) it should have low latency to ensure real-time responsiveness;

and 4) it should be deployable to a low-power single-board computer, meeting the SWaP-C constraints of PURE. The pre-print of this study is available [75].

Study 3 - Integration and Validation of a Vision Guided Shared-Control System for a Riding Ballbot

The third study focused on transitioning the perception system onto PURE by integrating the high-level perception system with the low-level dynamic control system of PURE, and demonstrating the system's ability to perform vision-based shared-control. This study details the full system layout, as well as a human-robot interaction evaluation designed to evaluate the efficacy of the shared-control paradigm. The shared-control algorithm detects obstacles and slow or stop the device to protect the PURE system and user from unwanted collisions, aiming to reduce the users' cognitive load. The requirements for the shared-control were: 1) it needs to be reactive with low latency; 2) it needs to be passive, avoiding control oscillations and allowing for soft collisions; 3) it needs to accommodate PURE's dynamic characteristics. Twenty subjects, including ten able-bodied individuals and ten manual wheelchair users, were recruited for the human-robot interaction test. Test participants were asked to ride the device at low speed (≤ 1 m/s) with and without the assistance of the shared-control in test courses designed to simulate indoor navigation scenarios (e.g., S-turn and Zigzag) to assess the ability of shared-control to minimize collisions (with cardboard walls). This study represents the first attempt at implementing a shared-control system for a ridable self-balancing device.

*The majority of the content in this chapter was previously published in [73] as "A CNN-based vision-proprioception fusion method for robust UGV terrain classification" by Chen, Y., Rastogi, C., and Norris, W.R. (2021) in IEEE Robotics and Automation Letters, Volume 6, Issue 4, pp. 7965-7972. Digital Object Identifier 10.1109/LRA.2021.3101866. Minor edits have been made to integrate it more effectively into this dissertation.

CHAPTER 2

INVESTIGATION OF A VISION-PROPRIOCEPTION FUSION METHOD FOR ROBUST ROBOT TERRAIN CLASSIFICATION

2.1 ABSTRACT

The ability for ground vehicles to identify terrain types and characteristics can help provide more accurate localization and information-rich mapping solutions. Previous studies have shown the possibility of classifying terrain types based on proprioceptive sensors that monitor wheel-terrain interactions. However, most methods only work well when very strict motion restrictions are imposed including driving in a straight path with constant speed, making them difficult to be deployed on real-world field robotic missions. To lift this restriction, this study proposes a fast, compact, and motion-robust, proprioception-based terrain classification method. This method used common on-board UGV sensors and a 1D Convolutional Neural Network (CNN) model. The accuracy of this model was further improved by fusing it with a vision-based CNN that made classification based on the appearance of terrain. Experimental results indicated the final fusion models were highly robust with strong performance, with over 93% accuracy, under various lighting conditions and motion maneuvers.

2.2 INTRODUCTION

The development of precision agriculture has been greatly enhanced due to advances in sensors, robotics, and artificial intelligence. For instance, automatic inspection robots can make assessments of the soil (terrain) quality of farmland and use that information to optimize the scheduling of farming tasks including plowing, watering, and fertilization. Similarly, scout and surveillance robots can be used to explore sites of interest remotely to identify hazardous terrain that might sink and trap or damage heavy machinery. They can provide reference data for excavation in construction and mining operations. These robotics applications all rely on the robot's ability to perceive and characterize terrain. Additional benefits of terrain classification include the development of terrain-aware traction control and motion planning that minimizes fuel consumption for field robots. The knowledge of site-specific terrain characteristics can improve localization accuracy by serving as landmarks, and contributing terrain-related information on a high-definition map.

Previous efforts to address robotic terrain classification (RTC) have explored the use of exteroceptive sensors such as cameras [60], [61], 2D and 3D laser scanners [62], [63], ultrasonic and infrared sensors [64], as well as microphones [65]. In other studies, proprioceptive sensors such as Inertial Measurement Units (IMUs) [66] and acceleration or vibration sensors [68], [76] were used to characterize terrain properties. In terms of the classifier, prior studies have investigated both traditional machine learning techniques (i.e., SVM [77]–[79], kNN [80], and Bayes model [66], etc.) and artificial neural networks (i.e., LSTM [65], RNN [76], CNN [81], FCN [68], etc.), which have led to immense successes in solving RTC problems.

The key to a robust RTC solution is to extract terrain characteristics that are invariant to vehicle motions and environmental factors such as noise and unstable illumination. Vision-based methods are well-studied and relatively more accurate, but are susceptible to influences from environmental illumination [67], and motion blur caused by strong vibrations. Proprioceptive solutions are robust to environmental factors and require only common sensors used on most modern robots. However, the solutions are less accurate and highly dependent on the vehicle's motion. Many previous studies[78]–[83] tested their methods when the UGVs move on a straight path with constant speed, while others did not reveal their testing conditions. These restrictions occurred because the vehicles used in these studies were primarily skid-steer drive, which induced additional slippage and vibration to the system while turning. Also, the vehicle's driving speed is proportional to the driving effort and the frequency of the vibrational response [68], [84]. These motion-dependent interferences cloud the judgment of the proprioception classifiers. To overcome the shortcomings in different sensing modalities, previous studies considered leveraging data from multiple sensor sources. For example, [61] used the visual and texture features gathered by a stereo camera, vibration sensors, and a belly-mounted camera to improve the classification performance between three terrain classes. The study in [78] fused the classification decisions made by an image-based SVM and a vibration-based SVM on a 14-class terrain classification problem. And more recently, [79] combined color and three different proprioceptive features to assess terrain in an agriculture setting. While the aforementioned work demonstrated improved performances using multi-model classifiers, they did not fully address the problem of motion-dependent interference of the UGV nor quantify the robustness of their models under challenging lighting conditions.

In this research, a fast, compact, and motion-robust proprioception-based classifier using common on-board UGV sensors and a 1D CNN model was developed. Previous work like [66], tackled the motion-dependent interference problem using a hierarchical classifier and performing feature selection on hand-crafted features. As opposed to this approach, the CNN model in this study learns the most effective features from the input data directly. Using this unified pipeline, over 90% accuracy was achieved on data sets recorded under arbitrary and continuous vehicle motions with minimum knowledge in the signal processing and proprioception domains. Furthermore, this study showed that by adopting and fusing an image-classifying CNN module pre-trained on a different data set, the classification accuracy was over 98% under appropriate illumination. The same approach maintained a robust performance of over 93% accuracy under challenging lighting conditions.

The following chapter is organized as follows: Section 2.2. provides implementation details of a novel terrain classification approach, including the data set collection and feature generation process, as well as the construction and training of the neural network models. Section 2.3. presents the experimental results, and the performance of the neural network models under realistic and challenging conditions are demonstrated and discussed. Section 2.4. concludes the study and provides insights towards future work.

2.3 METHODOLOGY

A. Overview

The proposed proprioception model uses similar procedures from [79] to generate proprioceptive signal inputs. As opposed to the model in [79], which had no knowledge of the vehicle's motion, wheel encoder readings were fed to the model to serve as motion cues. Also, instead of fully relying on the proprioceptive signals of the vehicle body, this study utilized the signals from the left and right sides of the vehicle to help the classifier reject motion-dependent interference. The approach used in this study retained the input signals in their raw form. So that the neural network model can learn the temporal correspondences between the vehicle's motion and proprioceptive feedback.

A vision-based classifier was built using a pre-trained module and was later fused with the proprioception-based classifier. This study investigated two different fusion mechanics and demonstrated that visual and proprioceptive signals were complementary. In combining their modalities, the classification performance and robustness can be improved. The model's ability to function across different motion and lighting conditions makes it more suitable for real-world mobile robotic missions on arbitrary outdoor fields which oftentimes require obstacle avoidance maneuvers.

All the data used in this study was collected using a mobile robotic platform and processed off-line. Once trained, the classifiers can be deployed on the robot and report a terrain label every second during normal operations (0.2 m/s to 1 m/s).

B. *Data Collection*

The Jackal UGV, from Clearpath Robotics, was used as the data collecting platform. This skid-steer four-wheel-drive vehicle, shown in Fig.2.1a, comes with an onboard IMU, two DC motors

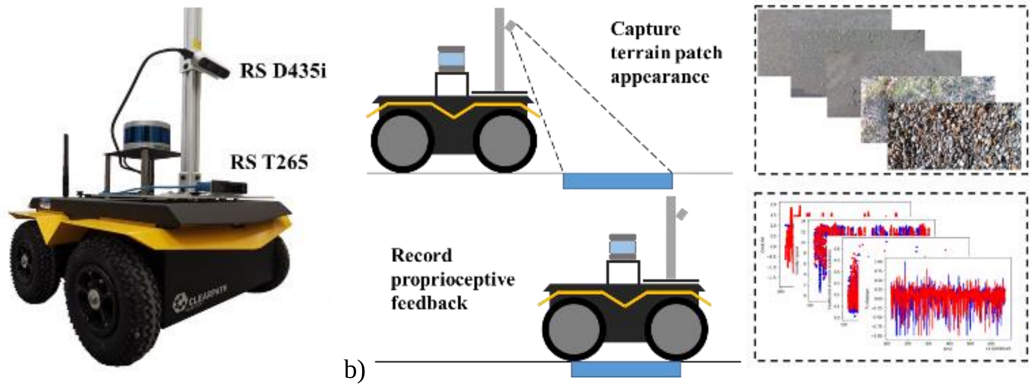

RS D435i
RS T265
Capture terrain patch appearance
Record proprioceptive feedback
a)
b)

Fig. 2.1: a) The Jackal robot platform used for data collection; b) The data collection pipeline.

with encoders that measure wheel angular speeds, and current sensors that measure motor current outputs. On each side of the robot, the front wheel and back wheel are jointed with a gearbox and so spin together at the same rate and direction. The IMU provides vehicle attitude measurements in terms of Euler angles, as well as linear acceleration and angular rate of the vehicle body in three Euclidean axes. Camera systems such as the RS D435i and RS T265 were mounted to an aluminum frame attached to the top of the robot platform. While the tracking camera T265, faced forward, the D435i depth camera was positioned and tilted in a way that the camera had a clear visual of the terrain patch. The patch size was about 680 mm × 340 mm with a look ahead distance of 150 mm relative to the chassis of the vehicle. The D435i served as a regular RGB camera for this study and the depth images reconstructed by the D435i were not included in any of the data sets. The T265 camera was used as a Visual-Inertia Odometry (VIO) solution that provided ego-motion estimations of the vehicle.

For this study, six different sensor signals were used: 1) current feedback, 2) wheel encoder readings from each side of the vehicle, 3) 6 DoF VIO measurements from the T265, 4) three-axis linear acceleration, 5) attitude measurement from the IMU, and 6) RGB images taken by the D435i. In addition to the RGB images, all other sensor signals were used as proprioceptive features.

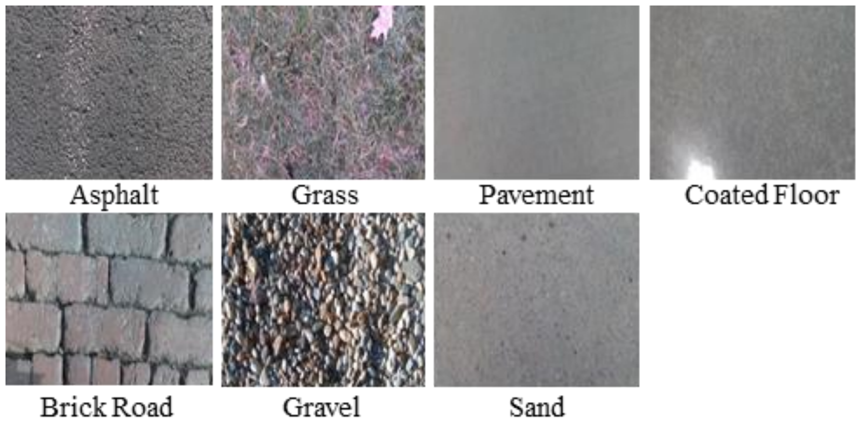


Fig. 2.2: Sample images of the different terrain classes.

Seven terrain classes were investigated, including asphalt, brick road, grass, gravel, pavement, sand, and coated floors. Fig.2.2 shows sample images of the different terrain classes in the data sets.

Training and development data were collected by driving the Jackal robot on each terrain class across different days to account for variance in lighting conditions. Data collection occurred during sunny days from Nov. 2020 to Feb. 2021 in Champaign, Illinois. The average temperatures ranged from 6 to 27 °C. An experienced human operator (one of the authors) was designated for remote control of the Jackal robot during data collection. For each terrain class, multiple independent trials were taken for at least 6 minutes for the *Straight Driving* sessions and at least 10 minutes for the *Remote Control* sessions. The protocols of these sessions were defined as follows:

- *Straight Driving*: the robot was programmed to follow a straight path at the speed of 0.5 m/s and 1 m/s with no reverse driving, stopping, or turning. This session was added to help the neural network understand the proprioceptive baseline for each terrain surface without the interferences induced by aggressive vehicle movements.

- *Remote Control*: the robot arbitrarily drove around the test site to simulate normal robot operations, controlled by a human operator remotely. The operator was instructed to take each path as randomly as possible to reduce the motion bias in the data set. The session conditions included no reverse driving, stopping only if necessary, with a top speed limited to 1 m/s out of safety concerns.

Finally, 600 sec of data from the *Straight Driving* sessions and 1800 sec of data from the *Remote Control* sessions were randomly chosen to form the data set for each terrain class. In total, a data set that contained 7 terrain classes × (600 s + 1800 s) = 16,800 s (4.67 hours) of image-signal data was gathered for training the neural network models. This data set was divided into a training set and a development set with a ratio of [4:1]. The split between the training and development set was a uniform random selection from the shuffled data set. And for both data sets, the number of samples in each class was kept equal to prevent uneven training.

C. Data Processing

1) Proprioceptive signals: Raw sensor signals were sorted and stored in the form of 1-second data segments as conducted in many previous studies [61], [66], [68], [76]–[78], [80], [83].

Resampling Zero-Order-Hold (ZOH) interpolation and subsampling were used to make sure signals from different sources shared the same sampling frequency (100Hz). After the resampling, one sample of the proprioceptive signal was a $100 \times n$ vector, where n was the number of signal channels. The use of ZOH ensures easy transfer when deploys on-line.

Data Cleaning All the data segments that contained stopping motions, where vehicle linear velocity < 0.2 m/s were removed, as the proprioception-based method was not effective under this condition.

Feature Generation The following proprioceptive features were selected/derived from raw signals: wheel angular speed, motion resistance coefficient, and percentage slip for each side of the vehicle. Also, the linear acceleration of the vehicle body in three Euclidean axes. A total of $2\times(1+1+1)+3=9$ proprioceptive features were used as input to the *Proprioception Net*. The details of the feature derivation are provided in section 2.2.D.

2) RGB images: RGB images were recorded at 25 Hz by the D435i. There were 25 images for each second of proprioceptive signals.

Data Association The approximation was made such that only the first image of a second was used to represent the appearance of the terrain patch that the robot was about to traverse, as shown in Fig.2.1b. Since the robot primarily operated with a speed between 0.5 m/s to 1.0 m/s, and each data collection site contained only one terrain class, the assumption was that the fixed frame approximation was sufficient. The correspondence between image and proprioceptive signals did not need to be exact for this application. Ideally, techniques like Simultaneous Localization and Mapping (SLAM) would be applied to provide more accurate image-signal data association to account for misalignments due to differences in speed and steering.

Inverse Perspective Mapping (IPM) IPM was applied to the selected RGB images to transform them into clear and homogeneous bird's-eye view terrain patch visualizations (500 × 250 pixels). The transformed images were resized to 224 × 224 pixels to fit the input size of the *Vision Net*.

3) *HDF5 File*: The processed image-signal pairs were stored in the HDF5 format for fast retrieval. The data pairs were organized by their unique timestamps. The HDF5 data files used in this study are open-source and available at IEEE DataPort:

https://ieee-dataport.org/open-access/jackal-robot-7-class-terrain-dataset-vision-and-proprioception-sensors

D. *Proprioception Net*

1) Feature Derivation: Among the proprioceptive features used in this study, the wheel angular speed (rad/s) and the three-axis vehicle linear accelerations (m/s^2) were taken in their raw forms. The derivation and definition of the motion resistance coefficient was adopted from [79]: The motion resistance caused by the deformation of the wheel-terrain interface shifts F_z, the vertical load experienced by the wheel

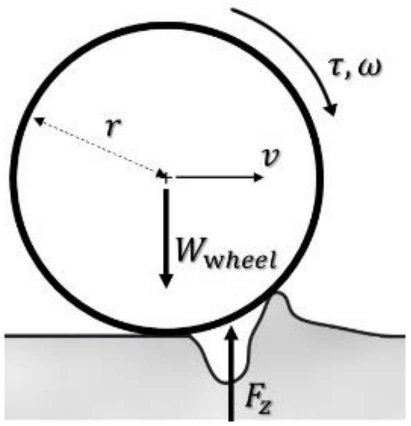


Fig. 2.3: Motion resistance that arises from wheel-terrain interaction.

forward with respect to the wheel's geometric center, as shown in Fig.2.3. Assuming all of the torque generated by the motor is used to overcome the resistance moment, the required driving torque is $\tau = f_r \, r \, F_z$, where f_r is the motion resistance coefficient on a local terrain patch, and r is the radius of the wheel.

From the motor's perspective, the output torque can be roughly estimated by the amount of current, I, drawn by the DC motor: $\tau = \varepsilon \, k_t \, I$. ε is the gear ratio and k_t is the torque constant of the DC motor. Therefore, the motion resistance coefficient can be estimated using current feedback and wheel vertical load:

$$f_r = \frac{\varepsilon k_t}{r} \frac{I}{F_z} \tag{2.1}$$

The wheel vertical load is not a variable that can be directly measured or easily calculated. However, the variable can be approximated using a quasi-static dynamic model, proposed by [79], to neglect complex inertial effects caused by vehicle motion. This approximation is valid as the Jackal robot only operates at a relatively low speed with a maximum of 1 m/s. Applying Newtonian mechanics, the vertical forces in the vehicle coordinate frame can be expressed. As this study was only concerned with the motion resistance coefficient on each side of the vehicle, as shown in Fig.2.4, the vertical forces for the left and right sets of the wheels are:

$$F_{z,l} = \frac{W}{2} cos(\phi) \, cos(\theta) \; - \; W cos(\theta) \, sin(\phi) \frac{h}{D} \tag{2.2}$$

$$F_{z,r} = \frac{W}{2} cos(\phi) \, cos(\theta) \; + \; W cos(\theta) \, sin(\phi) \frac{h}{D} \tag{2.3}$$

ϕ and θ are the vehicle's roll and pitch angles measured by the onboard IMU. h is the height of the vehicle's center of gravity, and D is the track width. Additional details on the derivation of (2.2) and (2.3) can be found in [79].

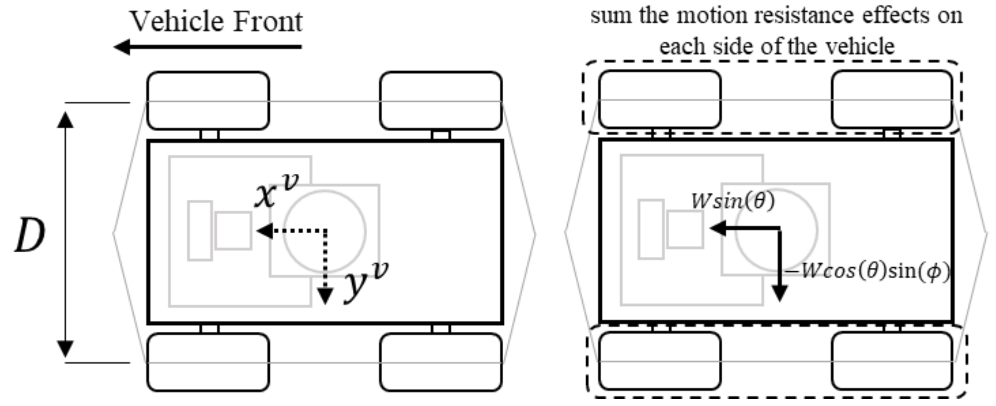

Vehicle Front
sum the motion resistance effects on each side of the vehicle
D
x^v
y^v
$W sin(\theta)$
$-W cos(\theta) \sin(\phi)$

Fig. 2.4: Free-body diagram of the vehicle system (top-view)

The above method is a rough approximation of the motion resistance that the vehicle experiences over a local terrain patch, as the noise in current feedback, the friction loss in the gearboxes, and inertial effects are not accounted for. However, the results showed this estimation was sufficient for use as an indicator of the terrain hardness level.

The percentage slip of the left and right sides of the vehicle are defined as follows:

$$\%slip_l = 1 - \frac{\omega_{vio,l}}{\omega_{enc,l}} \tag{2.4}$$

$$\%slip_r = 1 - \frac{\omega_{vio,r}}{\omega_{enc,r}} \tag{2.5}$$

While $\omega_{enc,l}$ and $\omega_{enc,r}$ are the left and right wheels' angular speeds measured by the encoders, $\omega_{vio,l}$ and $\omega_{vio,r}$ are the left and right wheels' angular speeds calculated from the VIO estimations and skid-steer vehicle kinematics:

$$\omega_{vio,l} = (v_{vio,x} - \frac{D}{2} w_{vio,z})/r \tag{2.6}$$

$$\omega_{vio,r} = (v_{vio,x} + \frac{D}{2} w_{vio,z})/r \tag{2.7}$$

$v_{vio,x}$ and $w_{vio,z}$ are the vehicle linear speed on the x-axis and vehicle angular rate on the z-axis, both estimated by the T265. As opposed to [79], which took the average of the motion resistance coefficient and percentage slip of the whole vehicle to form their proprioceptive features, this study retained the values across the data segment window. To form the desired motion-robustness, the model needed to learn how to distinguish and eliminate motion-dependent interference from noisy proprioceptive features. As a result, the model was given the temporal correspondences between the robot's motions and proprioceptive feedback. This is the reason why this study did not use Fast Fourier Transform (FFT) when handling acceleration data like many previous studies [68], [77], [78], [80], [83], [84]. Doing so would essentially destroy the underlying temporal information within the data segment.

Finally, at each time step k, a vector $\mathbf{p_k}$ (1×9) that contained nine proprioceptive features $\{\omega_{enc,l}$, $\omega_{enc,r}$, $accel_x$, $accel_y$, $accel_z$, $\%slip_l$, $\%slip_r$, $f_{r,l}$, $f_{r,r}\}$ was drawn, where $accel_x$, $accel_y$, and $accel_z$ were the three-axis vehicle linear accelerations. And $f_{r,l}$, $f_{r,r}$ were the motion resistance coefficients for the left and right sides of the vehicle. These proprioceptive features informed the model about the vehicle motion, as well as the evenness, slipperiness, and motion resistance of the terrain. Finally, the input to the *Proprioceptive Net* was a 2D vector **p** (100×9), where there were 100 time steps in a one-second data segment and with 9 feature channels.

2) Model building: This study explored using 1D CNN, Multi-branch CNN, CNN with skip-connections, Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), CNN-LSTM [85],

TABLE 2.1
THE PROPRIOCEPTION NET MODEL STRUCTURE

Layer type	Kernel Size / Stride	Output shape
Conv1D	9*9 / 1	(100, 64)
Max Pooling	2*2 / 2	(50, 64)
Dropout (0.2)	-	(50, 64)
Conv1D	5*5 / 1	(50, 64)
Max Pooling	3*3 / 3	(16, 64)
Dropout (0.4)	-	(16, 64)
Flatten	-	1024
Softmax	-	7

and ConvLSTM2D [86] as building blocks of the *Proprioceptive Net*. For a similar number of parameters with the same order of magnitude, the 1D CNN provided the best results. The best model was a simple two-layer 1D CNN (with 32,967 parameters). The detail of the network structure is provided in Table 2.1.

This 1D CNN model performed convolution along the temporal axis, and a Rectified Linear Unit (ReLU) was used at each convolution layer as the activation function. Many previous Natural Language Processing (NLP) studies applied similar practices and achieved great success. Recurrent neural networks like LSTM and GRU were also heavily used in NLP [87]. For this task, while keeping the number of parameters within the same order of magnitude as the 1D CNN model, increasing the number of recurrent units per layer, and the depth of the network did not result in an obvious improvement in accuracy. A similar result was observed in [88], when using LSTM to process haptic signals.

For other hybrid models, the difficulties of structural arrangement and hyper-parameter tuning increased proportionally with their model complexities. Since a simple 1D CNN model achieved a high level of accuracy, the implication is that the features used in this study were very effective for extracting motion-independent terrain characteristics. As a result, there are marginal benefits in using a “deeper” model.

E. Vision Net

To further improve the accuracy of the classification, a *Vision Net* model for capturing the visual characteristics of terrains was constructed. Generally, the training of an image-classifying CNN requires a large amount of data to ensure generalizability and prevent overfitting. To get high performance with a limited amount of data, the *MobileNet v2* module [89] that was pre-trained on

ImageNet was adopted. *MobileNet v2*, a compact and efficient CNN architecture developed by Google, was designed for image classification on a device with limited computational power like single-board computers or mobile phones. ImageNet is a large image database that contains multi-millions of image samples.

The weights of this pre-trained CNN module were frozen to reduce the number of trainable parameters (41,223) and save training time. Numerous data augmentation techniques were applied, such as horizontal and vertical flipping as well as random rotation, and random brightness on the training data. Using the model shown in Table 2.2, the *Vision Net* achieved over 98% accuracy on both training and development sets.

TABLE 2.2
THE VISION NET MODEL STRUCTURE

Layer type	Kernel Size / Stride	Output shape
MobileNet v2	-	1280
Dropout (0.25)	-	1280
Dense	-	32
Softmax	-	7

F. Fusion Net

The hypothesis was made that by fusing the proprioception- and vision-based model, higher levels of performance would be achieved due to their complementarity. To better understand the fusion mechanics, two different fusion schemes were explored, namely the feature map-level and decision-level fusion.

1) Fusion at the feature map-level: First, the Softmax (the last) layer of the trained *Proprioception Net* was removed. As a result, the output of this module was a 1×1024 feature vector. This module was denoted as a Proprioception CNN. The pre-trained *MobileNet v2* module was used and denoted as a Vision CNN. The weights of these two modules were frozen, and they were used as feature extractors to process the image-signal hybrid input. The activations (ReLu) from these two CNN modules were concatenated. The complete pipeline of feature map level fusion can be found in Fig.2.5.

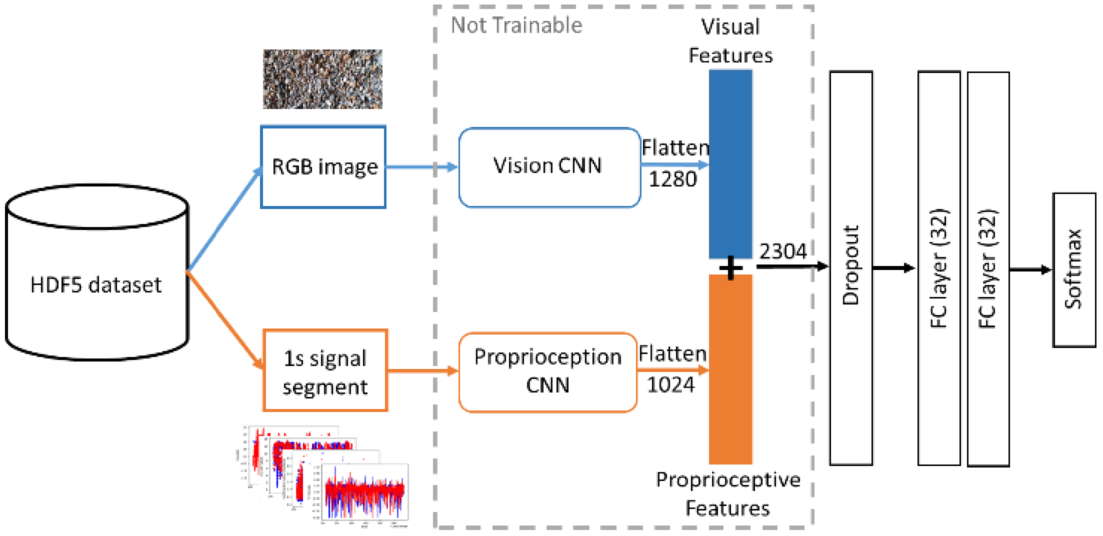


Fig. 2.5: Feature map-level fusion model pipeline

To simulate the possible illumination conditions (i.e., darker lighting, etc.) in test time, additional and aggressive data augmentation techniques were applied on images during training, including random channel shift, motion blur, and blackout.

2) Fusion at the decision-level: In this model, the learned weights of the *Vision Net* and the *Proprioception Net* were directly transferred. These two networks ran in parallel and made

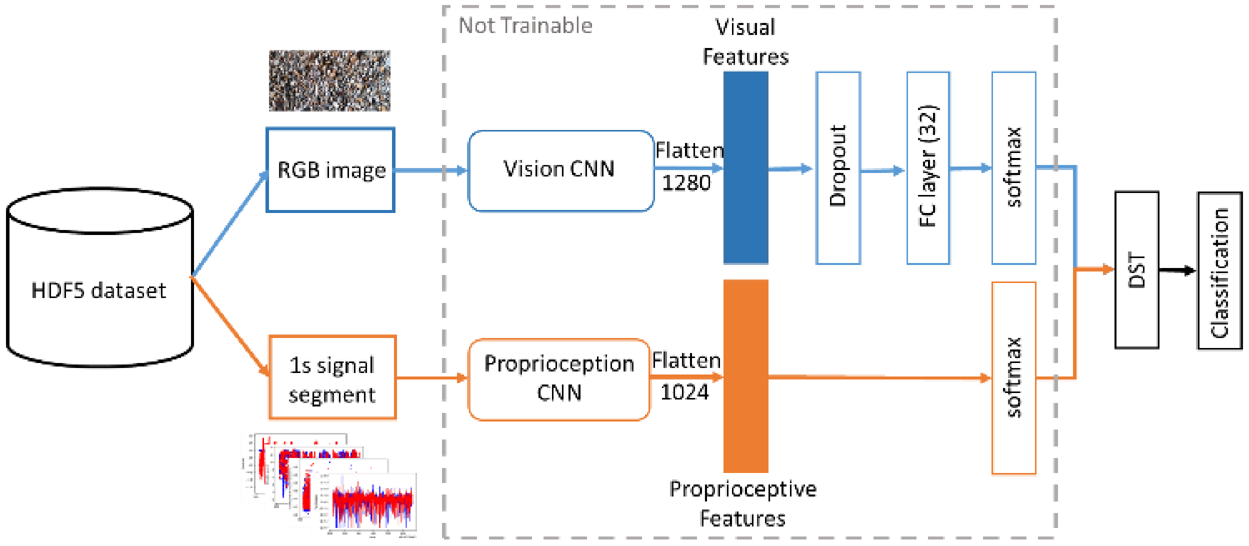


Fig.2.6: Decision-level fusion model pipeline

independent predictions (decisions) at their Softmax layers. The prediction outcomes were received by a fusion operator, denoted as a DST, which outputs the final classification, as shown in Fig.2.6. DST is the acronym for the *Dempster–Shafer Theory* [90]. It was used in this model to solve the problem of combining multiple belief functions. DST was used in a previous fusion model [91] for classifying sound signals with great results.

In this study, two distinct sets of evidence, appearance and proprioceptive feedback, were used to estimate the belief about an event. The belief was the likelihood that a certain terrain type was detected. For this application, Dempster's rule of combination was an appropriate fusion operator. Specifically, the operator was defined as follows:

$$m_{1,2}(A)=(m_1 \oplus m_2)(A)$$

$$=\frac{1}{1-K}\sum_{B\cap C=A\neq\emptyset} m_1(B)m_2(C) \tag{2.8}$$

$m_1(B)$ and $m_2(C)$ were the mass functions of the *Vision Net* and the *Proprioception Net*, the outputs of Softmax layers. And $m_{1,2}(A)$ was the jointed mass function that encoded the belief distribution of the class labels and it satisfied the constraint $m_{1,2}(\emptyset)=0$. $K = \sum_{B\cap C=\emptyset} m_1(B)m_2(C)$ indicated the level of conflict between the two mass functions, and it was used for normalizing the mass functions. Since all the transferred weights were frozen and the DST layer did not contain any parameters, this fusion model did not require training.

2.4 VALIDATION RESULTS

1) Data Set: To validate the performance and robustness of the method, two independent sets of testing data were gathered. First, a test set was collected using the same protocol described in section 2.2.B. For all testing data, only the *Remote Control* sessions were included. This data set, denoted as the Test Set, had 4,020 samples (1.1 hours) from seven terrain classes collected under lighting conditions similar to the training set. The other data set, denoted as the Dark Set, had 3,912 samples (1.09 hours) and was also collected using the same protocol, only in this case the data were recorded under natural twilight conditions.

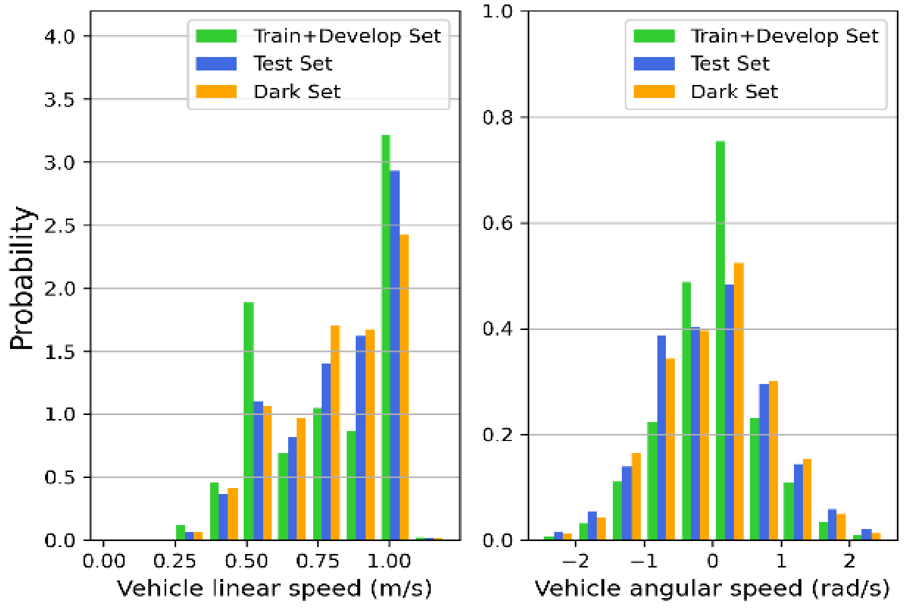


Fig. 2.7: Probability density histograms of vehicle linear and angular speed for Training / Development Set; Test Set; and Dark Set

Fig.2.7 provides a visualization of the vehicle linear and angular speeds collected for each data set. The probability distributions were similar across all data sets, except that the training and development sets contained more samples at 0.5 m/s and 1 m/s due to the addition of the *Straight Driving* sessions. This meant the data collected in the *Remote Control* sessions were sufficiently random (uniform). It can be observed that the human operator tended to have a preferred range of driving and turning speeds when requested to drive the robot with arbitrary motions.

Moreover, to test how well the models generalized to illumination conditions that were not accounted for during training, the images in the Test Set were augmented under different conditions, and two simulated data sets were created: the Sun Set and the Fog Set. A library called *albumentations* was used to generate realistic overexposed and foggy images. Samples of these test sets can be found in Fig.2.8.

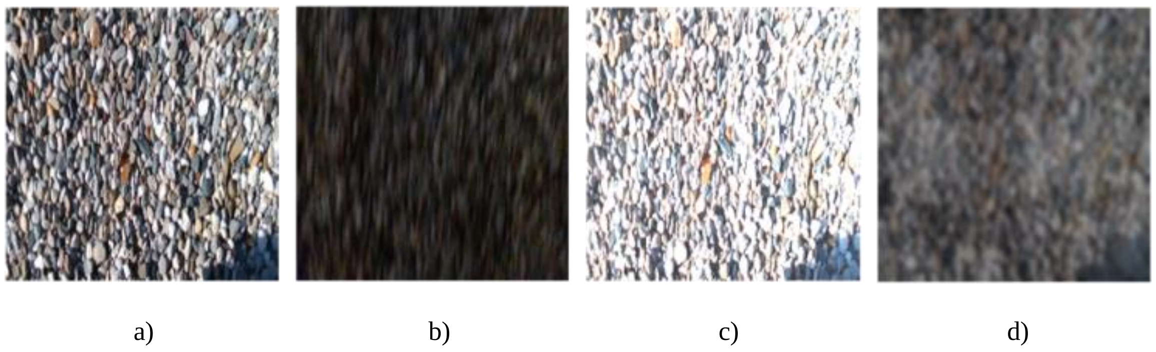

a) b) c) d)

Fig. 2.8: Sample images of gravel in a) Test Set, b) Dark Set, c) Sun Set, and d) Fog Set

2) Experiment Analysis: Table 2.3 demonstrates the confusion matrices of four models tested under the two testing sets. It was expected that the *Proprioception Net* struggled to tell the differences between asphalt (ASP), pavement (PMT), and coated floor (CFL) since they are relatively flat and solid [68]. Most of the confusion between these three classes occurred when the robot was moving straight, where the terrain characteristics of these classes were difficult to distinguish. Additionally, if the robot turned (skidded) on asphalt or pavement, the robot induced strong vibrations due to high friction. The difference between asphalt and pavement was more subtle in the eyes of the *Proprioception Net*. The darker lighting condition in the Dark Set did not hinder the accuracy of the *Proprioception Net* since it did not rely on visual information. However, the *Vision Net* failed with the Dark Set and tended to guess all input samples to be either pavement or a coated floor. This follows from the general knowledge that image CNNs are sensitive to lighting conditions. The two *Fusion Net* models had the highest overall accuracy (up to 99.38%) on the Test Set and the Dark Set as shown in Table 2.3. Leveraging two complementary modalities, the feature map-level fusion model demonstrated strong performance on the two testing sets. The decision-level fusion model was able to strategically shift its belief between the two CNN models and achieved a similar level of performance. Some samples which were indistinguishable by the

TABLE 2.3

THE CONFUSION MATRIX

Confusion Matrix for Test Set (overall: 90.18%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	640	0	0	6	0	1	9
GRS	3	374	0	10	0	55	0
GRL	0	8	500	1	4	0	0
PMT	128	12	0	433	0	25	31
SND	0	0	3	1	600	0	0
BRK	0	23	7	1	0	526	0
CFL	63	2	0	2	1	0	551

Confusion Matrix for Dark Set (overall: 92.78%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	494	5	0	6	0	32	1
GRS	0	498	7	1	0	5	0
GRL	0	0	529	0	0	0	0
PMT	97	0	0	436	0	1	35
SND	0	0	1	0	565	0	4
BRK	8	23	5	0	0	528	0
CFL	32	1	2	20	2	1	573

Confusion Matrix for Test Set (overall: 99.76%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	656	0	0	0	0	0	0
GRS	0	442	0	0	0	0	0
GRL	0	0	512	0	1	0	0
PMT	0	0	0	623	6	0	0
SND	0	0	0	0	604	0	0
BRK	0	2	0	0	0	554	1
CFL	0	0	0	0	0	0	619

Confusion Matrix for Dark Set (overall: 33.33%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	0	0	0	537	0	0	1
GRS	0	103	0	0	0	0	408
GRL	7	271	19	2	3	0	227
PMT	0	0	0	347	0	0	222
SND	0	0	0	225	1	0	344
BRK	0	0	0	1	0	541	22
CFL	0	0	0	300	0	0	331

Confusion Matrix for Test Set (overall: 99.38%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	656	0	0	0	0	0	0
GRS	0	442	0	0	0	0	0
GRL	0	0	512	1	0	0	0
PMT	23	0	0	604	0	2	0
SND	0	0	0	1	603	0	0
BRK	0	0	0	0	0	557	0
CFL	0	0	0	0	0	0	619

Confusion Matrix for Dark Set (overall: 93.81%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	**432**	**7**	**0**	**44**	**0**	**44**	**11**
GRS	**0**	**507**	**0**	**1**	**0**	**1**	**2**
GRL	**0**	**2**	**526**	**0**	**1**	**0**	**0**
PMT	**23**	**0**	**0**	**476**	**0**	**0**	**70**
SND	**0**	**0**	**0**	**0**	**567**	**0**	**3**
BRK	**2**	**0**	**0**	**0**	**0**	**562**	**0**
CFL	**5**	**0**	**0**	**23**	**3**	**1**	**599**

Confusion Matrix for Test Set (overall: 98.95%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	655	0	0	1	0	0	0
GRS	0	442	0	0	0	0	0
GRL	0	0	512	1	0	0	0
PMT	24	2	0	586	0	16	1
SND	0	0	0	0	604	0	0
BRK	0	0	0	0	0	557	0
CFL	0	0	0	1	0	0	618

Confusion Matrix for Dark Set (overall: 93.68%)

	ASP	GRS	GRL	PMT	SND	BRK	CFL
ASP	413	0	0	91	0	19	15
GRS	0	508	2	1	0	0	0
GRL	0	2	526	0	1	0	0
PMT	28	0	0	499	0	0	42
SND	0	0	0	0	562	0	8
BRK	0	0	0	0	0	564	0
CFL	2	0	0	36	0	1	592

ASP	**GRS**	**GRL**	**PMT**	**SND**	**BRK**	**CFL**
Asphlat	Grass	Gravel	Pavement	Sand	Brick	Coated Floor

Orange: the *Proprioception Net;* Blue: the *Vision Net;* Green: the *Fusion Net* at feature map-level; and Yellow: the *Fusion Net* at decision-level, tested under the Test Set and the Dark Set.

Proprioception Net and the *Vision Net* were correctly classified with both fusion models. For the Dark Set, even when almost no visual information was available for classification, both fusion models maintained a higher level of performance compared to the *Proprioception Net* and the

Vision Net. These results confirmed the earlier hypothesis that by fusing the proprioception- and vision-based models, higher levels of performance can be achieved due to their complementarity.

TABLE 2.4
THE CLASSIFICATION ACCURACY COMPARISON TABLE

	Test Set	**Dark Set**	**Sun Set**	**Fog Set**	**Avg.**
Proprioception Net	90.18%	92.78%	90.18%	90.18%	90.83%
Vision Net	**99.76%**	33.33%	88.70%	72.73%	73.63%
Feature map Fusion	99.38%	**93.81%**	95.55%	**96.33%**	96.27%
Decision Fusion	98.95%	93.68%	**96.73%**	96.22%	**96.40%**
C. Weiss et al. PSVM	74.39%	70.51%	74.39%	74.39%	73.42%
C. Weiss et al. VSVM	62.13%	22.66%	24.23%	37.17%	36.55%
C. Weiss et al. FSVM	79.80%	52.11%	52.59%	64.72%	62.31%
G. Reina et al. PSVM	58.52%	55.52%	58.52%	58.52%	57.77%
G. Reina et al. VSVM	84.52%	17.40%	41.46%	60.81%	51.05%
G. Reina et al. FSVM	90.25%	26.52%	48.18%	72.22%	59.29%

Blue: models from this study; Orange: C. Weiss et al.; Green: G. Reina et al.; "PSVM", "VSVM", and "FSVM" stand for proprioception SVM, vision SVM, and fusion SVM respectively. The term accuracy refers to weighted overall accuracy, and the **bold** numbers are the highest accuracy in each column.

Table 2.4 provides an accuracy comparison between the models developed in this study as well as the models from [78] and [79]. The methods in [78] and [79] were recreated with the best effort and trained using the same training set. The test results of [78] and [79] are presented as baselines. As the Sun Set and the Fog Set were synthesized using the Test Set, the test accuracies for all the proprioceptive-based models were the same as the Test Set.

Table 2.4 demonstrates that the fusion models developed in this study have consistently high performance (over 93%) even in illumination conditions that were not accounted for during training, namely the Sun Set and the Fog Set. It can also be observed that even though [78] and [79] did improve the classification accuracy in the Test Set by fusing the SVM models, this trend did not generalize well to other test sets. In some cases, the accuracy of the proprioception SVM was higher than the fusion SVM, which implies the fusion mechanics used in [78] and [79] could not strategically shift their beliefs between the proprioception and the vision models according to the illumination conditions.

TABLE 2.5
THE TIME PROFILING AND STORAGE COMPARISON TABLE

	Inference Time	**Storage Size**
Proprioception Net	~1 ms	0.8 Mb
Vision Net	6.83 ms	12.1 Mb
Feature map Fusion	6.84 ms	12.8 Mb
Decision Fusion	6.95 ms	12.4 Mb

Tests were performed using an AMD Ryzen7 4800H Processor (2.9 GHz), no GPU was used.

Table 2.5 provides information on the inference time and storage size of the models in this study. The *Proprioception Net* was very fast and compact, while the other models required over 6 ms to process a one-second data pair. As the collection of proprioceptive feedback required a whole second, comparatively, an inference time of millisecond-level can satisfy real-time operation requirements. Moreover, storage sizes for all of the models can easily fit in the memory of a single-board computer like a Raspberry Pi or a NVidia TX2.

2.5 ABLATION STUDY

An ablation study was conducted to help understand the efficacy of the use of the derived features mentioned in Section 2.2.D: Instead of using the features adopted from [79], a Proprioceptive CNN model, denoted as a *Raw Proprioception Net*, that takes the raw proprioceptive sensor signals as inputs was built and compared to the *Proprioception Net*. To ensure a fair comparison, both networks were trained under the same conditions. Two pairs of training and development sets were generated using the same random seed such that they had the same data frames as their counterparts. One pair of the training and development sets used in the ablation study was formed following the same procedures described in Section 2.2.C, while the other excluded the feature derivation procedure and packed the proprioceptive signals untreated (i.e., $\omega_{enc,l}$, $\omega_{enc,r}$, $accel_x$, $accel_y$, $accel_z$, $v_{vio,x}$, $w_{vio,z}$, I_l, I_r, ϕ, θ; where I_l and I_r were the current feedback from the left and right DC motors). Note that in (2.1, 2.2, 2.3), current feedback I_l and I_r, and vehicle attitude ϕ and θ were used to compute the motion resistance coefficients $f_{r,l}$ and $f_{r,r}$. Therefore, by omitting these computations, the input space of the Proprioceptive CNN increased from 100 × 9 to 100 × 11. The network structure of the *Raw Proprioception Net* was the same as the *Proprioception Net* except for a larger input layer to accommodate for the change in input space. As a result, the total number of trainable parameters was marginally larger (35,527 parameters in total).

TABLE 2.6
THE CLASSIFICATION ACCURACY COMPARISON TABLE

	Develop Set	**Test Set**	**Dark Set**
Proprioception Net	93.21%	89.54%	92.90%
Raw Proprioception Net	91.49%	88.86%	92.54%

Table 2.6 shows the accuracy comparison between the *Raw Proprioception Net* and the *Proprioception Net*. As mentioned earlier, the training and development sets used in the ablation study were newly generated. The data composition was slightly different from before due to the random selection procedure described in Section 2.2.B, which caused accuracy fluctuations for the *Proprioception Net* (less than ±0.64% compared to Table 2.4) in both the Test Set and the Dark Set.

As shown in Table 2.6, the performance margin of using the proposed derived features was small -- about 1.72% in the development set. The accuracy of the *Raw Proprioception Net* in the Test Set and Dark Set also suggested the 1D CNN structure given in Table 2.1 can be trained to process raw proprioceptive signals directly and achieve a similar level of accuracy as the one using the proposed derived proprioceptive features. This ablation study implied that the proposed data processing pipeline can be further simplified in a follow-up study.

2.6 Conclusion and Future Work

This study successfully developed a fast, lightweight, and motion-robust proprioception-based terrain classification method using a CNN model and signals from common on-board UGV sensors. The strong performance (over 89.54%) and the robustness of this method were demonstrated by testing it under data sets that contained arbitrary vehicle motions. Furthermore, it was shown that the proprioception and vision-based models were complementary. By fusing the two models, a higher level of accuracy (up to 99.38%) was observed in both the feature map-level and decision-level fusion models. Four distinct lighting conditions were used to validate the generalizability of the fusion models. The validation results (over 93.68% accuracy) showed the fusion models in this study can strategically cope with different environmental illumination without human interference and achieve significantly higher accuracy than the baseline methods. The decision-level fusion model achieved the highest average accuracy (96.40%) over the four test sets. Compared to the feature map-level fusion, the decision-level fusion was more stable under different test conditions and did not require further training. It is worth noting that the *Proprioception Net* and *Vision Net* were independently trained before their integration into the decision-level fusion model as frozen layers. This suggested the association between the proprioceptive signals and the image was not critical to the success of the decision-level fusion in this study. Additionally, the time profiling indicated the online deployment of the models in this study was possible. Lastly, an ablation study showed the proposed data processing pipeline can be further simplified by removing the use of all manually derived features. Doing so permitted the proposed method to be less model-dependent, as the knowledge of the vehicle specifications, dynamic, and kinematic models were no longer required.

In the future, more human operators should be recruited to enrich the variety of driving motion in the data sets, or a program can be developed to automate the data collection process. Data augmentation on proprioceptive signals should be performed to further improve accuracy. A larger range of vehicle motions and data collection sites should be included in the test sets to further validate the generalizability of the models. Feature importance analysis would need to be conducted to reduce the number of required signals and the size of the model. More accurate data association techniques like SLAM should be applied for online deployment. And other fusion mechanics should be investigated so that the correlations between proprioceptive signals and images can be effectively utilized. Moreover, as this method is vehicle-specific, the portability and scalability of the method should be addressed in future studies.

CHAPTER 3

DEVELOPMENT OF A NEURAL NETWORK AUGMENTED RGB-DEPTH PERCEPTION SYSTEM FOR INDOOR ROBOT NAVIGATION

3.1 ABSTRACT

Determining the drivable area, or free space segmentation, is critical for mobile robots to navigate indoor environments safely. However, the lack of coherent markings and structures (e.g., lanes, curbs, etc.) in indoor spaces places the burden of traversability estimation heavily on the mobile robot. This study explored the use of a self-supervised one-shot texture segmentation framework and an RGB-D camera to achieve robust drivable area segmentation. With a fast inference speed and compact size, the developed model, MOSTS (Miniature One-Shot Texture Segmentation), is ideal for real-time robot navigation and various embedded applications. A benchmark study was conducted to compare MOSTS's performance with existing one-shot texture segmentation models. Additionally, a validation dataset was built to assess MOSTS's ability to perform texture segmentation in the wild, where it effectively identified small low-lying objects that were previously undetectable by depth measurements. Further, the study also compared MOSTS's performance with two State-Of-The-Art (SOTA) indoor semantic segmentation models, both quantitatively and qualitatively. The results demonstrated that MOSTS offered comparable

accuracy (85.7% mIoU) with up to eight times faster inference speed (93.13 FPS) in indoor drivable area segmentation.

3.2 INTRODUCTION

Robot perception is one of the essential prerequisites for mobile robot navigation. Depending on the complexity of the robot's locomotion mechanism and the expected operating environment, mobile robots are designed to have specialized perception capabilities to navigate and interact with the environment appropriately and safely. Specifically, indoor mobile robots often need to operate in cluttered and confined spaces initially designed for human access, which raises some unique challenges for their perception systems. These robots (e.g., smart wheelchairs, service robots, etc.) usually have relatively small form factors such that they can access narrow doorways or under-counter spaces. This characteristic restricts the battery, computation power, and sensor systems that can be carried on the robot. Thus, size, weight, power, and cost (a.k.a. "SWaP-C") are important constraints to consider. Likewise, indoor mobile robots generally have low ground clearance and small actuation wheels; they are more susceptible to jamming and tipping caused by small obstacles, which are usually ignored and driven over by larger outdoor robots. Hence, the perception systems for indoor robots need to detect and identify small obstacles. Complex floor

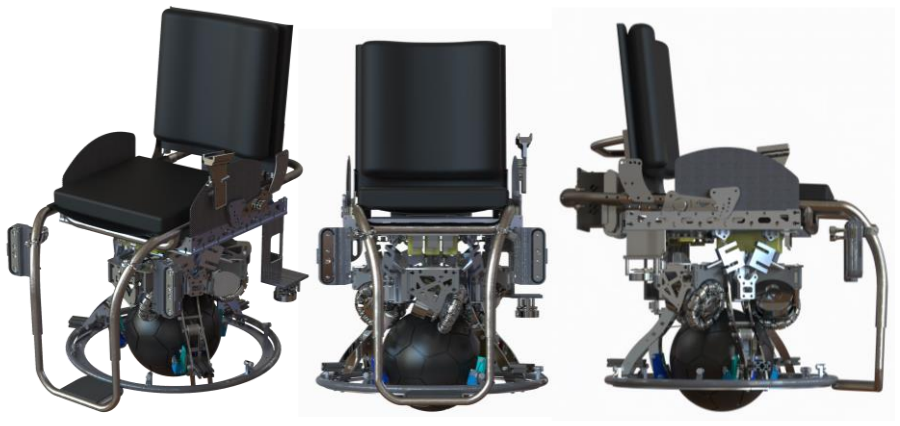


Fig. 3.1. The assistive rideable ballbot system, PURE

plan layouts and scattered interior objects can create a large number of occlusions, reflections, and confusing situations that hinder the performance of robot sensors and limit their FOVs (Field of View). As a result, the robot's perception system must be fast and responsive enough to react to sudden obstacles and facilitate avoidance maneuvers.

Our research group is developing a personal mobility device named PURE (Personalized Unique Rolling Experience, Fig.3.1). PURE is an assistive rideable ballbot system, characterized by low ground clearance and driven by a ball. Besides facing the challenges common to traditional ground robots, PURE also contends with under-actuated dynamics, which induces tilting motions during acceleration [92]. The commonly held assumption that the transformation between the ground plane and the robot's sensor coordinates is (entirely or partly) constant and known [74], [93] does not apply to dynamically balanced devices like PURE, as these robots are not constrained to strict planar motion due to the ability to tilt when accelerating or decelerating.

Inspired by the need to safeguard PURE, this study proposed an efficient indoor Drivable Area Segmentation (DAS) framework that is robust to tilting motion and capable of detecting small unknown obstacles in real-time on a low-power Single Board Computer (SBC). The proposed method approached the problem of DAS in two key steps. First, a real-time and motion-robust ground plane segmentation node segmented the point cloud collected by an RGB-D camera and made an initial estimation of the drivable area. Second, a one-shot segmentation model refined the drivable area by filtering out small obstacles and anomalies in the RGB image. The outputs of the proposed method were a simplistic 2D obstacle map that encoded all the large obstacles (e.g., walls, furniture, pedestrians, etc.) and a traversability map that indicated the surrounding regions that were free of small obstacles or anomalies. With a reference texture of the traversable surface, this approach sought to identify all potential obstacles threatening the robot system, including small, unknown objects that may not be detectable using point clouds or depth images.

Recently, there has been a growing trend of training large Vision Transformer (ViT) models, such as the Segment Anything Model (SAM) [94], on unprecedentedly large datasets to achieve unparalleled accuracy and generalizability. However, these models often require extensive computational and data resources, making them impractical for real-time deployment on low-power and embedded systems. Additionally, the resource requirements for training these models are often beyond the reach of many researchers and individuals. In contrast to this approach, this study focused on developing a lightweight and versatile model that can be trained and deployed with significantly fewer resources. The training of the proposed method did not rely on any dense (pixel-wise annotations) manually labeled dataset as it followed a self-supervised methodology.

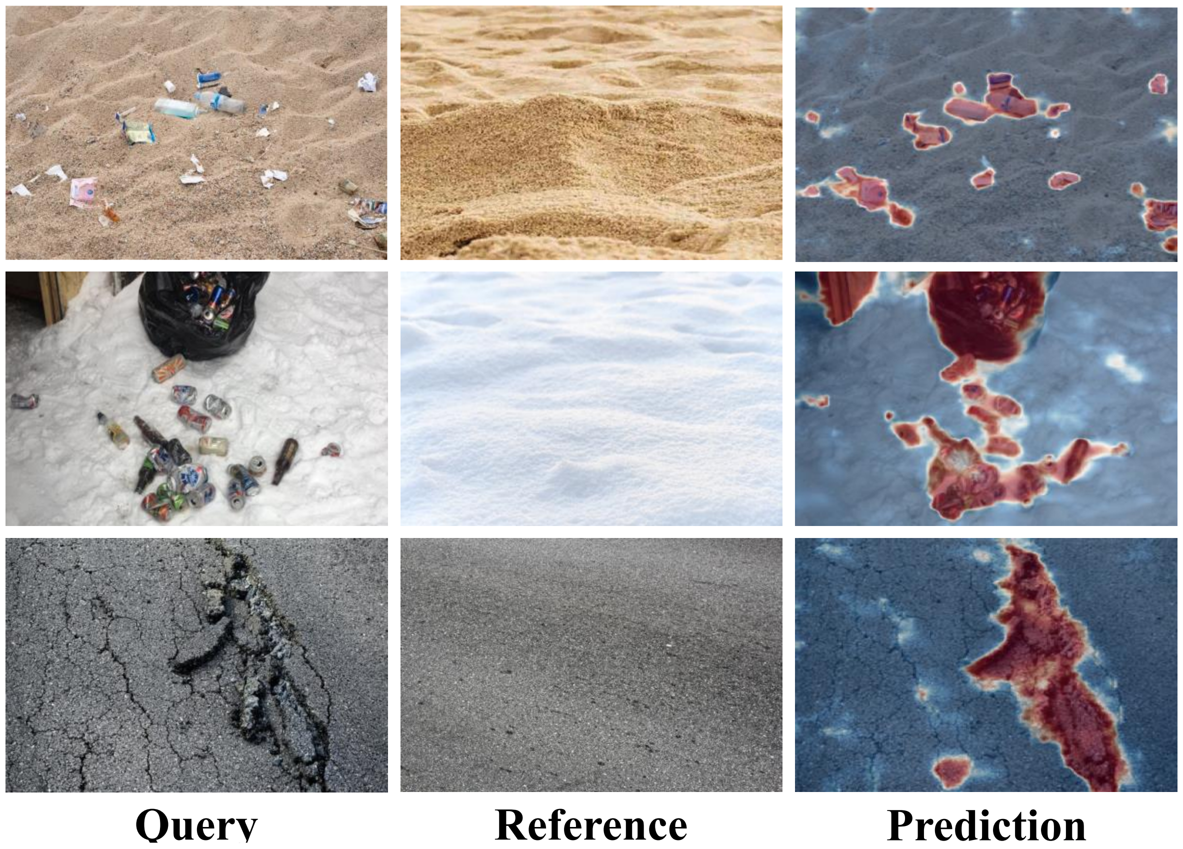


Fig. 3.2. One-shot texture segmentation in the wild using MOSTS. Query and reference images were collected from the internet.

The short inference time enabled the proposed method's deployment on an SBC (93.13 FPS for the neural network model alone and about 30 FPS for the entire framework due to the limited sampling rate of the camera). By using a one-shot segmentation formulation, this approach allows the user to specify traversable regions since not all floor areas are suitable for indoor robots. Furthermore, the proposed framework can be re-configured to adapt to a different working environment or robot mission by changing the reference image during runtime (Fig.3.2).

The key contributions of this study included the following:

1. A novel Miniature One-Shot Texture Segmentation (MOSTS) model optimized for an embedded, real-time application.
2. A novel Perlin-Noise-based [95] collage generation technique for training robust texture segmentation models.
3. An Indoor Small Objects Dataset (ISOD) that contained diverse and densely labeled images for validating the performance of the proposed method in the wild.

The code and dataset used in this study are publicly available at:

https://github.com/mszuyx/MOSTS

The rest of this chapter is organized as follows: Section 3.2. is a brief literature review of related studies. Section 3.3. offers an overview of the proposed framework. Section 3.4. and Section 3.5. provide the implementation details of the ground plane segmentation algorithm and the one-shot DAS model. Section 3.6. explains the experimental setup. Section 3.7. presents and discusses the validation results, followed by conclusions and suggestions for future work in Section 3.8.

3.3 RELATED WORK

A. Indoor Vision-based Drivable Area Segmentation

Earlier attempts [95], [96] used vision-based methods that segmented the obstacle-free floor from monocular images based on the assumption that the floor has a uniform texture (e.g., [96], [97]). More recent work [98]–[100] approached this problem using supervised Deep Neural Networks (DNNs) and achieved promising results. The training of these DNNs required a large amount of densely labeled data. While outdoor datasets can be effectively collected by driving a sensor-mounted vehicle on open roads, the collection of indoor datasets is expensive and difficult due to manual labor cost, lack of building access, and privacy concerns.

Furthermore, Kumar et al. [101] pointed out that the difficulty of indoor drivable area segmentation went beyond the lack of coherent vision marks. They found that existing homography-based floor segmentation methods often fail to capture low-lying small obstacles (typically $\leq 3\ cm$ height) in RGB images. The authors further stated that it would also be difficult for depth sensors such as laser range finders and stereo cameras to detect low-lying obstacles based on noisy depth values. To combat this issue, they developed a homography-based method that involved the use of superpixelling [102], Markov Random Field (MRF) [103], and Graph Cut [104]. Following that study, the authors of [105]–[109] also acknowledged the importance of detecting small and unexpected obstacles (or anomalies) in drivable areas. These studies tackled the problem by creating or using datasets that contained small obstacles for the training of their DNNs.

B. Point Cloud Ground Plane Segmentation

A point cloud is a standard data format representing the robot's surroundings in 3D space. However, raw 3D point clouds with no appearance information of the detected subjects can be challenging to interpret. It treats the surrounding objects and the traversable surfaces equally while they hold different values for navigation purposes. Therefore, in ground vehicle applications, ground plane segmentation of 3D point clouds is a common and essential step to filter the points with less interest [110]–[116]. Himmelsbach et al. [110] mapped all the 3D points into cylindrical sub-segments and simplified the ground plane segmentation task to a 2D line fitting problem. A gaussian-process-based iterative scheme, GP-INSAC, was introduced to separate points into ground and non-ground groups [111]. The work in [112] showed that 3D points can be effectively segmented and classified in normal space. Similarly, Wang et al. [113] estimated the plane segments by calculating the normal vector of each voxel grid. Byun et al. [117] followed a probabilistic approach and utilized MRF. More recent studies like [114], [115], [118] used RANdom SAmple Consensus (RANSAC) based methods and achieved tremendous success. Study like [24] also approached this problem using a DNN. Most studies showed they could achieve real-time performance and high accuracy on datasets containing large roadway objects (e.g., vehicles, walls, pedestrians, etc.). However, as most of the point cloud data used in these studies were collected with statically stable platforms (vehicles), it is unclear whether these methods would work effectively under the influence of rapid and large tilting motions.

In the context of this study, ground plane segmentation was used as a pre-processing step to obtain a preliminary estimation of the drivable areas.

C. Efficient RGB-D Semantic Segmentation

RGB-D semantic segmentation models like [119]–[125] are extremely powerful and versatile computer vision tools. As compelling as they might be, most were not designed to be used on robots or embedded systems. This drawback led to the development of efficient RGB-D semantic segmentation models like the RedNet [126] and ESANet [127], which aim to achieve adequate accuracy with optimized computation requirements. Although state-of-the-art models like [126], [127] have shown the promising potential of solving the problem of drivable area segmentation with their high accuracies and real-time performance, these models can be unpredictable (i.e., neglecting or misclassifying) while coping with unknown objects. The proposed method handled the task of drivable area segmentation based on comparing texture differences. It did not rely on semantic or contextual information from the scenes, and therefore, was predicted to be more generalizable to novel and unknown environments and objects.

To rigorously validate the efficacy of the proposed method, this study collected and labeled Indoor Small Objects Dataset (ISOD), an indoor RGB-D dataset with numerous small low-lying objects. The dataset also contained scenes with tilted views and motion blur to simulate the motion effects of dynamically balanced robots. This study trained and converted [126] and [127] into drivable area segmentation models using the commonly used indoor RGB-D datasets SUNRGB-D [128]. The converted models were evaluated with ISOD, and their validation results will be presented in this study as performance baselines and compared with the proposed method.

D. One-Shot Texture Segmentation

Textures are effective visual cues humans use to perform daily low-level visual inference and scene understanding [129], [130]. The use of texture representations is also very common in applications like terrain recognition [131] and surface defect (or anomaly) detection [132]. While the studies on texture classification and segmentation are extensive, one-shot texture segmentation (or retrieval) is a relatively recent field of research. Recent studies in [129], [130] leveraged a Voronoi-based collage generation technique to automatically generate pixel-wise annotations to train their one-shot texture segmentation models. The trained models identified the region containing the targeted texture when given a reference image.

The OSTR model in [130] was intended for texture retrieval in fashion and e-commerce, this study explored using one-shot texture segmentation for indoor real-time robot navigation. This study conducted a benchmark test and demonstrated that the developed model, MOSTS, outperformed the SOTA models in accuracy and speed. Additionally, this study proposes a novel Perlin-Noise-based collage generation method that significantly improved the robustness of MOSTS.

3.4 FRAMEWORK OVERVIEW

Fig.3.3 provides the overall framework of the proposed method and visualization for each data stream. The RGB-D camera (Intel RealSense D455) offers four types of data: RGB image, depth image and point cloud that represent the 3D structures of the captured scene, and Inertial Measurement Unit (IMU) measurements that represent the orientation of the camera. The proposed framework fed the point cloud and IMU data to a ground plane segmentation node that separated the raw point cloud into a ground plane group and a non-ground group. The segmented point clouds were used to construct a 3D Octomap (i.e., a grid map made by 3D voxels) [133] that described the 3D scene. Then, the 3D voxels were projected to the ground to generate a more simplistic 2D occupancy grid for subsequent high-level control tasks like path planning and navigation. The

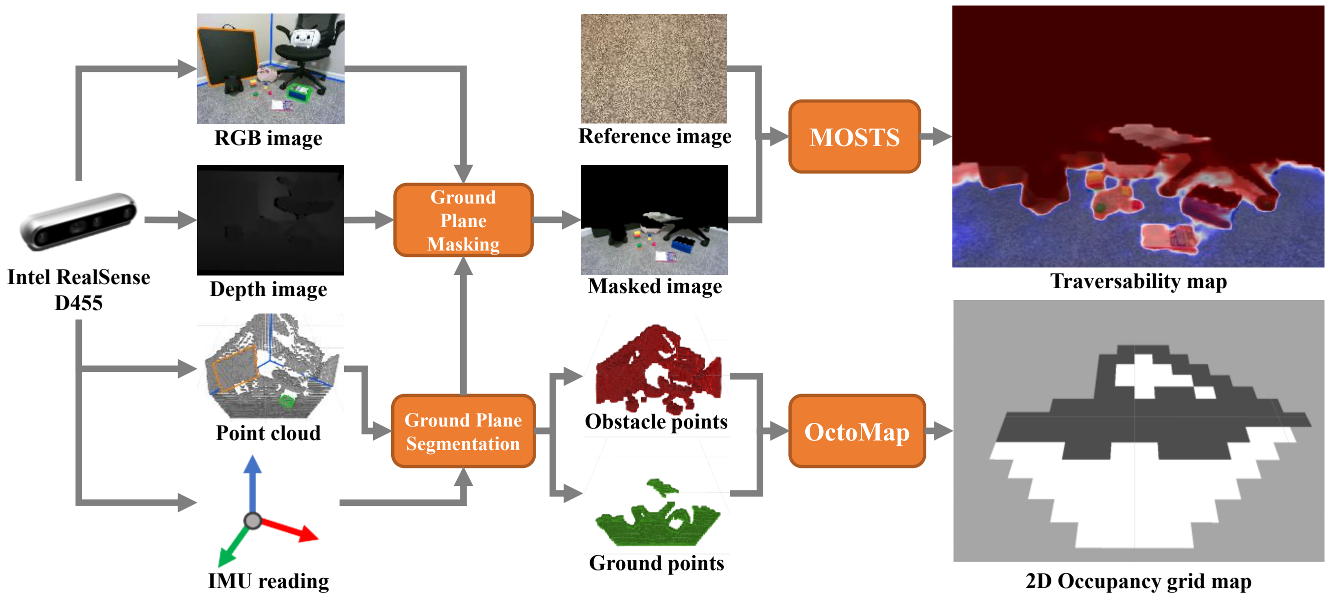


Fig. 3.3. The overall framework of the proposed method.

ground plane model and the depth image were used to mask the raw RGB image, such that most of the obstacle pixels were masked out (painted black). This pre-processing step prepared the query image for MOSTS, which further segmented the ground regions into drivable areas and anomalies based on a target terrain surface given by a reference image. The outcome of MOSTS can be re-projected to the robot workspace coordinate to form a traversability map.

3.5 Real-Time Motion-Robust Obstacle Grid Map

A. Problem Formulation

The ground plane segmentation of 3D point clouds is a common and essential step to filter the points with less interest [114], [115]. In the context of this study, ground plane segmentation was used as a pre-processing step to obtain a preliminary estimation of the drivable areas. As the proposed framework is expected to operate primarily indoors, this study made the approximation that the geometry of an indoor ground plane can be roughly modeled using a point-normal equation:

$$ax + by + cz + d = 0 \tag{3.1}$$

where $[a, b, c]^T$ are constant values that form the normal vector $\vec{\mathbf{n}}$ of the ground plane, and d can be seen as the orthogonal offset distance between the ground plane and the point cloud's frame. The position vector of the i^{th} point that belongs to the ground plane can be written as $\mathbf{P}_{ground,i} = [x, y, z]^T$. (3.1) can be rewritten into matrix form:

$$\vec{\mathbf{n}}^{\mathrm{T}}\mathbf{P}_{ground,i} = -d \tag{3.2}$$

The objective of the ground plane segmentation node is to estimate the values of the normal vector $\vec{\mathbf{n}}$ and offset distance d such that they can be used to determine whether a given point belongs to the ground plane $\mathbf{P}_{ground}$ or not.

B. Filtering

The proposed ground plane segmentation algorithm was primarily inspired by a real-time RANSAC-based ground plane segmentation method [115]. For RANSAC-based methods, the most crucial part is to select a reasonably good initial guess (candidate points) for the targeted plane. This can reduce the need for more RANSAC iterations, and the algorithm is more likely to converge to the true plane. To reliably obtain these candidate points, the proposed method applied a series of filters and operations to the input point cloud (Fig.3.4c-e).

1) Orientation Alignment

First, the synchronized IMU data from the camera were utilized to align the point cloud orientation with the gravity vector to cancel the robot's tilting motion. The raw IMU data consisted of angular rates $\mathbf{\Omega} = [\dot{\omega}_x, \dot{\omega}_y, \dot{\omega}_z]$ measured by the gyroscope, and the linear accelerations $\mathbf{A} = [\alpha_x, \alpha_y, \alpha_z]$ measured by the accelerometer in three Euclidean axes. The Euler angles

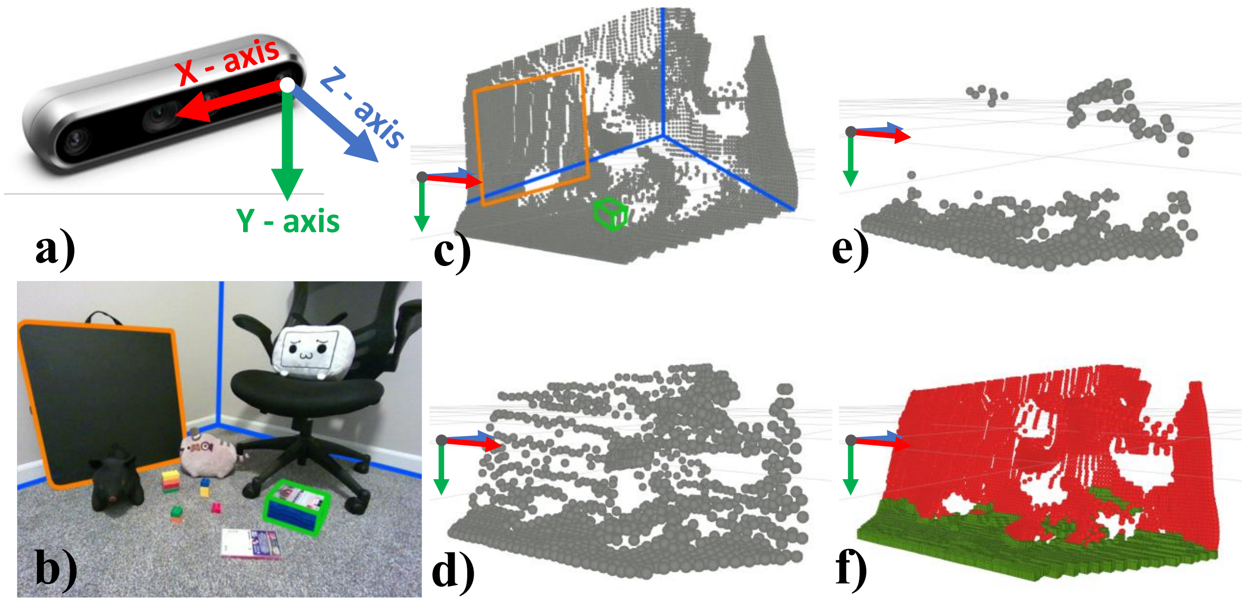


Fig. 3.4. a) The point cloud frame definition; b) the RGB image of the scene; c) the raw input point cloud; d) the point cloud after IMU alignment and the voxel grid filter; e) the point cloud after the radius outlier removal filter; f) the final output, **green** indicates ground points, and **red** indicates obstacle/non-ground points. The corner of the shown scene and some objects are highlighted in b) and c) for easier visual understanding. The coordinate frame in c)-f) indicates the camera position and orientation.

$(yaw: \theta_y, pitch: \theta_x, roll: \theta_z)$ that described the camera orientation were calculated using an advanced complementary filter [134] that fused $\mathbf{\Omega}$ and $\mathbf{A}$. The pitch $\theta_x$and roll θ_z angles were used to construct a rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$. The yaw angle θ_y was set to zero as this operation has no interest in changing the yaw rotation. **R** follows a Y-X-Z convention such that gimbal lock only happens if the camera points straight up or down (which rarely happens for a forward-facing camera). Assuming that $\mathbf{P}_{input} \in \mathbb{R}^{3\times n}$denotes the 3D positions of all the raw input points, this operation was simply:

$$\mathbf{P}_{aligned} = \mathbf{R}\,\mathbf{P}_{input} \tag{3.3}$$

After the alignment, the resulting ground plane in $\mathbf{P}_{aligned}$ appeared mostly upright in the camera frame (Fig.3.4d). There are two main benefits of doing an orientation alignment. 1) The alignment can simplify the problem to mimic the case where the robot always stays upright. And as the ground (floor) is orthogonal to the gravity vector in most indoor spaces, RANSAC only needs to determine one parameter, the offset distance d in this simplified problem. 2) The alignment allows the downstream mapping algorithms like Simultaneous Localization and Mapping (SLAM) to consistently map the environment in a unified planar frame.

2) Box Crop

Second, a box crop operation was applied to retain the points within a 7 m planar range to the camera frame, where the measurements were most reliable. The points 1 m above the camera were also removed to discard ceiling points, which served little value in ground robot navigation. The resulting point is denoted as $\mathbf{P}_{cropped}$.

3) Voxel Grid Filter

Third, a voxel grid filter downsampled the raw point cloud data by taking a spatial approximation of the points in each voxel (i.e., a 3D cube with a defined size). This filter generated voxel indexes $\boldsymbol{V}$ over the cropped point cloud $\mathbf{P}_{cropped}$:

$$\boldsymbol{V}_i = \begin{bmatrix} Round\left(\frac{\mathrm{p}_{i,x} - x_{min}}{\sigma_x}\right) \\ Round\left(\frac{\mathrm{p}_{i,y} - y_{min}}{\sigma_y}\right) \\ Round\left(\frac{\mathrm{p}_{i,z} - z_{min}}{\sigma_z}\right) \end{bmatrix}, \ \mathbf{p}_i \in \mathbf{P}_{cropped} \tag{3.4}$$

where x_{min}, y_{min}, and z_{min} are the minimum point position values of $\mathbf{P}_{cropped}$. In the proposed algorithm, the size of the voxel σ was empirically defined to be *0.03m* × *0.3m* × *0.03m* (X-Y-Z convention) such that the output point cloud had a lower resolution on the vertical axis (i.e., y-axis, the coordinate frame is defined in Fig.3.4a). Points were then sorted based on their voxel index. The set of points within the same voxel shared the same voxel index and were replaced by their centroid. This filter can significantly reduce the number of points for subsequent processing without losing vital features (Fig.3.4c).

4) Pass Through Filter

For the purpose of selecting ground plane candidate points, any point above the camera height should be automatically disqualified. Therefore, a pass-through filter was applied to remove points having negative y axis readings (according to the camera frame definition, Fig.3.4a).

5) Radius Outlier Removal Filter

Last, a radius outlier removal filter was used. This filter used a Kd-Tree radius search to determine the number of neighbors k that lied within a radius r of a query point. If k was less than a threshold k_{min}, then this query point was removed from the point cloud. As the filtered point cloud had a lower resolution on the vertical axis (y-axis), points in vertical structures (e.g., walls, large furniture) were sparse and had fewer neighboring points. Hence, they were more likely to be filtered out. Most remaining points were likely to belong to the largest plane orthogonal to the gravity vector (Fig.3.4e) and made good RANSAC candidates.

C. Ground Plane Segmentation

Once the candidate points were determined, they were sorted based on their heights, and the lowest 50% of points were sampled to form the initial seed for the RANSAC algorithm. (Appendix A: Alg.3.1). Note that due to the frame definition of the point cloud (Fig.3.4a), having large y-values meant they were low-lying points. The RANSAC algorithm then iteratively estimated the ground plane model using the candidate points. At the final iteration, the algorithm used the estimated $\vec{\mathbf{n}}$ and d to infer $\mathbf{P}_{cropped}$ and segmented it into two groups: the obstacle group (red) and the ground plane group (green) (Fig.3.4f). Due to the imperfect in the point cloud measurement, the ground points might not form a clear-cut surface. Therefore, a set of threshold values T_{up} and T_{low} were needed to determine the ground plane's upper and lower padding thickness (tolerance). Any point within this tolerance range was considered a ground point; others were obstacle points.

There were two settings for the function $\mathbf{estimatePlane}(*)$ (see APPENDIX A). Users could either choose to estimate the value of *d* alone or estimate the value of $\vec{\mathbf{n}}$ as well. As mentioned earlier, the orientation alignment could keep the ground plane mostly upright, so the normal vector estimation was unnecessary. However, intense accelerations and vibrations might compromise the IMU's ability to estimate orientation accurately and cause the aligned ground plane to be slightly tilted. In this case, a Principal Component Analysis (PCA) was used to estimate the normal vector of the aligned ground plane and refined the result. First, a covariance matrix $\mathbf{C} \in \mathbb{R}^{3\times 3}$ needed to be computed from the candidate points $\mathbf{P}_c \in \mathbb{R}^{3\times m}$:

$$\mathbf{C} = (\mathbf{P}_c - \overline{\mathbf{P}_c})(\mathbf{P}_c - \overline{\mathbf{P}_c})^{\mathrm{T}}$$

$$\overline{\mathbf{P}_c} = \begin{bmatrix} \bar{p}_x \\ \bar{p}_y \\ \bar{p}_z \end{bmatrix} [1 \quad \cdots \quad 1]_m \tag{3.5}$$

where $\overline{\mathbf{P}_c} \in \mathbb{R}^{3\times m}$ consists of the mean position of the candidate point cloud along the row direction. As explained in [115], the covariance matrix $\mathbf{C} \in \mathbb{R}^{3\times 3}$ represents the dispersion of the candidate points, and the direction that accounts for the least variance is likely to be orthogonal to the ground plane (a flat surface) and parallel to the normal vector. Singular Value Decomposition (SVD) was used to identify this direction by finding the singular vector corresponding to the smallest singular value. In general, the segmentation algorithm required 7 to 13ms computation time on the single board computer (depending on the size of the point cloud, the results might vary). If users choose to skip the SVD calculation to preserve computation power,

$\mathbf{estimatePlane}(*)$ will return $[0,1,0]$ as the default value of $\vec{\mathbf{n}}$. Note that, in both cases, $\vec{\mathbf{n}}$ only reflects the normal vector of the ground plane after orientation alignment. Therefore, an inverse operation (Appendix A: Alg.3.1 step 35) was needed to obtain the actual normal vector $\vec{\mathbf{n}}_{corrected}$ for subsequent algorithms.

D. Ground Plane Masking

The $\vec{\mathbf{n}}_{corrected}$ and d were sent to the ground plane masking node, where the depth image was used to mask the RGB image. First, a homography transformation was applied to wrap the depth image to align with the RGB image. This step was necessary as the depth sensor and the RGB sensor had different FOVs and coordinate frames. Second, the $\vec{\mathbf{n}}_{corrected}$ and d were used to screen the pixel values in the depth image to build a mask that blocked all the pixels belonging to the obstacle group. Finally, this mask was applied to the RGB image and formed the masked ground image, which served as one of the inputs of MOSTS. There are two benefits of using the masked image instead of the raw RGB image: 1) Vision-based DNNs are often influenced by contextual correlations [135]. Therefore, blocking out non-task-related background pixels can mitigate their effect and guide the model's attention to the subject of interest (the ground areas). 2) MOSTS was designed to segment the regions in the query image that share a similar texture to the reference image and does so without any scene understanding. When the surrounding walls or furniture have a similar texture as the ground/floor, MOSTS might mistake those regions as drivable areas. However, using the masked ground image can prevent these edge cases.

3.6 TEXTURE ORIENTED, SELF-SUPERVISED ONE-SHOT DRIVABLE AREA SEGMENTATION

A. Problem Formulation

Similar to other one-shot segmentation models, this model takes two inputs, i.e., the query image Q, which contains regions of different textures, and the reference image R, which embodies the target texture class. The output of MOSTS is a probability map that indicates regions of the query image that correspond to the target texture class. To train such a model, a triple T^{i} of training samples was required:

$$\mathrm{T}^{i} = (\mathrm{Q}, \mathrm{R}^{i}, \mathrm{G}^{i}) \tag{3.6}$$

where Q is the query image that contains at less one texture class, $\mathrm{G}^{i}$is the pixel-wise ground truth label corresponding to the i^{th} class texture in Q, and R^{i} is the reference image of the i^{th} class. The neural network model $f(*)$ can be presented in the following form:

$$\mathrm{P}^{i} = f(\mathrm{Q}, \mathrm{R}^{i}, \theta) \tag{3.7}$$

where P^{i} is the predicted segmentation result and θ is the set of learnable parameters in function $f(*)$. The goal of the training process is to learn the optimized θ^{*} given the training triple T^{i} such that the loss function is minimized:

$$\theta^{*} = \underset{\theta}{\operatorname{argmin}}\, Loss(\mathrm{P}^{i}, \mathrm{G}^{i}) \tag{3.8}$$

For the loss function, this study experimented with Focal Loss [136], Focal Tversky Loss [137], Balanced-Binary Cross Entropy Loss (B-BCE) [138], Dice Loss [139], Batch-Dice Loss [140], and Combo Loss [141]. The Combo Loss provided the best training performance, and this finding agreed with the conclusion in [130]. The Combo Loss is essentially a weighted sum of the B-BCE Loss and the Dice Loss:

$$L_{combo} = \lambda L_{B-BCE} + L_{Dice} \tag{3.9}$$

where λ is the weighting factor in favor of B-BCE Loss, in this study, the value of λ was set to 10. The B-BCE Loss can be written as follows:

$$L_{B-BCE}(\widehat{\mathrm{Y}}, \mathrm{Y}) = -\beta \mathrm{Y} \log(\widehat{\mathrm{Y}}) - (1-\beta)(1-\mathrm{Y}) \log(1-\widehat{\mathrm{Y}}) \tag{3.10}$$

where $\widehat{\mathrm{Y}}$ denotes the prediction made by the neural network, Y is the ground truth label, and $\beta = 1 - \frac{\sum_{y^+ \in \mathrm{Y}} y^+}{|\mathrm{Y}|}$ is the weighting factor that balances the contribution of the positive and negative pixel labels. y^+ denotes the pixel that contains a positive label. Similarly, Dice Loss can be written as:

$$L_{Dice}(\widehat{\mathrm{Y}}, \mathrm{Y}) = 1 - \frac{2|\mathrm{Y}\widehat{\mathrm{Y}}| + \epsilon}{|\mathrm{Y}| + |\widehat{\mathrm{Y}}| + \epsilon} \tag{3.11}$$

where ϵ is the smoothing factor that prevents the denominator from ever becoming zero when $|\mathrm{Y}| = |\widehat{\mathrm{Y}}| = 0$. While the Dice Loss is non-convex in nature, the combination of these two loss functions is smooth and able to handle class imbalance [142].

B. Perlin-Noise-based collage generation and Model Training

The studies in [129] and [130] used a benchmark texture database called the Describable Textures Dataset (DTD) [143] to train their models. DTD is a dataset that contains 5640 images organized into 47 texture categories inspired by human perception. A random Voronoi-based collage generation technique was used in both [129] and [130] to form synthetic query images (Fig.3.5a). However, query images generated in this manner generally have three biases: 1) there is no discontinuous region of the same textures; 2) there are only straight edges separating different textures; 3) there is no small region in the collage, and regional areas are relatively balanced. As a result, the model trained using these synthetic query images likely lacks the ability to deal with arbitrary class boundaries in the real world. Therefore, this study designed a novel Perlin-Noise-based collage generator to synthesize pseudo-random query images (Fig.3.5b).

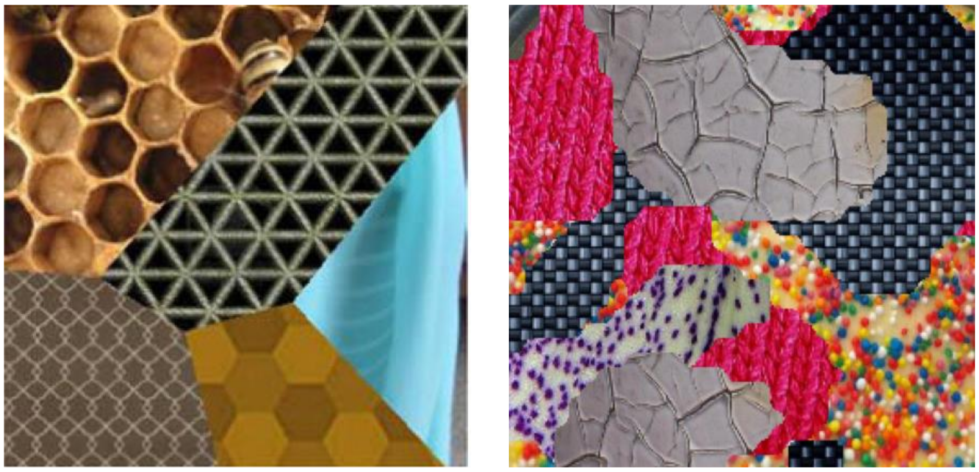


Fig. 3.5. A comparison between the generation techniques.

Perlin noise is a type of procedural generation technique widely used in the game development and visual effect industries to generate pseudo-random textures or terrain maps [95]. The smoothness and the amount of detail in the generated shapes can be manipulated by the grid size and the number of Octaves used in the Perlin noise function. This allows the user to have some degree of control over the random generation process. For each texture candidate in the query image, the generator created a 32×32 Perlin noise map (pixel values range from 0 to 1) with an Octave count of n ($n = 2$ by default, $n = 3$ if the number of texture candidates > 3). A cutoff threshold that ranged from 0.1 to 0.2 was randomly chosen to convert the 2D Perlin noise map to a binary mask. The binary map was then resized to match the size of the candidate texture image. A Gaussian blur filter was used to smooth the resized map before it was used to mask the candidate texture image. Finally, all the masked candidates were concatenated to form the synthetic query image. A visualization of this process is shown in Fig.3.6.

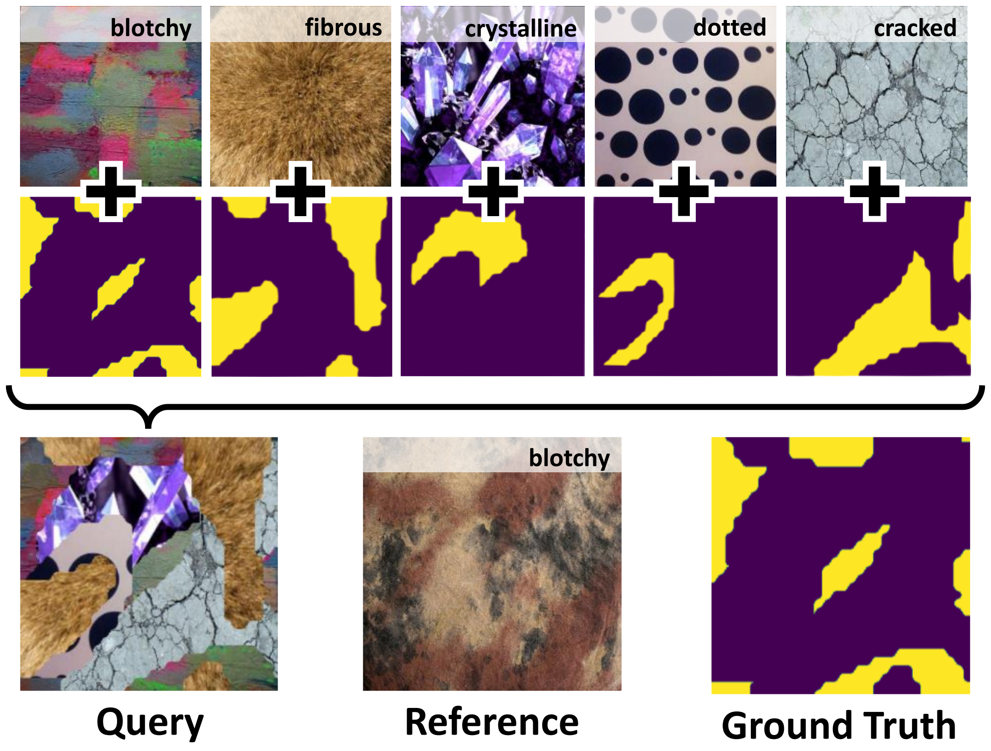


Fig. 3.6. The Perlin-Noise-based collage generation process. The yellow regions have pixel values of 1, whereas the purple regions are 0.

During training, all images were resized to 256×256, and up to five texture candidates were randomly sampled to form the query image. A reference image R^{i} that corresponded to the i^{th} class (the target class) in Q was randomly chosen. The pseudo-random mask of the i^{th} class was used as G^{i}. Together, Q, R^{i}, and G^{i} form the triple T^{i} in (3.6). All triples were randomly generated at each training or evaluation step. All random processes were tightly controlled by random seeds and random generators to ensure reproducibility. Similar to [130], among the 47 texture classes in DTD, 42 of them were used to form the training set, and the remaining five classes were used as an evaluation set.

C. Model Building and Ablation Study

Encoder: MOSTS employs a typical dual-branch encoder structure (Siamese encoder) found in many one-shot learning models (Fig.3.7). Since the encoder's throughput significantly contributes to the model inference time, the selection of an efficient backbone model for the encoder is vital for achieving real-time performance. This study conducted a benchmark test to identify the most suitable backbone model using seven popular models commonly utilized for building efficient segmentation models. The selected models included ResNet-based models[144] (i.e., ResNet-18, 34, and 50), EfficientNet [145], EfficientNet-v2 [146], MobileNet-v2 [147], and MobileNet-v3 [148]. All models were pre-trained on the ImageNet [149] dataset and converted to Float16 Intermediate Representation using the model optimizer from Intel OpenVINO (ver 2022.1). As the SBC used in this study was powered mostly by Intel hardware (Intel i7-8665UE CPU and Intel UHD Graphics 610 GPU), OpenVINO IR provided optimal performance. Fig.3.8 presents a bubble plot of the benchmark results for these models, with the bubble sizes indicating the relative sizes of each model. As observed from Fig.3.8, the chosen model, MobileNet-v3, demonstrated an inference speed more than twice as fast as the most accurate model (EfficientNet-

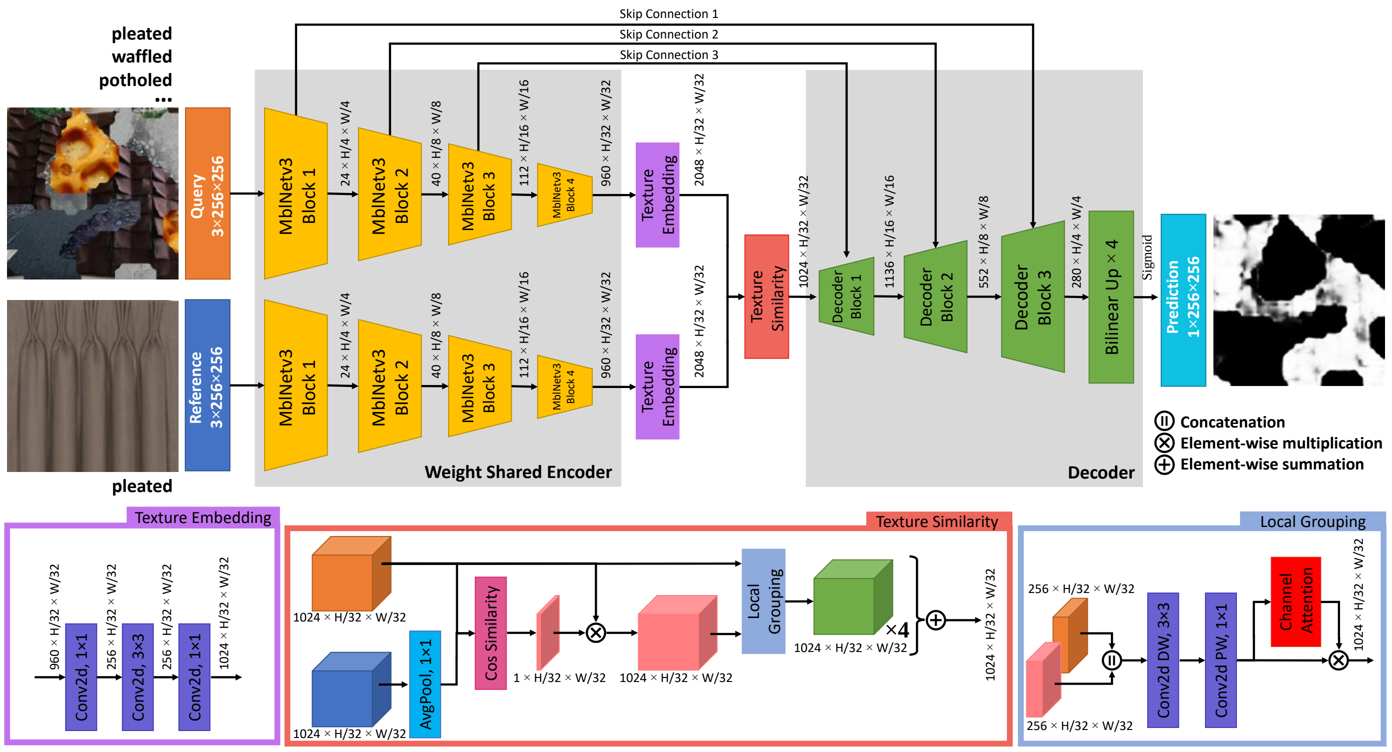


Fig. 3.7. The architecture of MOSTS.

v2), while showing only a 3% reduction in accuracy.

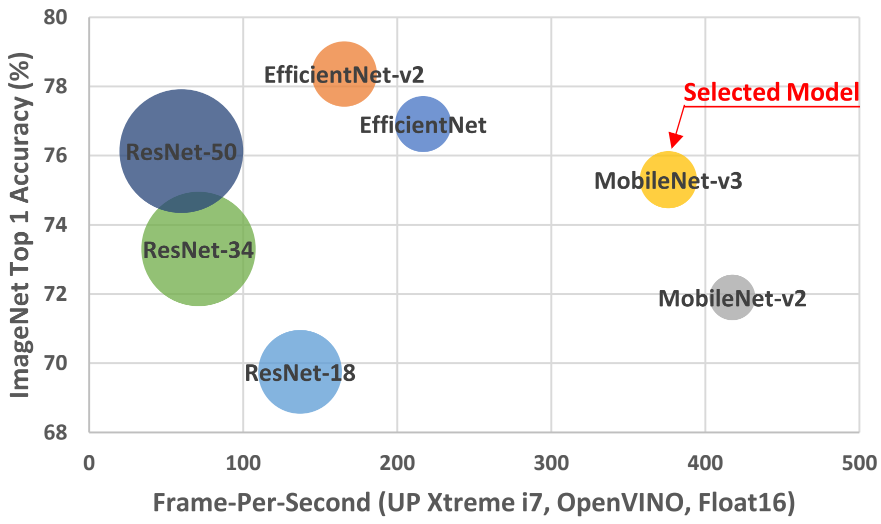


Fig. 3.8. The benchmark result of the backbone models on ImageNet with input size 224×224.

Texture Embedding: Given that the backbone encoder model was pre-trained on ImageNet for classification, its feature embedding might not provide optimal performance for texture segmentation tasks. Meanwhile, fine-tuning the backbone model could potentially jeopardize model robustness and generalizability, considering that the training dataset, DTD, is considerably less diverse than ImageNet. To overcome these limitations, a texture embedding block was introduced to refine the initial features into more appropriate texture feature embeddings. As illustrated in Fig.3.7, this block incorporated a 3×3 convolution layer flanked by two 1×1 convolution layers. Each convolution operation was succeeded by Batch Normalization and a ReLU activation function. This convolution block adopted a bottleneck structure, effectively compressing the channel dimension from its original size of 960 to 256. This design encouraged the model to learn to extract vital information from preliminary features efficiently.

Texture Similarity: Inspired by [150], the texture similarity block used cosine similarity as an attention mechanism (Fig.3.7) to emphasize the features of the query texture embedding if they showed strong similarities with the reference texture embedding in the spatial dimensions. First, an average pooling operation compressed the spatial dimensions of the reference texture embedding into a $1024 \times 1 \times 1$ tensor. Given that a visual texture essentially consists of a set of repeated primitive texels (i.e., texture pixel, the fundamental unit of a texture pattern) arranged on the image plane, its fundamental pattern is inherently spatially invariant. If the reference image effectively captures the essence of the target texture, the compressed tensor can be viewed as an encoded texel. This tensor was then used to compute the cosine similarity $s_{x,y}$ with the query texture embedding at each spatial position, enabling the identification and emphasis of similar texture features:

$$s_{x,y} = \frac{r \cdot F_{x,y}^{q}}{\|r\|_2 \cdot \left\|F_{x,y}^{q}\right\|_2} \tag{3.12}$$

The compressed reference tensor is denoted as $r \in R^{C\times 1\times 1}$ and the feature embedding of the query image is $F^q \in R^{C\times w' \times h'}$. The computed similarity map $s \in R^{1\times w' \times h'}$ encoded the similarity values $[-1,1]$ at each spatial position (x, y). The model masked the query texture embedding using the similarity map via element-wise multiplication.

Local Grouping: The local grouping block was initially used in [130] to aggregate features learned by different convolution branches. This study used it to aggregate the masked query features with the unmasked features to preserve information prior to the texture similarity block. It also allowed gradients to bypass the similarity block to refine the texture embedding directly. By organizing feature embeddings into smaller local groups, the model can keep related feature channels close and significantly decrease the number of convolution filters required. In this module, both masked and unmasked query texture embeddings were divided into four groups. Within each group, the embedding pair are concatenated and passed through a depth-wise (DW) and point-wise (PW) convolution block for aggregation (Fig.3.7). This specific structure was chosen for its efficiency. To further optimize feature selection, a learnable channel-wise attention module [151] was incorporated. This module let the model focus on useful features while suppressing the less relevant ones. Finally, the output features from the four local groups were combined using an element-wise summation.

Decoder: The decoder up-sampled the feature embedding via bilinear interpolation. The up-sampled feature was then concatenated with the residual feature provided by the skip connection. A 1×1 convolution layer subsequently aggregated these concatenated features. This procedure recurred in the three decoding blocks, followed by a final bilinear interpolation layer. This ensured that the model's output shares the same spatial dimensions as its input (Fig.3.7).

This study presents an ablation study to dissect the contribution of each model component. For ease of comparison, this study used the same four-fold validation groups for the DTD dataset defined in [130] (Table 3.1).

TABLE 3.1
VALIDATION GROUP DEFINITION FOR DTD

Groups	Validation Classes
i=0	'banded','blotchy','braided','bubbly','bumpy'
i=1	'chequered','cobwebbed','cracked','crosshatched','crystalline'
i=2	'dotted','fibrous','flecked','freckled','frilly'
i=3	'gauzy','grid','grooved','honeycombed','interlaced'

All the models in the ablation study were trained using the DTD dataset for 80 epochs with a batch size of 16. The training optimizer was SGD with a momentum of 0.9 and a weight decay of 0.0001. The initial learning rate was 0.001, and a step scheduler was used. The training was performed on a pre-built PC with 16G of RAM, an Intel i7-11700 CPU, and an NVIDIA GeForce RTX 3060 GPU (12Gb).

Table 3.2 presents the results of the ablation study, justifying the design choices made for

TABLE 3.2
ABLATION STUDY

Model	i=0	i=1	i=2	i=3
Baseline	49.1	42.8	52.7	44.2
+Tex Emb	48.9	42.8	48.3	43.8
+Tex Sim	48.4	47.3	53.4	44.1
+ Tex Emb & Tex Sim	53.2	46.9	55.1	50.3
+ Tex Emb & Tex Sim & Local Grouping	**55.6**	**49.4**	**59.3**	**50.8**

each module. A baseline accuracy (measured in mIoU) was established using a naive model, which concatenated the outputs of the Siamese encoder before directing the result to the depth-wise and point-wise convolution block and the decoder. The ablation study revealed that the addition of the Texture Embedding (Tex Emb) module did not independently improve accuracy. However, when combined with the Texture Similarity (Tex Sim) module, there was a noticeable increase in the four-fold average mIoU by 4.2, representing approximately a 9% improvement over the baseline model. The most significant improvement (about 14%) was achieved when the Local Grouping module was incorporated alongside the Tex Emb and Tex Sim modules.

D. Comparison with SOTA one-shot segmentation models

To assess the performance discrepancy between MOSTS and existing models, this study replicated the benchmark test conducted in [130], comparing MOSTS with OSTS [129] and OSTR [130] using identical four-fold validation groups as defined in Table 3.1. All models underwent

TABLE 3.3
COMPARISON WITH SOTA ONE-SHOT TEXTURE SEGMENTATION MODELS

Model	i=0	i=1	i=2	i=3	mean
OSTS	27.7	36.0	29.1	29.7	30.6
OSTR	52.6	47.3	53.8	49.6	50.8
MOSTS	**55.6**	**49.4**	**59.3**	**50.8**	**53.8**

training and testing under conditions that were identical to Section V.C. The results in Table 3.3 demonstrate that MOSTS, despite its smaller and simpler structure, outperforms OSTS and OSTR across all validation groups, achieving an average mIoU of 53.8. This represents an improvement in average mIoU by approximately 6% and 76% over models OSTR and OSTS, respectively.

3.7 Experimental Setup

A. Validation Dataset

This study aimed to explore the performance of MOSTS beyond synthetic data, specifically focusing on potential applications during real-time robot navigation. To quantify the proposed method's performance in real-world scenarios, this study collected and labeled a custom Indoor Small Objects Dataset (ISOD). ISOD contained 2,000 manually labeled images from 20 diverse sites, each featuring over 30 types of small objects randomly placed amidst the items already

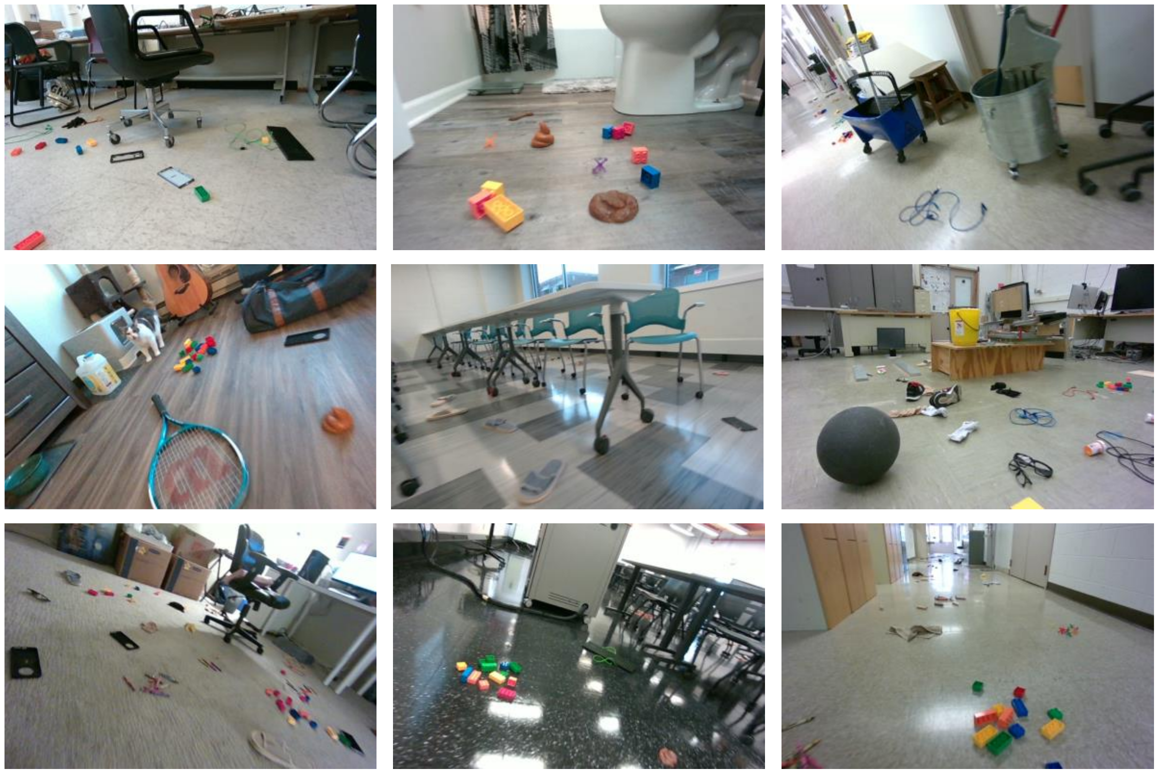


Fig. 3.9. Example RGB images from ISOD.

present in the scenes. These objects, typically ≤3cm in height, included LEGO blocks, rags, slippers, gloves, shoes, cables, crayons, chalk, glasses, smartphones (and their cases), fake banana peels, fake pet waste, and piles of toilet paper, among others. These items were chosen because they either threaten the safe operation of indoor mobile robots or create messes if run over. Example images from ISOD can be found in Fig.3.9. In addition to RGB images, ISOD also included corresponding depth images and IMU readings. To optimize storage space, the outputs of the ground plane masking node (the "masked image" in Fig.3.3) were logged as the query images for MOSTS instead of storing raw point clouds. A reference image of each floor type was also recorded using a smartphone. Data collection for ISOD was performed using a handheld device (Fig.3.10), with the operator videotaping the entire scene while moving around at walking speeds (≤1m/s). During collection, the device was held at varying heights (ranging from 0.2m to 1m) to introduce height variance. The operator was encouraged to simulate the operations of dynamically balanced robots by incorporating tilting motions. Each data collection session lasted three minutes per scene. One hundred samples were randomly selected and added to the ISOD from the raw data recordings. Subsequently, the data labeling team was instructed to annotate the drivable regions in the RGB images with positive labels (pixel value =1), marking the remaining areas with negative labels (pixel value =0).

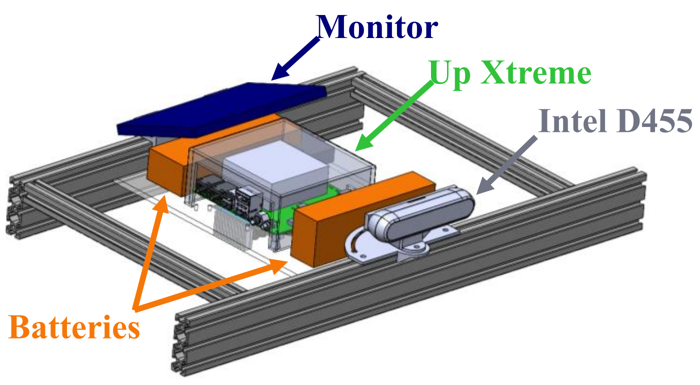


Fig. 3.10. The handheld device used for data collection.

B. Preparation for the Benchmark Models

It should be noted that this study used ISOD solely as a validation dataset, while MOSTS was trained utilizing the DTD dataset [143] via the method introduced in Section V.B. Training was conducted for 500 epochs with a batch size of 16, using the ADAM optimizer. Since indoor DAS using one-shot textural segmentation is a relatively recent research area, directly comparing the performance of the proposed method with existing methods proves challenging. Nevertheless, efficient RGB-D semantic segmentation models such as those detailed in RedNet [126] and ESANet [127] served as valuable performance indicators to understand and quantify the framework's efficacy.

As indoor semantic segmentation arguably poses a more complex task than DAS, unmodified versions of models RedNet and ESANet may pay unnecessary attention to objects unrelated to the DAS task. To ensure a fair comparison, this study retrained models RedNet and ESANet as DAS models using modified data from SUNRGB-D [128], a popular indoor semantic segmentation benchmark dataset. It contains 10,335 RGB-D images collected across 37 scene categories. A label reduction process was conducted to convert this semantic dataset into a drivable area dataset. First, this study filtered out samples from [128] whose areas contain less than 1% of "floor" or "carpet" labels. Second, the labels of the remaining 7,946 samples were converted into binary maps ("floor" & "floor_mat" =1, all others =0). These samples were then randomly divided into a training set and an evaluation set at a ratio of 4:1. The retrained models RedNet and ESANet achieved training accuracies (mIoU) of 82.4% and 88.0%, respectively.

For further comparison, vanilla versions of OSTR [130], RedNet, and ESANet were also pre-trained and included in the benchmark as baselines. While models RedNet and ESANet utilized

RGB and depth images as inputs during the tests, model in OSTR and MOSTS used the RGB (masked) and texture reference images as inputs. All image data were resized to 256×256. Lastly, all tested neural network models were converted into Float16 IR and deployed to the same SBC to estimate the inference speed.

3.8 RESULTS AND DISCUSSION

The performance metrics used in the validation test included accuracy (measured in mean Intersection over Union, mIoU) and inference speed (measured in Frame Per Second, FPS). As Table 3.4 demonstrates, OSTR attained an accuracy of 73% on ISOD. Training OSTR with the Perlin-Noise-based collage generator (as opposed to the Voronoi-based) resulted in an accuracy boost of +10.5%, while MOSTS trained with the Voronoi-based collage generator exhibited a decrease in accuracy (-6.6%). This indicates that synthetic training examples generated via the

TABLE 3.4
QUANTITATIVE RESULTS FROM THE VALIDATION STUDY

Method	Backbone	MIoU	FPS
Ground Plane Seg	-	82.7	100*
MOSTS	**MobNetv3**	**85.7**	**93.13 (48.22)**
MOSTS (Voronoi)	MobNetv3	79.1	93.13 (48.22)
MOSTS-Res34	Res34	86.2	22.77 (18.54)
MOSTS-Res50	Res50	86.3	17.60 (14.97)
OSTR	Res50	73.0	13.44 (11.85)
OSTR (Perlin)	Res50	83.5	13.44 (11.85)
RedNet	Res50	71.0	11.44
ESANet	Res34	80.7	21.71
RedNet (DAS)	Res50	85.1	11.44
ESANet (DAS)	Res34	83.0	21.71

The result of the proposed method is **bolded**. *: FPS was estimated using typical algorithm throughput. (): FPS adjusted for the latency of ground plane segmentation algorithm.

Perlin-Noise-based method better prepare models for real-world scenarios.

The pre-trained vanilla semantic models in RedNet and ESANet achieved accuracies of 71.0% and 80.7%, respectively. Fine-tuned DAS versions of these models showed improved accuracies, with RedNet reaching 85.1% and ESANet hitting 83.0%. This suggests RedNet possesses greater generalizability, whereas ESANet seemed to overfit the training data as it had higher training accuracy. By design, ESANet is more speed optimized. The inference speed results suggest that ESANet was about two times faster than RedNet. With an inference speed of 21.71 FPS, ESANet will suffice for most indoor robot perception needs. However, MOSTS outperformed ESANet in terms of speed by over two-fold while having competitive accuracy, thereby freeing up even more computational resources for subsequent robot control algorithms. The latency of the ground plane segmentation algorithm can be minimized through parallel processing. In that case, MOSTS could attain up to four times faster inference speed (93.13 FPS) than ESANet, positioning it as a superior choice for indoor DAS.

Further analysis of the performance differences resulting from various backbone selections was conducted, benchmarking MOSTS with ResNet-34 and ResNet-50 as backbones. The surge in inference speed was largely due to the faster backbone. Meanwhile, the accuracies of these ResNet-based MOSTS models were comparable to the proposed one, with MOST-Res50 exhibiting the highest accuracy in this test and an FPS similar to RedNet.

The validation study also included the accuracy of the ground plane segmentation algorithm. This accuracy was computed by comparing the ground truth with the masked image, where blackened pixels were mapped to 0 and all others to 1. Despite the algorithm's inability to identify small objects ($\leq 5cm$ in height), it achieved a mIoU of 82.7% as the pixel areas of these small objects account for only a small fraction of the entire scene. Thus, the complete omission of these objects did not necessarily result in a significant mIoU reduction. This logic is also applicable to the results of the neural network models.

This study included some qualitative results from the validation test to gain better insights and further demonstrate the performance differences between the tested methods. Fig.3.11 illustrates that ISOD poses significant challenges for indoor semantic models. Both RedNet and ESANet struggled to identify floors with small objects. The DAS versions demonstrated noticeable

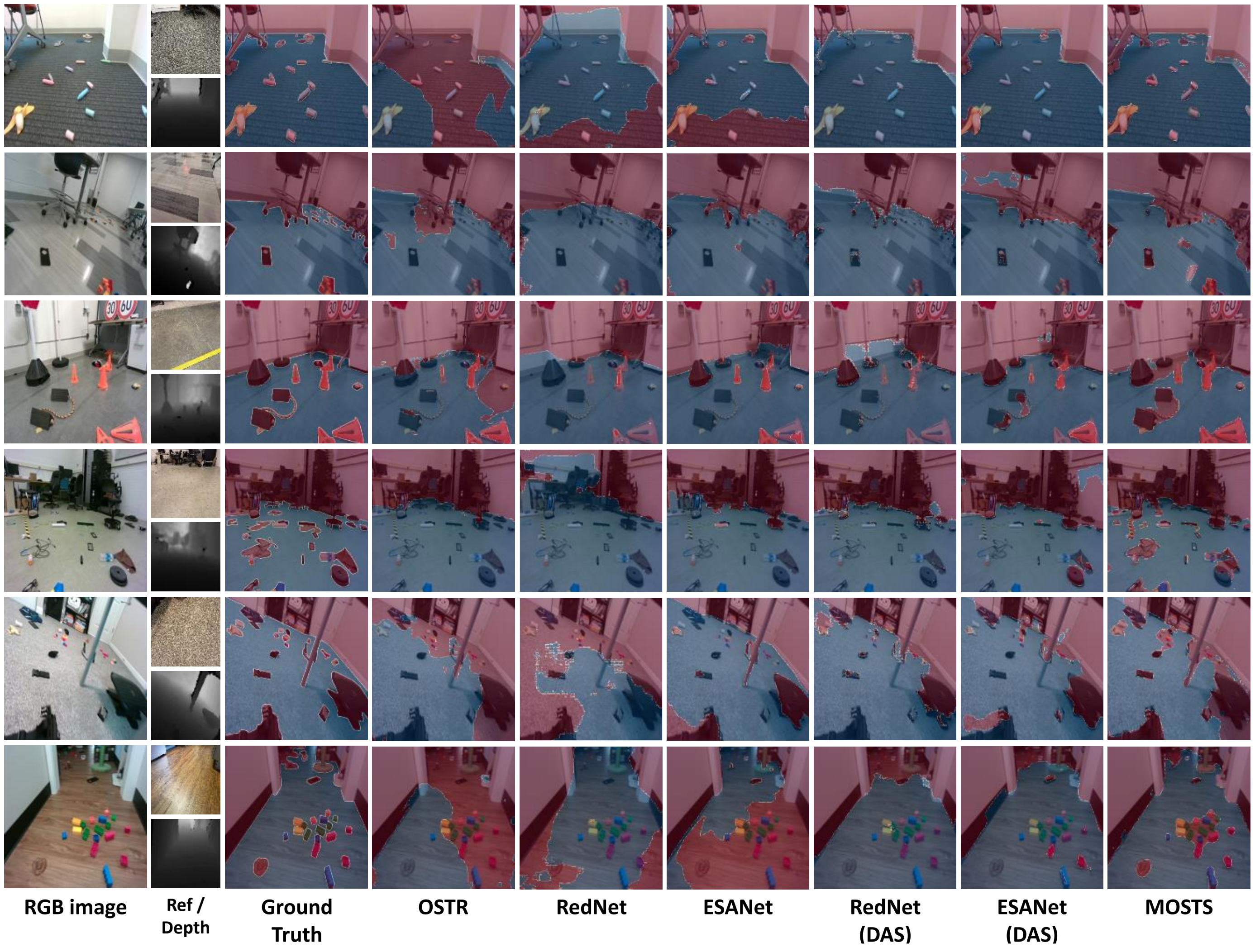


Fig.3.11. Qualitative results from the validation test. Drivable areas are highlighted in **blue**, whereas obstacles are highlighted in **red**. Reference images and depth images are shown in smaller size.

improvements, as they had a more general open set for categorizing non-floor objects. While both models managed to capture floor regions even with scene tilting, they largely overlooked smaller objects in the scenes due to the lack of such examples in their training data from SUNRGB-D. In contrast, the proposed model MOSTS can identify small, low-lying objects irrespective of shape, color, and size. Trained entirely using synthetic images, it does not require manually labeled small object training examples like the semantic models. Moreover, as the proposed method first estimates the ground plane using an analytic algorithm, it rarely produces false positive labels on walls and furniture. This safety feature can help reduce the uncertainty inherent in data-driven methods like neural networks.

3.9 CONCLUSION AND FUTURE WORK

This study presented a fast, motion-robust RGB-D Drivable Area Segmentation (DAS) framework for indoor robot navigation. This framework used a ground plane segmentation algorithm to estimate the drivable area, which was further refined using a Miniature One-Shot Texture Segmentation (MOSTS), a rapid texture-oriented one-shot segmentation neural network. The MOSTS model was trained using a novel Perlin-noise-based collage synthesis technique. An Indoor Small Objects Dataset (ISOD) was created to evaluate the proposed framework's efficiency. The validation test demonstrated that MOSTS offered competitive accuracy and considerably higher inference speed when compared to two state-of-the-art efficient RGB-D semantic segmentation models. Moreover, the proposed framework was deployable on a low-power Single Board Computer (SBC), capable of delivering real-time performance.

Despite the promising results, future studies are needed to handle the limitations of the proposed method. The DAS framework performs best when a single general floor texture type (e.g., tiled, wooden, carpeted) is present in the scene but suffers from floors featuring mixed texture types.

The potential applications of MOSTS extend beyond refining DAS. For example, Fig.3.2 shows that MOSTS can effectively detect trash and litter on outdoor terrains like sand and snow. When given an image of well-maintained asphalt, MOSTS can identify cracks on a runway image. It demonstrates that MOSTS can facilitate applications like autonomous cleaning robots and robot runway patrol. Importantly, the same MOSTS model used in the validation test was directly transferred (without any retraining or fine-tuning) to generate the predictions in Fig.3.2. This illustrates the flexibility of MOSTS, as it can be easily repurposed and adapted to new environments or tasks.

CHAPTER 4

A VISION-GUIDED SHARED-CONTROL FOR NAVIGATING A RIDING BALLBOT SYSTEM

4.1 ABSTRACT

This study introduces a shared-control approach for collision avoidance of mobility devices, particularly one based on a self-balancing riding ballbot (a.k.a. PURE). Due to its ballbot drivetrain and torso-dynamics estimation system, PURE has unique features such as its dynamic stability, omnidirectional motion, and hands-free control interface. The proposed shared-control approach uses a RGB-D vision system, as well as secondary sensors like LiDAR and Time-of-Flight sensors, to enable safe and intuitive navigation through deceleration assistance and haptic/audio feedback. The haptic feedback is introduced by a novel Passive Artificial Potential Field method that provides motion resistance when the operator drives PURE towards an obstacle. We first conducted a robotic system evaluation to quantify the deceleration performance of PURE and its shared-control system. Then, a human-robot interaction (HRI) evaluation was done to explore the performance and effort of navigating PURE with and without shared-control by manual wheelchair users (mWCUs) compared to able-bodied users (ABUs). Twenty test subjects (10 mWCUs + 10 ABUs) operated PURE through two obstacle courses (S-Turn and Zigzag shapes) with wide and narrow widths. Repeated Measures MANOVA tests assessed objective (collision index and completion time) and subjective metrics (NASA TLX scores and a post study questionnaire) to

evaluate the performance of the proposed shared-control scheme. Results from the HRI evaluation suggested that: 1) shared-control reduced collisions without compromising travel speed; 2) shared-control reduced mental demand, physical demand, effort, and frustration, while improving perceived performance; and 3) there were virtually no differences between ABUs and mWCUs in terms of completion time and collision likelihood. The post-study questionnaire found the shared-control to be intuitive, natural, and safe. Therefore, this study concluded the proposed shared-control approach successfully provided effective collision avoidance assistance for this unique self-balancing riding device.

4.2 INTRODUCTION

Navigating through crowded environments with a mobility device can be mentally taxing, requiring significant cognitive effort from the rider. However, employing a shared-control approach that combines the capabilities of a semi-autonomous device with the input of the human rider has the potential to enhance operational performance. By distributing the control effort

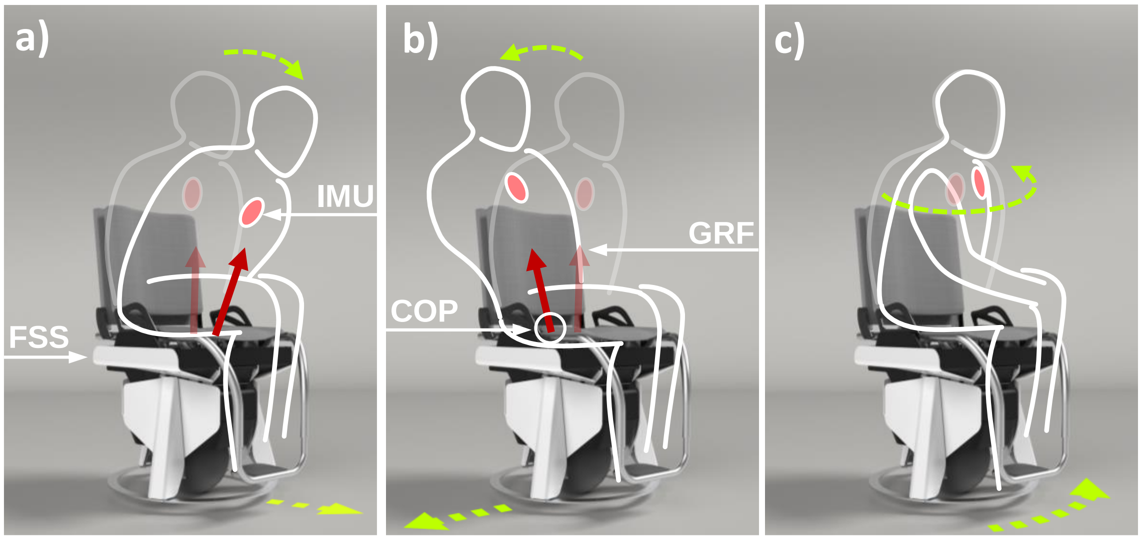


Fig. 4.1. PURE control schematics: The rider can control PURE's motion in a) forward-backward and b) left-right directions by leaning their upper body (COP: center of pressure, GRF: ground reaction force vector). c) Turning control is achieved by twisting the upper body.

between the device and the rider, the shared-control approach aims to alleviate the cognitive burden on the rider and improve the overall efficiency and effectiveness of the mobility device.

Our research group has developed a novel mobility device called PURE (Personalized Unique Rolling Experience). PURE is a self-balancing riding ballbot system that affords the rider a hands-free control interface, omnidirectional motion, and a compact form factor [152], [153]. Leveraging the self-balancing nature of the ballbot system, riders simply lean in the direction they want to go, and the device will naturally comply. To accommodate for riders with reduced torso range of motion, the Torso-dynamics Estimation System (TES), consisting of an instrumented seat (Force Sensing Seat, FSS) and a wearable sensor (inertial measurement unit, IMU), are used to control direction and speed [154] (Fig.4.1). However, driving a mobility device with this level of maneuverability presents unique challenges. First, the omnidirectional ability to spin in place or travel in both longitudinal and lateral directions requires riders to be constantly cautious for possible collisions or falling risks from all directions. Second, PURE's under-actuated dynamics [155] requires it to accelerate forward to bring the center of gravity backward (proportional to the current driving speed) prior to applying deceleration on the ball. Estimating the braking/stopping distance and timing of this highly maneuverable device is a challenging task, particularly for novice riders navigating in congested areas.

To alleviate the demands on rider proficiency and effort, this study proposed a shared-control approach using a novel Passive Artificial Potential Field (PAPF) method. This proposed shared-control method followed an Artificial Potential Field (APF) [156] formulation with additional adjustments to accommodate for the dynamic characteristics of PURE. The goal of PAPF shared-control was to minimize the rider's collision avoidance efforts by providing deceleration assistance and intuitive haptic feedback.

This study consisted of two evaluation experiments. First, a robotic system evaluation that focused on assessing the deceleration performance of PURE and its shared-control system. Second, a human-robot interaction evaluation was conducted to assess the efficacy of the proposed approach with 20 participants (10 manual wheelchair users (mWCUs) and 10 able-bodied users (ABUs)). The evaluation involved having the participants navigate PURE through obstacle courses with complex shapes and challenging widths. This evaluation aimed to answer the following research questions:

1. Does the performance improve when using the shared-controller?
2. Does the operational effort decrease when using the shared-controller?
3. Are there performance and preference differences between ABUs and mWCUs in relation to the use of the shared-controller?

The key contributions of this study are: 1) integration of a vision-guided perception system and a high-level controller with PURE for the purpose of achieving real-time shared-control; 2) introduction of a novel PAPF shared-control paradigm specifically designed to ensure the safety of a self-balancing riding device like PURE; and 3) conducting a HRI evaluation to evaluate the effectiveness of the proposed shared-control paradigm and gather valuable feedback for further improvement.

4.3 RELATED WORK

A. Powered Wheelchair Shared-Control

The development of "smart" powered wheelchairs [17], [18], [21], [34], [157], [158] has been driven by the intention to accommodate individuals who are unable to safely operate powered wheelchairs due to physical limitations or cognitive challenges. Initially, the focus of development was on creating autonomous wheelchairs [157], but achieving fully automated wheelchairs capable of navigating complex and unfamiliar environments remains highly challenging. Additionally, some studies [44], [45] have emphasized the importance of leveraging residual motor and cognitive capacities to promote independence, making fully autonomous wheelchairs potentially counterproductive.

In response to these challenges, shared-control has emerged as an alternative solution, where the human rider makes high-level decisions while the robot agent is responsible for low-level motion control and collision avoidance. The philosophy behind shared-control is to reduce the cognitive and physical effort required to operate a powered mobility device by offloading certain tasks to the robot agent. This allows users to concentrate more on reaching their destination or performing social activities rather than the act of navigating itself [11], [46]–[48].

Traditionally, wheelchair shared-control can be categorized into two main types: deliberative and reactive. Deliberative shared-control follows a “Sense-Plan-Act” formulation [159], using the robot and environment models to plan ahead (using techniques such as Rapidly Exploring Random Trees (RRT) [49], Model Predictive Control (MPC) [50], [51], Reinforcement Learning (RL) or Bayesian methods [53]). These shared-control approaches are often complex, considering the current and future states of the robot and the environment to plan efficient and risk-averse paths. On the other hand, reactive shared-control directly maps the current state of the robot and the environment into the robot’s control signal (including Vector Field Histograms (VFH) [54], Dynamic Window Approach (DWA) [55], and Artificial Potential Fields (APF) [47], [56]–[59]). Reactive shared-control is fast, simple, and robust but cannot reason about complex navigation tasks and find optimal paths. While both methods have their strengths and weaknesses, a reactive shared-control paradigm aligns more closely with this study's research goal: making the best use of the human's abstract reasoning and the robot's high control bandwidth.

B. Artificial Potential Field Based Shared-Control

Artificial Potential Field (APF) based shared-control approaches have been widely used in wheelchair shared-control and related domains. Petry et al. [56] generated an attractive force from the user command and repulsive forces from the obstacles detected with ultrasonic sensors. The resultant wheelchair's behavior was simply the sum of the attractive force and all the repulsive forces at a given position. Urdiales et al. [47] treated human and robot commands as different goals in an APF. These goals were weighed by the agent's respective local efficiencies at each time instant to encourage emergent cooperation between humans and robots. The study by Li et al. [57] proposed a wheelchair brain-machine interface that utilized an APF for safeguarding the system from collisions in case of noisy and faulty control signals. Multiple studies [54], [58], [59] used similar "force field" approaches to establish haptic bilateral communication between the wheelchair and the user.

Besides wheelchair shared-control, APF has been applied in other areas. APF was used to calculate braking force for a passive intelligent walker [160]. Similar to the proposed method, the APF approach in [160] only applied deceleration effort to the system. The driving effort of the system was provided by the human rider. The work by Gottardi et al. [161] proposed a shared-control teleoperation framework for a robotic arm using an APF approach improved by the dynamic generation of escape points around the obstacles. The study by Baek et al. [41] proposed an APF approach using a time-derivative sigmoid function to enable shared-control for whole-body tele-locomotion of a wheeled humanoid.

While APF-based shared-control approaches have been studied and shown promising results, their specific adaptation for self-balancing riding devices remains unexplored. Unlike existing studies [41], [161], where APF can provide motion correction and haptic feedback separately, this study finds that the motion correction and haptic feedback for a self-balancing riding device are interconnected due to its dynamic nature. The haptic feedback achieved through tilting the self-balancing riding device directly affects its motion, and vice versa, resulting in a combined effect. Moreover, traditional APF approaches proactively push the device away from obstacles even when the device is stationary [162]. However, soft collisions, dockings, and close proximity social interactions are common scenarios where proactive pushing is not suitable. Moreover, the conventional shared-control approach is known to exhibit control oscillation issues in narrow passages [47], [163]. This poses a significant challenge for tight indoor use and pronounced control oscillations could lead to unstable states. Furthermore, surrounding objects can affect the motion of the device even when it is moving parallel to them, leading to unnecessary speed reduction [162]. Thus, a revised approach to the traditional APF approach is needed for these situations.

C. Ballbot Systems

A ballbot is a type of self-balancing mobile robot that rides on top of a ball. The ballbot was first introduced by [7] and was later studied by [8], [9], [42], [43] and many others as mobile robot platforms, and assistive or interactive robots. Most ballbot drivetrain designs utilize three or more omniwheels to drive the ball. By carefully configuring the contact angles between the omniwheels and the ball, the drivetrain enables the ballbot to move omnidirectionally in any direction and rotate in place. A traditional ballbot system achieves self-balancing behavior by tracking the system's center of mass (COM). When the ballbot detects that the projection of the COM moves outside of the base of support, it will accelerate to align with the COM. Its self-balancing characteristic allows it to carry heavy payloads with a high center of mass while maintaining a small footprint. However, since the base of support of a ballbot is a small contact area between the ball and ground, the ballbot is constantly moving, or dithering, even when trying to self-balance in a set location. Leveraging its high maneuverability and minimal footprint, the ballbot can access existing infrastructures designed for able-bodied individuals.

Developing a shared-control paradigm to safeguard a ballbot system like PURE is a non-trivial task. Conventional braking systems cannot be used on a dynamically-stable system like a ballbot because holding the ball stationary would disrupt the device's ability to maintain balance through dithering. The acceleration and deceleration of a ballbot are closely coupled with its tilting motion, i.e., safe deceleration must be achieved by tilting the device in the opposite direction. Moreover, unlike traditional joystick-controlled wheelchairs, where the rider's command signal can be completely overwritten, the passive compliance nature of a riding ballbot allows the rider to move the device by significantly shifting their COM. These unique challenges invalidate most existing shared-control paradigms, emphasizing the need for a novel shared-control design that caters to the safety requirements of self-balancing systems like PURE. By studying and developing such a paradigm, this study aims to share and generalize its findings to mobile robots with similar or simpler dynamic characteristics as ballbots.

4.4 METHODOLOGY

A. System Specifications

The hardware design of the PURE Gen2 prototype focuses on optimizing compactness and safety [152], [153]. The overall footprint and size of PURE are approximately that of a seated person, and the system weighs approximately 37 kg. The current prototype has an estimated

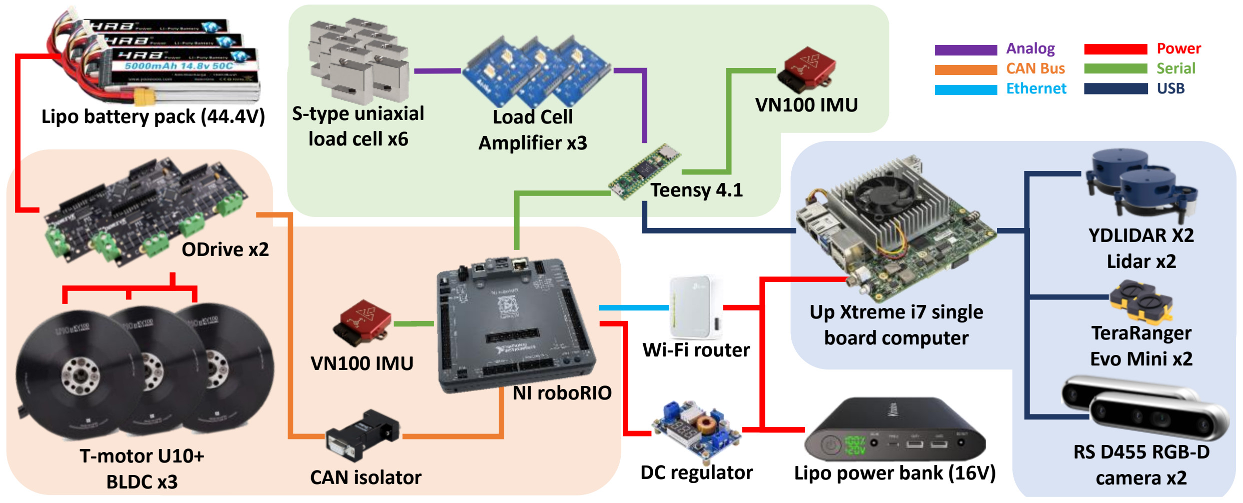


Fig. 4.2. PURE hardware setup and communication schematics. The low-level dynamic controller is highlighted in orange. The TES is highlighted in green. The high-level controller is highlighted in blue.

runtime between 2.5 to 4 hours, and it can achieve a top speed of 2.3 m/s. PURE is comprised of three electronic modules (Fig. 4.2): 1) A low-level dynamic controller, which provides state feedback of the drivetrain, ensures system balance, and performs velocity tracking using an LQR-PI controller [153]. 2) The Torso-dynamics Estimation System (TES) [154], which measures the rider's upper-body movements and maps them to velocity control signals. 3) A high-level controller, responsible for environmental awareness and collision prevention. The three brushless DC motors (BLDCs) (T-motor U10+, T-motor Inc., China) are powered by a pack of three LiPo batteries (HRB 4S-6000mAh LiPo, HRB Inc., USA), while the remaining electronics are powered by a laptop power bank (Volessence 50000mAh Laptop Power Bank). The communication and control loop rates for the LQR-PI, TES, and shared-control are set to 400 Hz, determined by the current computation capacity. The motor driver's torque/current control loop operates at a frequency of 8,000 Hz, ensuring precise and smooth trajectory tracking.

The high-level controller is hosted by a low-power Single Board Computer (SBC), specifically the UP Xtreme i7 (UP Bridge the Gap, USA), running Ubuntu 20.04 and ROS Noetic. The SBC was connected to a speaker (AVL Mini Speaker System, China), providing audio

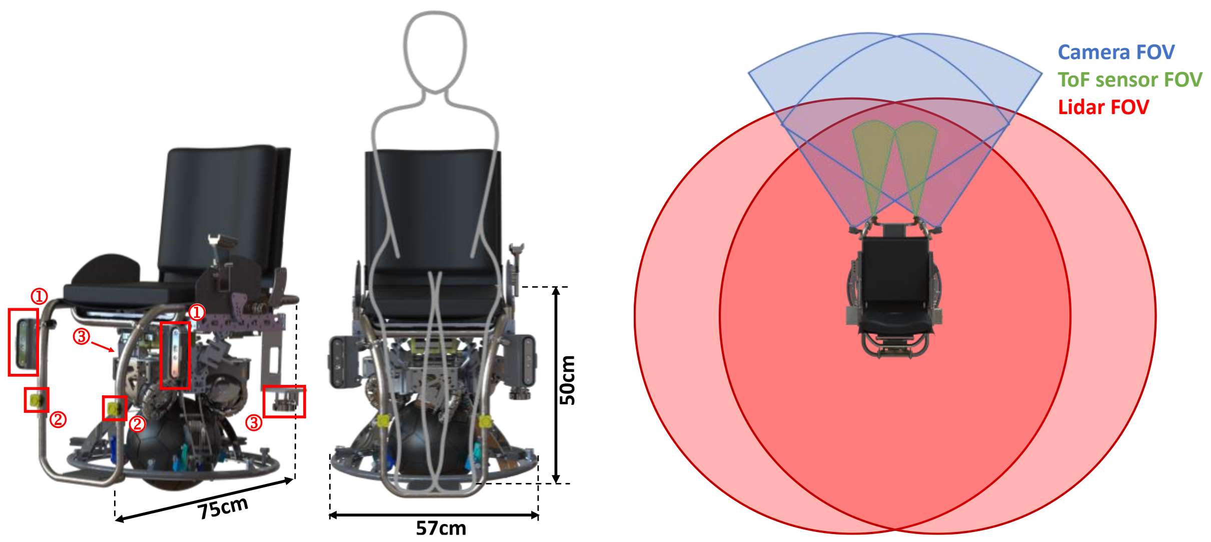


Fig. 4.3. PURE perception system sensor placement and FOV schematics. ①RGB-D camera, ②ToF sensor, and ③LiDAR.

feedback and issuing alarms when the chair approached obstacles. The perception system of PURE consisted of two RGB-D cameras (RealSense-D455, Intel Co., USA) as the main sensors, as well as two Time-of-Flight (ToF) distance sensors (TeraRanger Evo Mini, TERABEE, France) and two single-beam LiDAR sensors (YDLIDAR X2, Shenzhen EAI Technology Co., China) as secondary sensors. By combining the Field of Views (FOV) of all sensors, the perception system could detect obstacles from a full 360° perspective (Fig. 4.3). The effective FOV of each RGB-D camera is 87° wide × 58° height with a measurement range of 0.32 m to 7 m. The ToF sensors were added to compensate for the RGB-D camera's deficiency in close range. Each ToF's FOV is 27° × 27° with a measurement range of 0.03 m to 3.3 m. The LiDAR sensors were added to safeguard the sides and the rear of PURE. Each single-beam LiDAR has FOV of 360° and a measurement range of 0.1 m to 10 m. The update rate of the RGB-D camera was 30 Hz, while the update rates for the LiDAR and ToF sensor were 10 Hz and 40 Hz, respectively.

B. Control Architecture

The TES utilized a Force Sensing Seat (FSS) equipped with six specially oriented uniaxial load

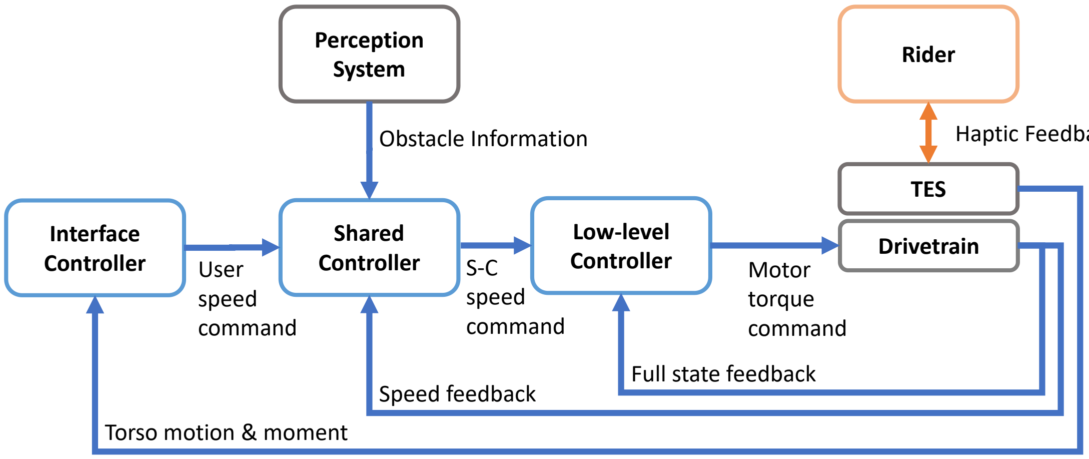


Fig. 4.4. PURE control architecture.

cells to collect force measurements, which are then used to infer the reaction forces and moments generated by the rider's upper body on the seat in three Euclidean axes [154]. To obtain the upper body twist in Euler angles, a wearable IMU (VN-100, VectorNav, USA) was mounted on the rider's chest. To mitigate the impact of vibration and magnetic disturbances originating from the drivetrain, a first-order low-pass (Butterworth) filter with a cut-off frequency of 1 Hz was employed to filter the FSS moment readings. The interface controller (Fig.4.4) used the following mapping function to map motion signals to velocity command signals (revised from [153]):

$$v_x^{usr}, v_y^{usr}, \omega_z^{usr} \leftarrow f_{TES}(\theta_x^{usr}, \theta_y^{usr}, \theta_z^{usr}, M_x^{usr}, M_y^{usr}) \tag{4.1}$$

Where $\theta_x^{usr}, \theta_y^{usr}$ and θ_z^{usr} represent the user's torso Euler angles measured by the wearable IMU and M_x^{usr} and M_y^{usr} are the user's torso moments measured by the FSS in the frontal and sagittal planes, respectively. The user command turning rate ω_z^{usr} is directly proportional to θ_z^{usr}, which represents the yaw angle of the upper body relative to the drivetrain. Meanwhile, the user command velocities v_x^{usr} and v_y^{usr} which correspond to the forward-backward (frontal) and left-right (sagittal) directions, are determined by a combination of upper body lean angles θ_x^{usr} and θ_y^{usr} and moments M_x^{usr} and M_y^{usr}. (Lean angles $\theta_x^{usr}, \theta_y^{usr}$ were added beyond [153] to compensate for the riders who might have lower upper body weight and the noise in moment measurements. See APPENDIX B) Although PURE's passive compliance makes it naturally drivable even without the help of the TES, augmentations of the rider's control signal (e.g., leaning motion captured by the FSS and IMU) are necessary for riders who have limited range of motions due to severe injury or deficit level. Additionally, the spinning control is only attainable using the sensor signal captured by the wearable IMU. To accommodate for different riders' range of motions and upper body weights, a set of weighting factors were used to adjust the contributions and sensitivities of the lean angle and moment measurements.

The perception system sensor measurements, in the form of point clouds, captured by all the sensors were combined and sent to a ground plane segmentation node [164]. Utilizing an algorithm from Chapter 3 (APPENDIX A), this node robustly separated obstacle points from the ground plane. This separation was achieved even under the tilting motion and vibration of PURE, preventing any misinterpretation of the floor as obstacles. The algorithm operated at an output rate of 30 Hz with a typical processing latency of 7 to 13 ms, which adequately met the real-time requirements of PURE. Taking into account the sensory information and the current velocity of the system, the shared-control system modified the user's velocity command to minimize the occurrence of undesired collisions. The implementation details of shared-control are described in section 4.3.C.

The shared-control speed command vector derived from the aforementioned controllers (Fig.4.4) was transmitted to the low-level controller, enabling the desired translational and spinning motions while ensuring dynamic stability of the robot–rider system. To achieve control over these motions, an LQR-PI cascaded control scheme [153] was employed. The ballbot drivetrain of PURE was divided into three 2D planar models, corresponding to the sagittal, frontal, and transverse planes. A wheeled-inverted pendulum (WIP) model was used to model the dynamics in the frontal and sagittal planes. Their system dynamics were obtained in the form of:

$$\dot{\boldsymbol{s}}_{\boldsymbol{j}} = f_j(\boldsymbol{s}_{\boldsymbol{j}}, \tau_j) \tag{4.2}$$

where $\boldsymbol{s}_{\boldsymbol{j}} = [\theta_j, \dot{\theta}_j, \dot{\phi}_j]$ is the state vector, θ_j is the chassis tilt angle, $\dot{\theta}_j$ is the chassis tilt angular rate, $\dot{\phi}_j$ is the ball speed, $j = x$ for the frontal plane, and $j = y$ for the sagittal plane. Optimal control gains $\boldsymbol{k}_{\boldsymbol{lqr}}$ were obtained for using the linearized system dynamics, such that the reference ball torque for WIP models were obtained using:

$$\tau_{j,ref} = \boldsymbol{k}_{\boldsymbol{lqr}}(\boldsymbol{s}_{\boldsymbol{j,ref}} - \boldsymbol{s}_{\boldsymbol{j}}) \tag{4.3}$$

The output torque was further utilized to obtain the reference wheel speed:

$$\dot{\phi}_{j,ref} = f_{j3}(\boldsymbol{s}_{\boldsymbol{j}}, \tau_{j,ref}) \tag{4.4}$$

where f_{j3} outputs the reference wheel speed of state vector. A similar method was utilized to obtain the reference wheel torque $\tau_{z,ref}$ and reference wheel speed $\dot{\phi}_{z,ref}$.

These reference torques $\tau_{j,ref}$ and reference speed values of the ball $\dot{\phi}_{j,ref}$ were then transformed into the corresponding torque $u_{n,ref}$ and reference speed $\dot{\psi}_{n,ref}$ of each individual motor ($n = [1,3]$). Additionally, a Proportional-Integral (PI) controller was cascaded within each motor to compensate for unmodeled friction while feeding forward the reference motor torque:

$$u_n = k_{Pn}(\dot{\psi}_{n,ref} - \dot{\psi}_n) + k_{In}(\int \dot{\psi}_{n,ref} - \dot{\psi}_n) + u_{n,ref} \tag{4.5}$$

where k_{Pn} and k_{In} are tuned proportional and integral control gains, and u_n is the final torque command that is received by the motor drivers.

C. Passive Artificial Potential Field Shared-Control

In the conventional APF setup, the robot's motion is determined by the sum of the attractive force from the goal and the repulsive force from the obstacles [156]. However, in the context of shared-control, where the long-term motion goal is determined by the rider, only the repulsive force is considered. This repulsive force can be calculated using the FIRAS (Force Inducing an Artificial Repulsion from the Surface) function [156], which is defined as:

$$f^{APF} = \begin{cases} \eta \left(\frac{1}{\delta} - \frac{1}{\delta_{thre}}\right)^2 & \text{if } \delta \leq \delta_{thre} \\ 0 & \text{if } \delta > \delta_{thre} \end{cases} \tag{4.6}$$

where δ_{thre} is a distance threshold representing the obstacle's radius of influence (m), δ is the minimum distance between the robot and the obstacle (m), and η is a scaling factor that can be adjusted for each axis. The quadratic behavior of the function ensured that the device exerts maximum effort to avoid collision when in close proximity to obstacles (Fig. 4.5). Since it is challenging to accurately estimate the shape and size of obstacles and compute δ, a simple adaptation was to treat all distance measurements as tiny obstacles. In this approach, the overall repulsive force was estimated by taking the average of all the forces. The repulsive force was calculated for each obstacle point (N points in total, measured by the sensors) using the relative distance δ_n and the relative angle α_n of the obstacle point with respect to the positive x-axis of the robot:

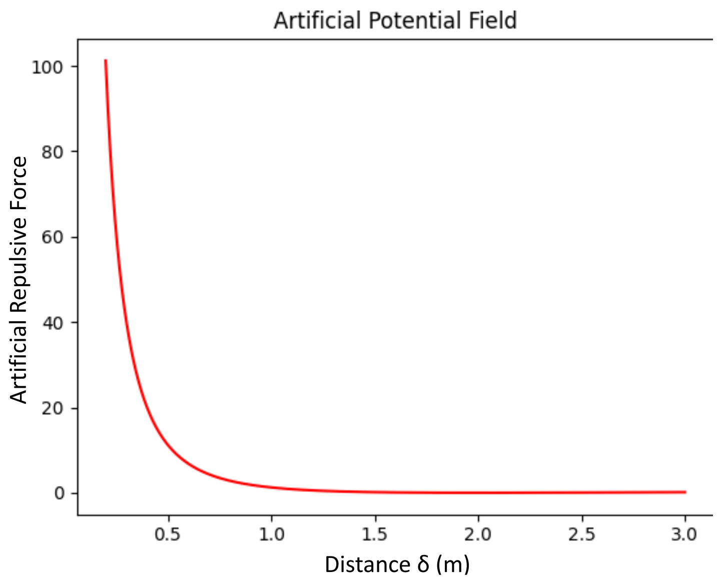


Fig. 4.5. The APF force profile with distance δ ranging from 0 to 3m. η is set to 5 and δ_{thre} is set to 2 for this visualization.

$$f^{APF} = \begin{cases} \frac{1}{N}\sum_{n=1}^{N} g(\alpha_n)\,\eta \left(\frac{1}{\delta_n} - \frac{1}{\delta_{thre}}\right)^2 & \text{if } \delta \leq \delta_{thre} \\ 0 & \text{if } \delta > \delta_{thre} \end{cases}$$

$$\mathrm{g}(\alpha_n) = \begin{cases} \cos(\alpha_n) & \text{for } f_x^{APF} \\ \sin(\alpha_n) & \text{for } f_y^{APF} \end{cases} \tag{4.7}$$

PAPF calculated the repulsive forces f_x^{APF} and f_y^{APF} independently, considering the relative position of the obstacle. For instance, if an obstacle point was positioned in the front-right region of the robot but did not directly overlap with the robot's frontal area, it generated a repulsive force only in the left-right* direction. (*If the repulsive force does not directly overlap with the robot's frontal area, this algorithm will only generate lateral repulsive force to prevent unwanted slow down.) This adjustment minimized unwanted resistance and allowed for smoother navigation in confined spaces. Furthermore, PAPF only considered obstacle points within the path of motion, defined by the semicircle of the command velocity vector. It resisted the rider's motion when approaching obstacles but did not introduce acceleration or proactively push the device away. This approach eliminated control oscillations and avoided unnecessary movement when the ballbot was stationary or moving parallel to objects.

There were many ways to incorporate f^{APF} into the robot's motion control loop. In this study, f^{APF} was first saturated between $[-1, 1]$ and then scaled by the norm of user commanded velocity signal, $v^{usr} \in [-1,1]$, to ensure it was only effective when the rider is driving. It was then added to the user commanded velocity to form the updated velocity command v^{ideal}. This approach of using f^{APF} as a commanded velocity correction term was easier to implement and more robust against latency compared to injecting it into the torque control loop. Similar practices can be found in [41], [161].

$$v^{ideal} = v^{usr} + (f^{APF}|v^{usr}|) \tag{4.8}$$

As mentioned earlier, the rider on PURE can continue to lean and drive towards an obstacle despite the shared-control's effort to slow down (Fig.4.6b). To address this issue, the shared-

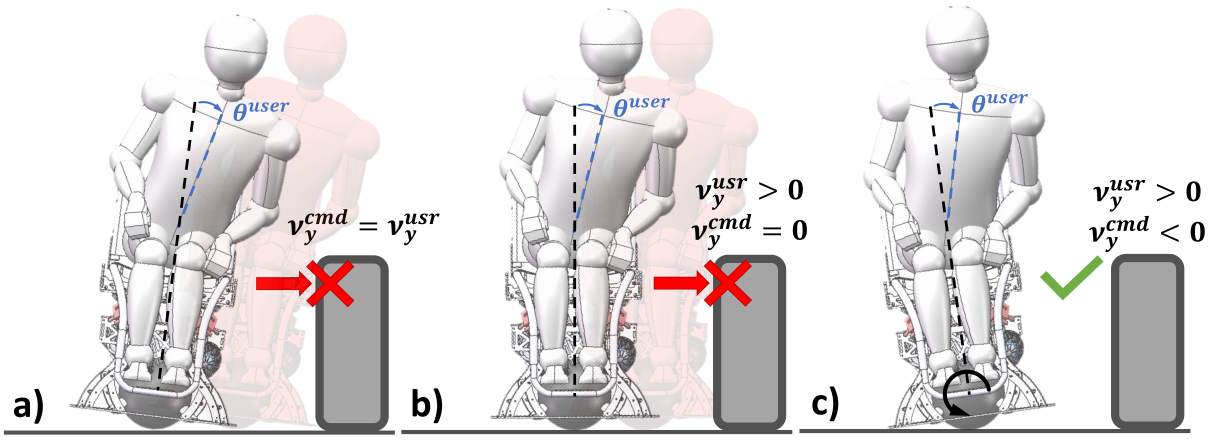


Fig. 4.6. PURE's behaviors under same operator leaning motion and different control paradigms. a) No Shared-Control, the commanded velocity is equal to user velocity command and collision is possible; b) Shared-Control without cascaded speed tracking, the operator can still force PURE to drive towards an obstacle despite the commanded velocity has set to zero by the shared-control; c) Shared-Control with cascaded speed tracking, an opposing commanded velocity is given to offset the operator's leaning motion and establish a new dynamic stable position.

control system incorporated a cascaded speed tracking loop. It compensated for undesired velocity by applying a counter lean and deceleration proportional to the unwanted velocity:

$$v^{comp} = v^{ideal} + \zeta\left(v^{ideal} - v^{fb}\right) \tag{4.9}$$

where ζ is a scaling factor that controlled the aggressiveness of the velocity tracking. It was set to zero if the feedback velocity was higher than the targeted velocity to ensure that the shared-control did not accelerate the system. The counter lean resulted from this tracking loop offset the COM shift caused by the rider and re-established a dynamically stable position. It also provided distinct haptic feedback to the rider and helped convey the shared-control's intention to the rider.

Finally, to mitigate the effects of noisy sensor measurements and rapid changes in the direction of motion, a low-pass filter was integrated into the system. This low-pass filter smoothed out jitteriness by penalizing the difference between v^{comp} and the commanded velocity from the last time step v^{old}, resulting in a more fluid and reliable movement.

$$v^{cmd} = v^{comp} + \epsilon(v^{old} - v^{comp}) \tag{4.10}$$

Note that, as PAPF calculated the repulsive forces for x and y axes separately, the above equations were repeated for each axis. The detailed implementation of PAPF can be found in APPENDIX C.

In situations where PURE detected an imminent collision risk despite its efforts to slow down the rider, an audio alarm was also activated to explicitly alert the rider to the danger. The volume of the alarm increased as the obstacle got closer, providing a clear indication of the proximity of the collision. During the human-robot interaction evaluation, the maximum volume of the alarm was adjusted based on the subject's preference to minimize frustration.

4.5 ROBOTIC SYSTEM EVALUATION

A. Experiment Setup

The robotic system evaluation involved two experiments to assess the deceleration performance of the ballbot and its shared-control system.

In the first evaluation, a deceleration test with a static obstacle was performed. The ballbot was loaded with different payload weights (ranging from 13.6 to 54.4 kg) and was commanded to accelerate to various steady-state initial velocities (ranging from 0.5 to 1.4 m/s) towards a cardboard box obstacle. The shared-control system automatically issued deceleration commands to ensure the system's safety and prevent collision. The distance and time taken for PURE to come to a complete stop after the shared-control initiated the deceleration signal were recorded for each combination of payload weight and initial velocity. Three measurements were taken for each case, and the average values were reported.

The second evaluation assessed the performance of PURE and its shared-control system in scenarios where obstacles suddenly appeared within the shared-control's FOV. These scenarios mimicked real-life situations such as a car unexpectedly reversing from a driveway or a person abruptly opening a door in a narrow hallway. Even for human riders, these "sudden obstacle" situations are challenging to foresee. To evaluate the shared-control's response to sudden obstacles, the shared-control sensor readings were altered to trick the shared-control into perceiving that no obstacle was present. The test conditions from the first evaluation were repeated with this altered perception, and this alternation was removed once the ballbot reached a certain "offset" distance (e.g., 2.0 m, 1.5 m, 1.0 m, and 0.5 m) mimicking a sudden "appearance" of the obstacle at the given offset distance. The objective was to determine the performance boundary of PURE and shared-control in the presence of a sudden obstacle. Consequently, only worst-case scenarios (with 54.5 kg payload) were considered, involving maximum payload and high velocities.

B. Results

The results of the first experiment (Fig.4.7) suggested that the deceleration effort of the PAPF shared-control is proportional to the rider commanded velocity (as defined by Eq .4.8). Consequently, lower initial velocities resulted in lower deceleration efforts, leading to longer deceleration distances and times. Conversely, when PURE was traveling at higher velocities (e.g.,

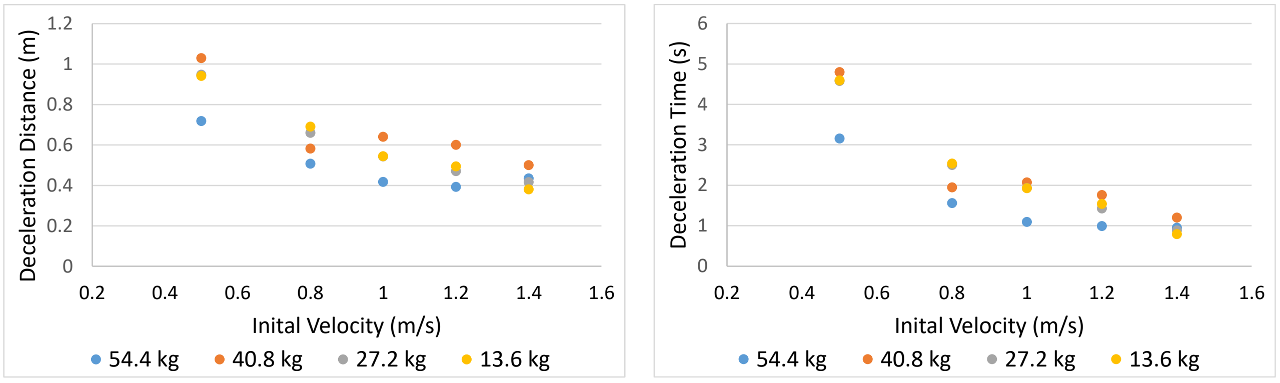


Fig. 4.7. Averaged deceleration distance (left) and time (right) for PURE to stop under different payloads and initial velocities.

1.4 m/s, similar to human jogging speed), the shared-control exerted significant deceleration effort to bring the ballbot to a stop before a potential collision.

The results of the second experiment (Table 4.1) suggested that when traveling at 1.0 m/s, PURE was able to effectively stop itself and prevent a collision if an obstacle suddenly appeared 0.5 m away. However, at a higher velocity (1.4 m/s), PURE could only avoid a collision if the obstacle was detected at least 2 m away.

C. Discussion

PAPF shared-control approach was designed such that the deceleration effort was proportional to the rider commanded velocity (Eq.4.8). This behavior provided riders with more freedom when traveling slowly and cautiously, while stepping in strongly to ensure system safety during critical collision risks. The results in Fig.4.7 highlighted the adaptability of the shared-control system in adjusting the deceleration effort based on the velocity of the platform, contributing to a safer and more forgiving user experience. Fig.4.7 also suggested that heavier payloads might lead to a shorter deceleration distance and time especially when traveling at lower velocities. This effect diminished as the velocity increased to 1.4 m/s.

TABLE 4.1
SHARED-CONTROL SUDDEN OBSTACLE TEST WITH 54.5 KG PAYLOAD

Offset Dist (m) / Initial vel (m/s)	2	1.5	1.0	0.5
1.0	success	success	success	0.08
1.2	success	0.18	0.25	failed
1.4	0.10	failed	failed	failed

Numbers in green indicate the distances to the obstacle (m) when PURE stopped.

The results of the sudden obstacle appearance evaluation (Table 4.1) suggested that it was challenging to safeguard PURE from sudden obstacles when the device was traveling with high initial velocity (1.4 m/s). It is important to note that PURE's deceleration required the device to lean in the opposite direction of travel, and the deceleration effort was proportional to this lean angle. However, the presence of a physical protection ring on PURE imposed a constraint on the maximum degree to which the device could lean (approximately 13°). To completely stop PURE traveling at 1.4 m/s within 1.5 m would require a deceleration effort (lean angle) exceeding this limit. Therefore, the true deceleration performance of PURE's drivetrain cannot be solely inferred from this test alone.

In the context of dynamic obstacles, the reactive shared-control approach proposed in this study assumed that the fast sampling rate and control loop rate (30 Hz for primary sensors and 400 Hz for the control loop) enabled the perception of most indoor moving objects as "static" from the shared-control's perspective. This assumption relied on the shared-control's ability to efficiently capture and respond to the dynamics of the environment. However, for more sophisticated behaviors that involve predicting the movements of objects and taking proactive actions, it is advisable to implement a deliberative shared-control approach.

4.6 HUMAN-ROBOT INTERACTION EXPERIMENT SETUP

A. Subject Demographics

Twenty participants were tested in the study, including 10 manual wheelchair users (mWCUs) and 10 able-bodied individuals (ABUs) (Table 4. 2). The inclusion criteria were: 18 to 35 years old, body weight less than 70 kg (due to limitations of the current prototype) and no visual impairment. For mWCUs, an additional inclusion criterion required minimal trunk control with sensation, specifically the ability to perceive pressure at least up to the level of the Xyphoid process. The study was approved by the University of Illinois Institutional Review Board (IRB #23664) and informed consents were obtained from all participants.

TABLE 4.2

SUBJECT DEMOGRAPHICS AND ANTHROPOMETRIC AVERAGE (STANDARD ERROR) DATA

		ABU	mWCU
Number [Male: Female]		10 [5:5]	10 [4:6]
Age (years)		23.6 (1.1)	27.4 (1.6)
Height (m)		1.68 (0.02)	1.62 (0.03)
Weight (kg)		58.1 (2.3)	51.6 (2.6)
Years of using mWC (yrs)		NA	19.4 (2.55)
Torso Range of Motion (˚)	frontal	63.1 (4.2)	46.6 (4.5)
	sagittal	58.6 (3.3)	41.1 (3.5)
	transverse	91.3 (4.2)	64.5 (9.5)
Maximum Torso/FSS Moment Range (Nm)	frontal	107.1 (8.1)	82.1 (12.3)
	sagittal	113.1 (7.7)	85.6 (11.3)

TABLE 4.3

BACKGROUND SUMMARY OF TESTED MANUAL WHEELCHAIR USERS

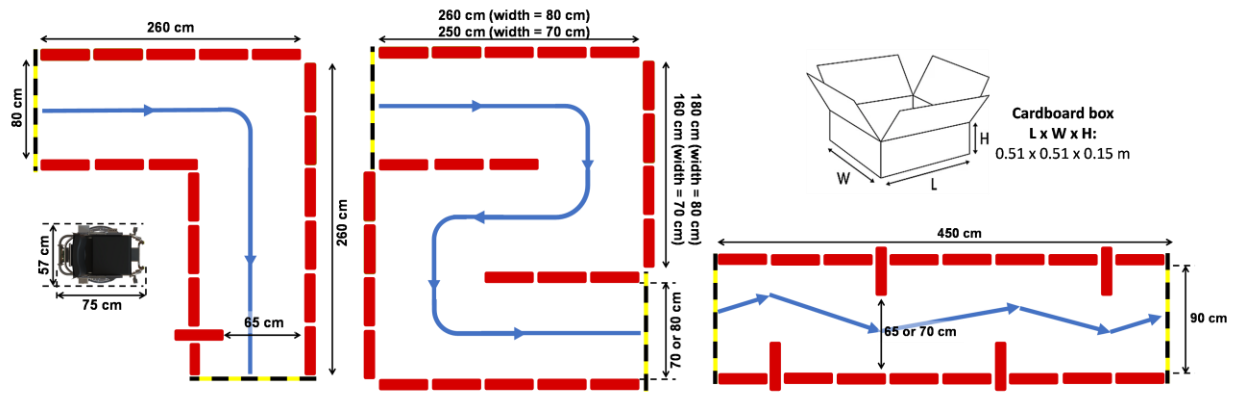
		Number
Injury	Amputation	1
	Arthrogryposis	1
	Cerebral Palsy	1
	Congenital	1
	Polio	1
	Spina Bifida	1
	Spinal Cord Injury	4
Spinal Condition	Spinal Fusion	3
	Scoliosis	6
	Kyphosis	3
mWC Athlete Rating	T34	1
	T53	2
	T54	5
	N/A	2

Torso range of motion (ROM) in the frontal θ_x^{ROM}, sagittal θ_y^{ROM}, and transverse θ_z^{ROM} planes were recorded using the wearable IMU attached to the participant's upper chest while seated on PURE. Participants were asked to perform a series of motions while PURE was parked stationary. These motions included leaning in the forward-backward, left-right, and diagonal directions to their comfortable extent. These ROM measurements capture the maximum leaning and twisting capabilities of the participants' torsos, providing valuable insights into the range of torso movements that PURE can accommodate. The maximum torso moments as measured by the FSS during these ROM movements in the frontal M_x^{usr} and sagittal M_y^{usr} planes were also recorded. Background information on the mWCU participants' injury types, spinal conditions, and mWC athlete rating scores [165] were recorded in Table 4.3. Four mWCUs presented with multiple spinal conditions, i.e., [*Scoliosis* and *Kyphosis*], [*Spinal Fusion* and *Scoliosis*], [*Spinal Fusion* and *Kyphosis*], and [*Spinal Fusion*, *Scoliosis*, and *Kyphosis*]. The successful completion of the HRI test by all mWCUs suggested that PURE and its associated shared-control could accommodate a wide range of mWCUs with diverse backgrounds and conditions.

B. Test Courses

To evaluate and quantify the efficacy of the proposed shared-control paradigm, a HRI study was conducted. The study comprised of one training course (Fig.4.8a) and two test courses (Fig.4.8b and Fig.4.8c). The training course was designed to introduce and familiarize participants with PURE and the shared-control operation. This allowed participants to become acquainted with the controls and prepare for the subsequent test courses, where the performance of the shared-control system was evaluated. To ensure fairness and prevent participants from memorizing specific sections, the training course was intentionally designed differently from the test courses.

The testing phase consisted of two distinct test courses, namely the S-Turn (ST, Fig.4.8b and Fig.4.9b) and Zigzag (ZZ, Fig.4.8c and Fig.4.9c). These courses were designed to simulate various challenging indoor navigation scenarios, including narrow hallways, tight turns, and static obstacles. To accommodate participants with different navigation skills, each test course had two width settings. For the S-Turn course, the track widths were set to 70 cm and 80 cm. In the case of the Zigzag course, the track width remained fixed at 90 cm, while the narrowing sections had widths of 65 cm and 70 cm. The width of PURE was approximately 60 cm, while the minimum width for wheelchair passage was 81.5 cm (32 in) at a point and 91.5 cm (36 in) continuously according to the Uniform Federal



a) Training course b) S-Turn test course (ST) c) Zigzag test course (ZZ)

Fig. 4.8. The bird's eye view schematics for training course and test courses setup. The dimensions of PURE are included for comparison.

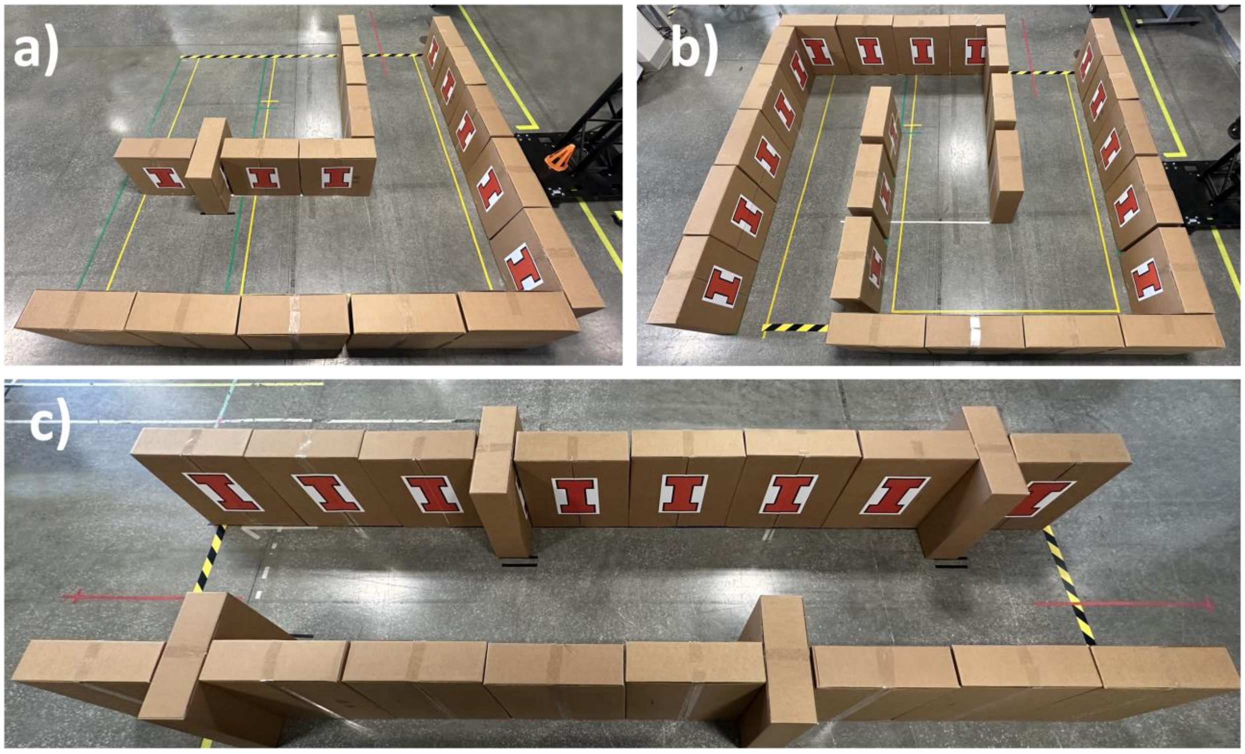


Fig. 4.9. Pictures of the physical a) training course, b) S-Turn test course, c) and Zigzag test course.

Accessibility Standards [166]. Therefore, these test courses can be considered highly challenging due to the narrow passages and constrained spaces that they present. The selection of these width settings was determined through a pilot test involving three ABUs and two mWCUs (who were excluded from the HRI evaluation). The aim of the width selection was to strike a balance between difficulty and minimizing frustration for the participants. It should be emphasized that, to the best of the authors' knowledge, these width settings represent the most challenging configurations compared to similar existing studies. Typically, the width range in such studies spans from 0.81 m to 3 m [11], [34], [54], [55], [57], [58], [158], making the chosen settings notably more demanding.

The walls of both the training and testing courses were constructed using cardboard boxes (0.51 × 0.51 × 0.15 m^3). The cardboard walls provided a lightweight and discrete structure, allowing for intuitive collision assessment during the evaluation process. During testing, collision incidents were categorized based on the impact made on the cardboard walls. If a participant lightly collided with a box but did not dislocate it, it was classified as a "touch" collision. On the other hand, if the collision resulted in the box being pushed out of its original position, it was recorded as a "move" collision. In cases where a participant caused significant disruption to the test course, rendering it unable to continue, the trial was marked as a "failure". Only "primary" collisions were counted and recorded, i.e., each collision between the robot and a specific obstacle was considered only once, specifically the first instance of collision. Adopting this protocol ensured a consistent and standardized approach for collision calculation. The subjects were instructed to treat the collided box as if it were no longer present after a collision occurred.

C. Training and Testing

During the training phase, participants engaged in movement within an open space and the designated training course to practice navigation using PURE. They were given the opportunity to adjust TES drive sensitivities according to their preferences for different directions of movement. Once participants felt comfortable riding PURE, the shared-control mode was introduced. Before proceeding, participants received a briefing on the operational principles of the shared-control system.

They were then instructed to repeat the training process with the shared-control mode enabled. The adjustable parameters of the shared-control, such as the resistive force and alarm volume, were tuned based on each participant's preference. The training phase was considered completed once participants expressed confidence in maneuvering PURE in all directions, both with and without the shared-control system.

During the testing phase, participants were instructed to ride PURE from the starting line to the finish line, marked by tape on the floor (Fig. 4.9). Each participant completed three trials for each test course, with and without the shared-control, and for two different widths of the test course. In total, each participant completed 24 trials (2 test courses × 2 control schemes × 2 widths × 3 trials per condition). To mitigate bias introduced by learning effect and fatigue, the testing order of the test courses and control schemes was randomized among all participants. For each test course, participants always completed the trials for the wider width first, followed by the narrower width. This sequencing was designed to minimize any anxiety participants may have experienced when starting with a narrow test course. They were instructed to navigate the course at their comfortable speeds while avoiding collisions. During each trial, the number of collisions, completion time, and video footage were recorded. All data streams within the robot were collected using ROS.

D. Metrics and Data Analysis

Inspired by previous studies [167], [168], a Collision Index (CI) was developed by assigning weights to different types of collisions, namely "touch," "move," and "failure." These weights reflect the severity of the collision and the potential risk of structural damage. The CI is calculated using the following equation:

$$Collision\ Index = 1 \times \#\ of\ Touches\ + 3 \times \#\ of\ Moves\ + 9 \times \#\ of\ Failures \tag{4.11}$$

Note that in the previous studies [167], [168], the performance index included the completion time as a factor. However, in this study, the completion time is evaluated separately from the CI because the numerical values of the completion time were significantly larger than the collision scores. This difference in scale would have caused the completion time to dominate the overall index score, making it less applicable to this specific study.

The study incorporated three sets of metrics to evaluate various aspects of the system's performance:

1) Objective performance metrics: These included the averaged Collision Index (CI), averaged completion time (T^c), averaged number of touch collisions, averaged number of move collisions, and averaged number of failures from three trials per test condition. The objective metrics provided quantitative measures of performance during the tests.

2) NASA Task Load Index (TLX): The NASA TLX survey [169], a widely adopted tool in wheelchair shared-control studies [34], [41], [48], [170], was utilized to assess the perceived workload and relative operation effort with and without shared-control. The survey included six score categories: mental demand, physical demand, temporal demand, effort, performance, and frustration. These scores were collected by having the subjects subjectively rate each shared-control mode. The TLX scores ranged from 0 to 100 points in 5-point increments [169]. A lower score indicated lower operational effort, except for the performance category. After completing each test course, participants were given a NASA TLX survey to evaluate the performance of PURE with and without the shared-control. The final TLX scores for each subject were computed by taking the average of the two TLX surveys, similar to the practice in [34].

3) Post-study questionnaire: A post-study questionnaire was designed to collect feedback and

preferences regarding various aspects of the shared-control design. The questionnaire utilized a 4-point Likert scale (i.e., “Very Little”, “Little”, “Much”, “Very Much”) to gather responses and insights. Participants were also asked to indicate their preferred shared-control paradigm for each of the testing courses, offering valuable insights into the scenarios in which the shared-control system would be most beneficial. A copy of this questionnaire can be found in APPENDIX D. Participants were asked to complete this post-study questionnaire once all their trials were finished.

To examine the impact of using shared-control, two repeated-measures multivariate analysis of variance (RM-MANOVA) tests and one one-way MANOVA test were conducted using IBM SPSS (SPSS Statistics Version 28.0.1, IBM Corp, USA). These tests were performed to address the research questions regarding the potential effects of shared-control on performance, operational effort, and user feedback, respectively. In the first test, the performance was assessed using the objective metrics, average CI, average T^c, average touch collisions, and average move collisions. The averaged number of failures was not included in the RM-MANOVA because there were only two occurrences during the study (for two ABU, both in the S-Turn Narrow test course without using shared-control) and these were considered as outliers. The operational effort was evaluated using the NASA TLX scores, while the user feedback was collected with the custom questionnaire. The first RM-MANOVA test included three independent variables: Shared-Control mode (No Shared-Control vs. Shared-Control), test courses (S-Turn Wide (STW), S-Turn Narrow (STN), Zigzag Wide (ZZW), and Zigzag Narrow (ZZN)), and user group (ABU, mWCU). The second RM-MANOVA test evaluated independent variables of shared-control mode and user group. The one-way MANOVA test tried to explore the potential preference differences between subject groups in post-study questionnaire results. Follow-up univariate ANOVA tests were conducted if RM-MANOVA tests reached statistical significance. A significance level of $\alpha = 0.05$ was used for all analyses, except for interaction effects which were modified for interactions of S-C mode and course ($\alpha = 0.05 / 8 = 0.006$) [171].

E. Results and Discussion

1. Objective Performance Metrics

The results of the two RM-MANOVA tests found statistically significant differences in performance and operational effort metrics due to the use of shared-control and test course (Fig. 4.10-4.11, 4.13-4.14, Tables 4.IV). The performance RM-MANOVA test results revealed statistically significant differences in the performance metrics (excluding average number of failures due to its rarity) based on the test course settings ($p < 0.001$), use of shared-control ($p < 0.001$), and interaction between shared-control and test course setting ($p = 0.005$) (Table 4.4). There were no differences in metrics between user groups ($p > 0.05$). Follow-up univariate ANOVA tests found that test course settings had a significant impact on average Completion Time (T^c), the

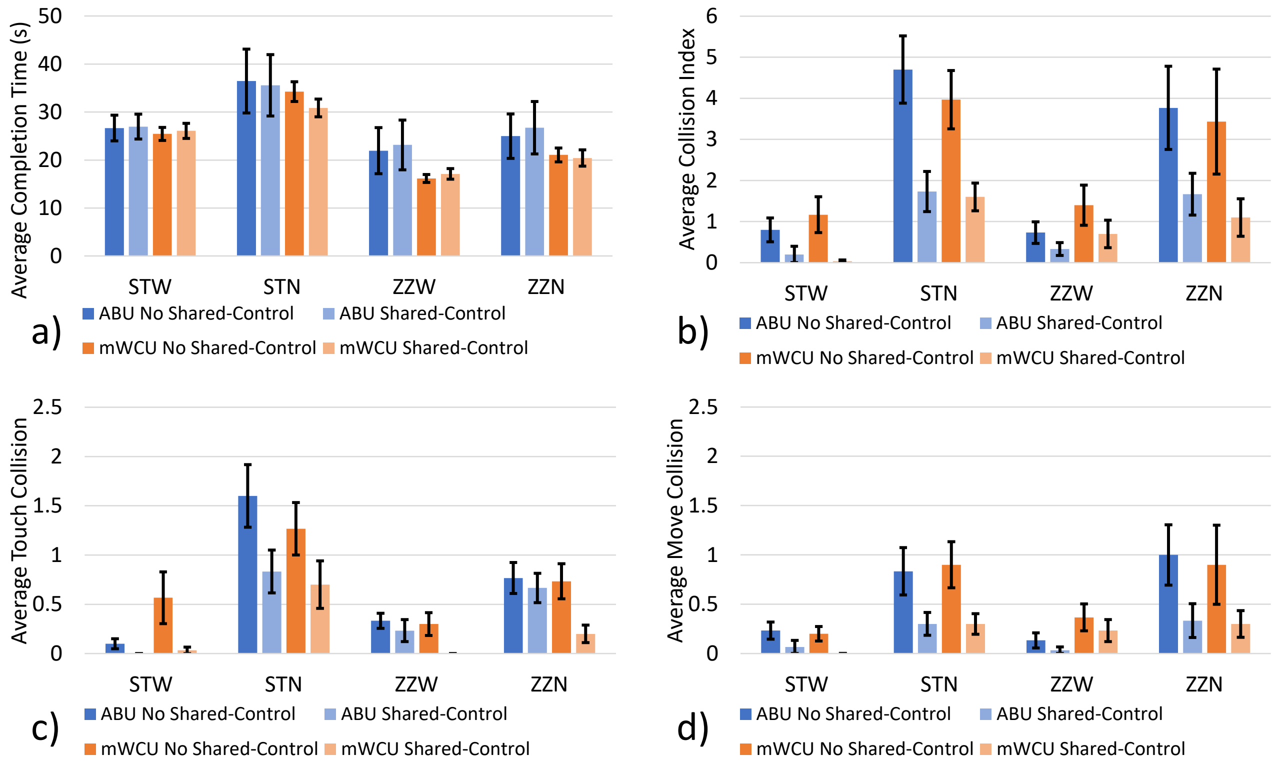


Fig. 4.10. Performance metrics by subject group and shared-control mode for each test course. a) Average Completion Time. b) Average Collision Index. c) Average Touch Collision. d) Average Move Collision. The error bars represent Standard Errors.

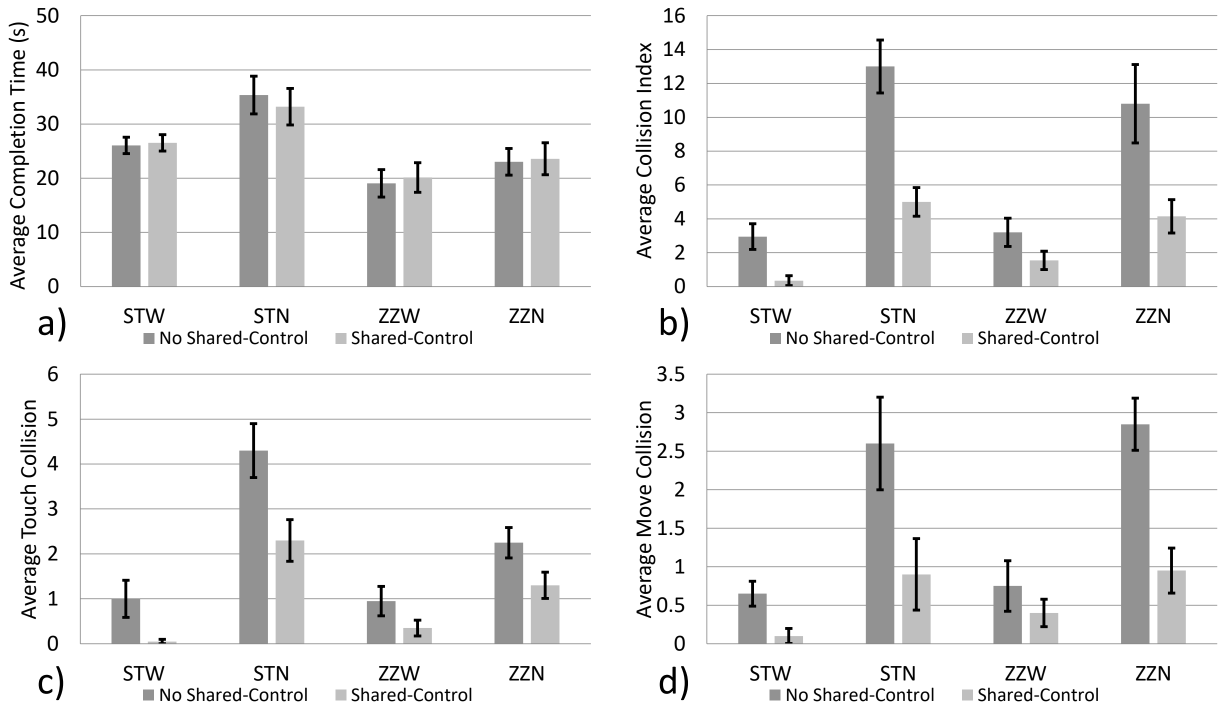


Fig. 4.11. Performance metrics by shared-control mode for each test course. a) Average Completion. b) Average Collision Index. c) Average Touch Collision. d) Average Move Collision. The error bars represent Standard Errors.

average touch collisions, the average move collisions, and average Collision Index (CI) (all p values < 0.001). These align with observations that changes in the test course width affected the difficulty and subsequently impacted the performance metrics (Fig.4.10 and 4.11). For example, the S-Turn Narrow course consistently took the longest (34.3 s) and had the most average touch collisions (1.1), while the Zigzag Wide course was completed the fastest (19.6 s). These results suggest that the width settings had a significant impact on completion time and collision-related performance. The use of shared-control affected the average touch collisions ($p = 0.002$), the average move collisions ($p < 0.001$), and CI ($p < 0.001$). According to Table 4.4 and Fig. 4.11b, on average, the addition of shared-control helped to reduce the CI by over 50% across all test courses. However, it did not have a significant impact on T^c ($p = 0.99$), suggesting that shared-control did not affect travel speed significantly (Table 4.4). This can be confirmed by Fig.4.11a where the presence of shared-control did not change the average T^c to a noticeable level. Lastly, there were interaction effects between the test course and shared-control mode on T^c (p = 0.027), CI (p = 0.008), and the move collision (p = 0.034), suggesting that shared-control may be particularly beneficial in certain cluttered environment settings (Fig.4.11).

TABLE 4.4
HUMAN-ROBOT INTERACTION TEST OBJECTIVE METRICS

	Shared-Control Mode		Test Course			
	No S-C (a)	S-C (b)	STW (c)	STN (d)	ZZW (e)	ZZN (f)
Avg Completion Time (s)	25.9 (2.5)	25.9 (2.6)	26.3 (1.5)^{de}	34.3 (3.7)^{cef}	19.6 (2.7)^{cdf}	23.3 (2.8)^{de}
Avg Touch Collision	0.7 (0.1)^{b}	0.3 (0.1)^{a}	0.2 (0.1)^{df}	1.1 (0.1)^{cef}	0.2 (0.1)^{df}	0.6 (0.1)^{cde}
Avg Move Collision	0.6 (0.1)^{b}	0.2 (0.1)^{a}	0.1 (0.0)^{df}	0.6 (0.1)^{ce}	0.2 (0.1)^{df}	0.6 (0.2)^{ce}
Collision Index	2.5 (0.4)^{b}	0.9 (0.2)^{a}	0.6 (0.2)^{df}	3 (0.4)^{ce}	0.8 (0.2)^{df}	2.5 (0.5)^{ce}

S-C: Shared Control; Values represent mean (standard error).
Note: A superscript letter denotes significant difference from indicated condition ($p < 0.05$).

The results of the performance RM-MANOVA (Table 4.4 and Fig.4.11) strongly suggest that the proposed shared-control system effectively mitigated collision risks without causing unnecessary slowdowns. These findings suggest that the proposed shared-control strikes a favorable trade-off between maintaining a reasonable travel speed for the user and effectively managing collision risk. It is worth noting that this trade-off was not always achieved by other shared-control systems, as highlighted by previous research [172]. Specifically, shared-control systems may exhibit a significant reduction in speed when the wheelchair approaches an obstacle.

To gain a deeper understanding of how the proposed shared-control system assisted the rider during the experiment, Fig. 4.12 provides a qualitative example from one of the trials of subject 14. In these particular trials, subject 14 completed the ST test course without shared-control in 33.8 s and with the shared-control in 24 s. For the ZZ test course, the completion time were 19.8 s without the shared-control and 19.3 s with the shared-control. The CI for the ST test course without and with the shared-control were 8 and 1 respectively. For the ZZ test course, the CI were 16 and 0 without and with the shared-control. Fig. 4.12 illustrates the trajectories taken by a mWCU for

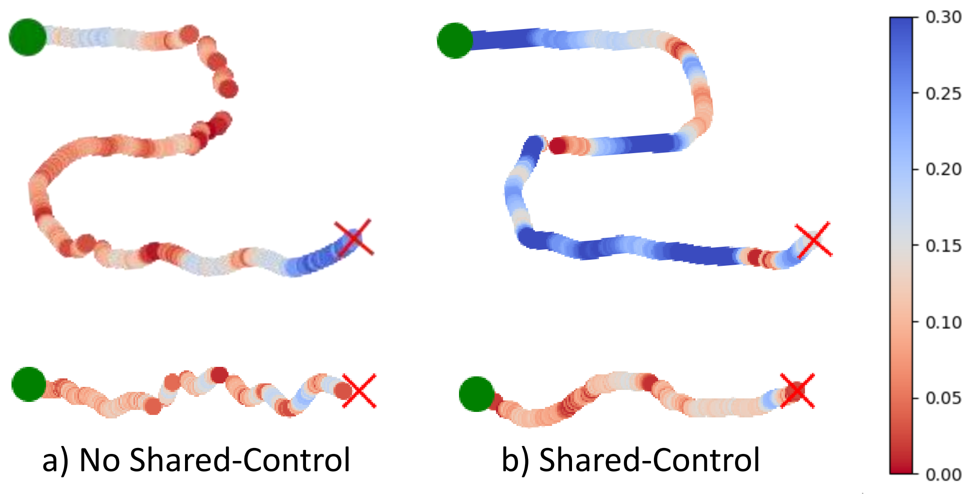


Fig. 4.12. Example trajectories (ST: top, ZZ: bottom) taken by subject 14. Color indicates velocity (m/s) in the forward direction. The "broken" segment indicates PURE drove backward in those instances. The **green** circles represent starting positions, and the **red** crosses are where the data collection ended.

both the ST and ZZ test courses. The mWCU of interest had a complex spinal injury condition, including Spinal Fusion, Scoliosis, and Kyphosis, making these data particularly insightful in assessing the collaboration between the rider and the shared-control system. The trajectories are color-coded based on the velocity of the ballbot in the forward direction. The trajectories demonstrate that with the assistance of the shared-control system, the rider was able to navigate the ST and ZZ test courses with much smoother paths. Prior to using the shared-control system, the rider exhibited a significant amount of jittering and back-and-forth motion while traveling through the ST course. The overall velocity during this period was below 0.1 m/s. However, when riding with the shared-control system, the rider's trajectories became smoother, with minimal jittering observed. Moreover, the overall velocity in the ST course increased, reaching above 0.2 m/s. The observed improvement in trajectory smoothness and increased velocity with the use of the shared-control system suggests that shared-control effectively reduces the amount of motion corrections required by the ballbot during tight turns by providing deceleration assistance. The rider no longer needs to rely on leaning back and forth to fine-tune the traveling speed, but rather, they can simply lean forward and allow the shared-control system to slow down the platform when necessary. This aspect of the shared-control system is particularly beneficial for individuals who have limited torso control, as it alleviates the need for precise upper body motion adjustments. Instead, the rider can focus on maintaining a stable and comfortable posture while relying on the shared-control system to handle the deceleration and speed control aspects.

2. NASA TLX Score

According to the operational load RM-MANOVA, shared-control had significant decrease in mental demand (p = 0.002), physical demand (p = 0.022), effort (p < 0.001), and frustration (p = 0.008), and increase in performance scores (p < 0.001), but there were no differences in metrics between user groups (p > 0.05). (Fig. 4.13 and Fig. 4.14). There was no significant effect on temporal demand (p = 0.06). These results suggest the decrease in operational effort and the increase in performance were clearly perceived by subjects while using the shared-controller. Although the RM-MANOVA found no statistically significant difference due to user group (p >

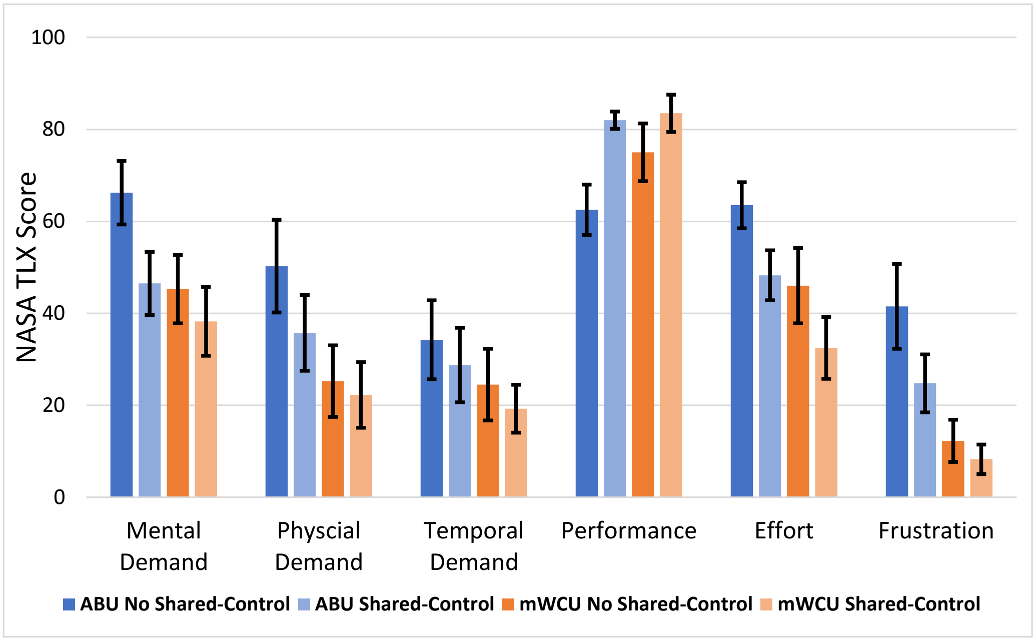


Fig. 4.13. NASA TLX survey results by subject group and shared-control mode. The error bars represent Standard Errors.

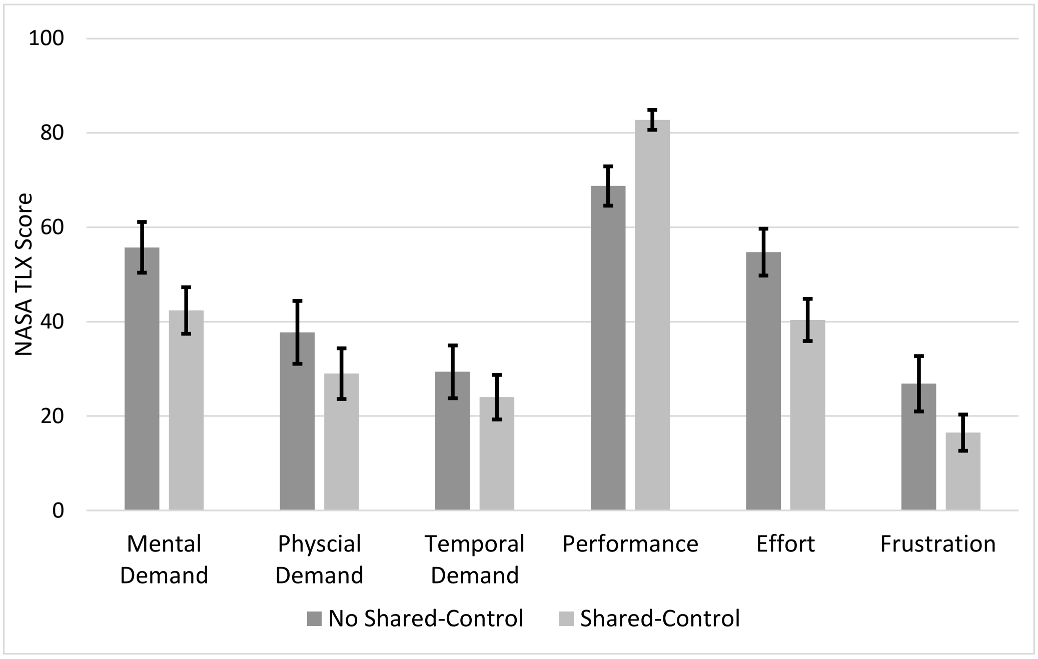


Fig. 4.14. NASA TLX survey results by shared-control mode. The error bars represent Standard Errors.

0.05), it was observed that mWCUs, on average, had a tendency to select lower scores particularly for mental demand, physical demand, effort, and frustration (Fig. 4.13).

As this study focused on comparing the difference between No Shared-Control and Shared-Control, the TLX scores in the No Shared-Control mode for a particular individual can serve as a baseline for that individual and help “normalize” the TLX scores across different subjects. To provide a more intuitive understanding of the impact of shared-control on TLX scores, it may be useful to examine the TLX percent difference for each individual subject on each category using the formula:

$$TLX^{\%diff} = 100\% * \frac{TLX^{SC} - TLX^{No\ SC}}{TLX^{No\ SC}} \tag{4.12}$$

Fig. 4.15 reports the average TLX percent differences for ABU and mWCU groups. An additional MANOVA test was performed on these TLX percent differences, and no significant difference was found due to subject group (p = 0.68). Thus, similar results were found for either group. The addition of shared-control reduced the mental demand (-22.05% averaged across

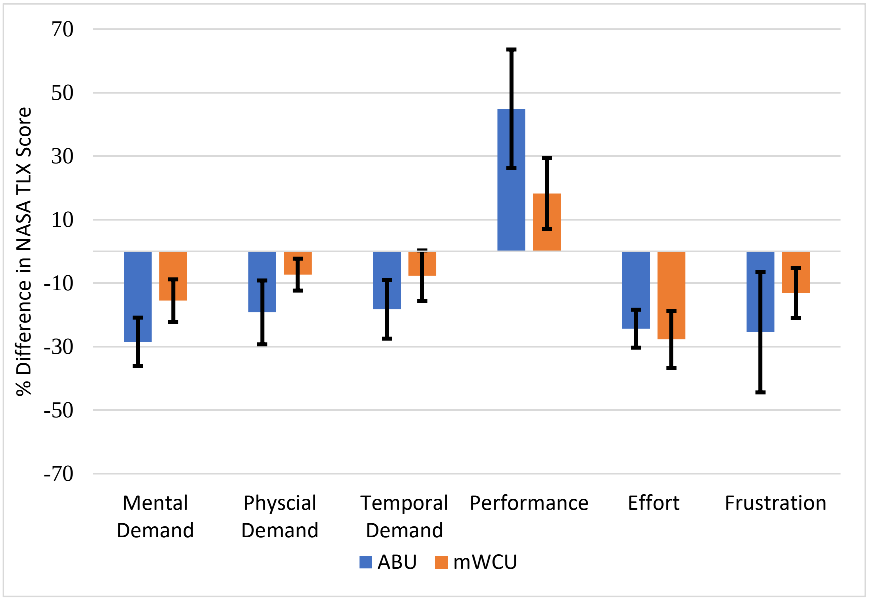


Fig. 4.15. NASA TLX % difference with and without shared-control. The error bars represent Standard Errors. Although the MANOVA test found no significant difference due to subject group, data are presented by subject group for full presentation of results.

subject groups), physical demand (-13.26%), temporal demand (-13.0%), effort (-26.05%), and frustration (-19.3%). Moreover, the inclusion of shared-control also resulted in a substantial increase in perceived performance, with a 31.55% improvement reported by all. These results suggest that the shared-control system enhanced the riders' confidence and satisfaction in controlling the PURE platform , contributing to an overall improvement in their perceived performance. The results from the operational load RM-MANOVA also support these observations, indicating a statistically significant effect of the shared-control on mental demand, effort, and performance scores (Fig. 4.14). The results highlight the positive impact of shared-control on reducing the cognitive workload of the riders and enhancing their perceived performance.

3. Post-study Questionnaire

The third research question aimed to explore participants' preferences regarding various aspects of the shared-control design. The post-study questionnaire was specifically designed to collect this information. Participants were asked to rate shared-control on ten categories using a 4-point scale (Fig.4.16). The one-way MANOVA test found that there were no significance differences in these questionnaire scores between subject groups (p = 0.304). Fig.4.16 suggests that all participants found the shared-control to be intuitive (average across all participants: 3.55), natural (3.15), and safe (3.5) to use. Although the MANOVA test did not reveal any significant

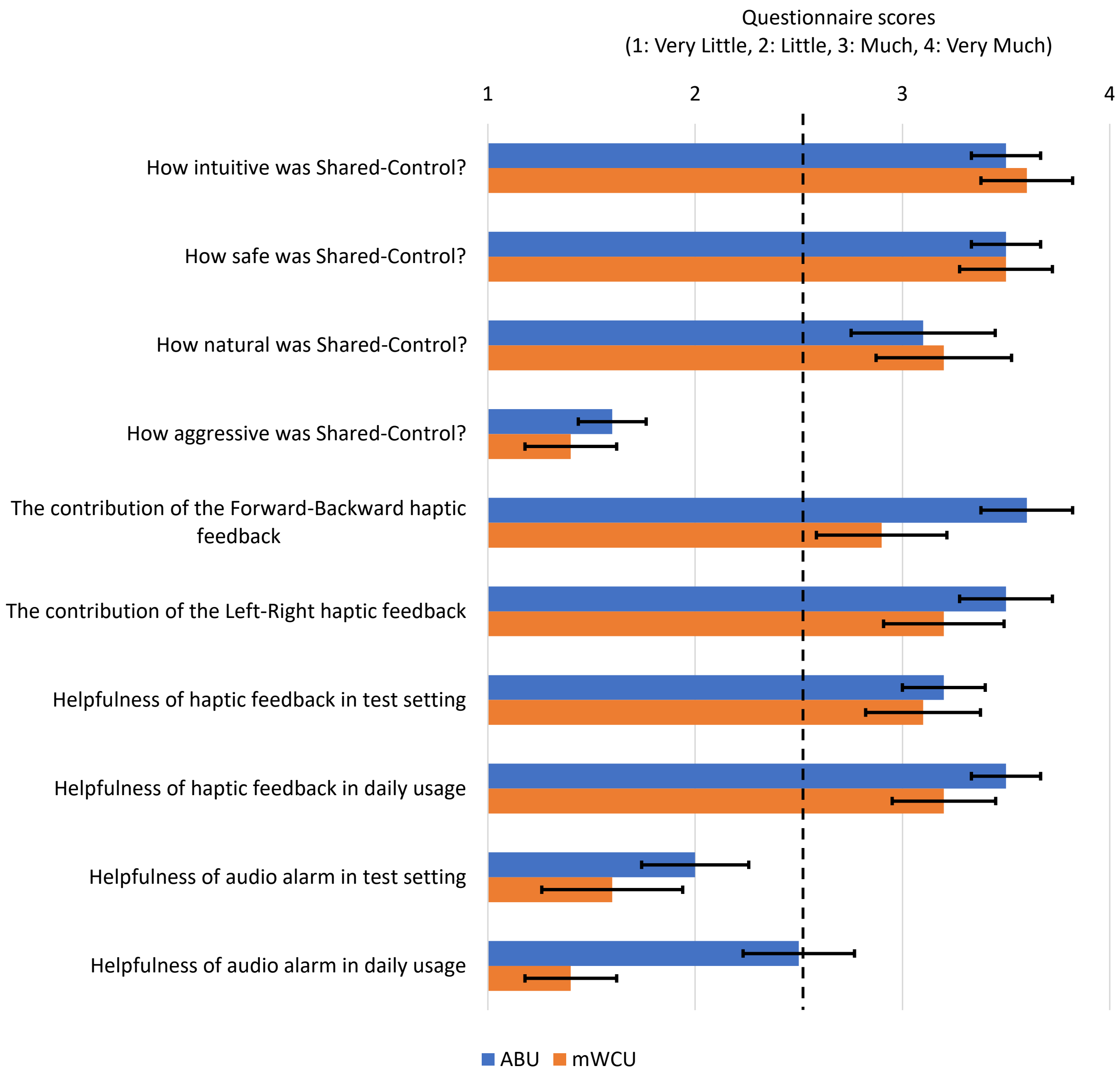


Fig. 4.16. Post-Study Questionnaire Scores. The error bars represent Standard Errors. The dashed line represents a neutral level, a score higher than this line indicates agreement. Although the MANOVA test found no significant difference due to subject group, data are presented by subject group for full presentation of results.

differences between subject groups, it is worth noting a perceptible trend in the responses of WCUs and ABUs. Neither group regarded the current shared-control design as aggressive (1.5 on the scale), with a noticeable inclination among mWCU participants to perceive it as even less aggressive than their ABU counterparts. ABU participants tended to attribute the haptic feedback's contribution more to the Forward-Backward direction than mWCU participants did. However, both groups generally reached a consensus regarding the contribution of haptic feedback in the Left-Right direction. Regarding the perceived helpfulness of the audio alarm in both the test setting and daily use, both subject groups showed a preference for haptic feedback over the audio alarm. Specifically, mWCU participants rated the helpfulness of the audio alarm as between "Little" and "Very Little" in the test setting, and even lower for daily usage.

The post-study questionnaire results shed light on the participants' perceptions and preferences regarding the shared-control system (Fig. 4.16). On average, all participants expressed positive feedback, considering the shared-control to be intuitive, natural, and safe to use. This suggests that the shared-control system was well-received and effectively facilitated the control of the PURE platform by both user groups. Notably, most participants found the haptic feedback to be particularly helpful in both the experimental setting and potential daily usage scenarios in general. Most participants showed a preference for implicit communication methods, such as haptic feedback, over explicit communication methods like the audio alarm. Meanwhile, mWCU participants had a stronger preference for receiving feedback through tactile sensations rather than auditory cues, especially during daily usage. Understanding these preferences is crucial for designing shared-control systems that can cater to the specific needs and preferences of the targeted user group, ensuring a more personalized and user-centric approach.

In the second part of the post-study questionnaire, participants were asked to subjectively vote for their most preferred shared-control paradigm for each test course (2 test courses × 20 subjects = 40 votes in total). The results differ by subject groups and for different test courses (Fig .4.17). Over 50% of ABUs preferred "Full Shared-Control (haptic + alarm)" for either test course, whereas 50% of mWCUs opted for "No Shared-Control" on the less taxing ZZ course and 50% favored "Haptic Feedback Only" during the more challenging ST course. According to the participants who preferred the "No Shared-Control" option, their main reason for this preference was that they desired a higher level of responsiveness or sensitivity from the device, specifically in the ZZ test course. This suggests that these participants felt that the shared-control system, while providing assistance and safety measures, may have introduced a certain level of control or response delay that affected their desired level of maneuverability in the ZZ test course. The preference of ABUs for the "Full Shared-Control" option suggests that they valued the additional safety feedback provided by the haptic force and audio alarm. As none of them had prior wheelchair driving experience, these features likely contribute to a sense of assurance and increased situational awareness for the ABUs, allowing them to navigate the test courses with

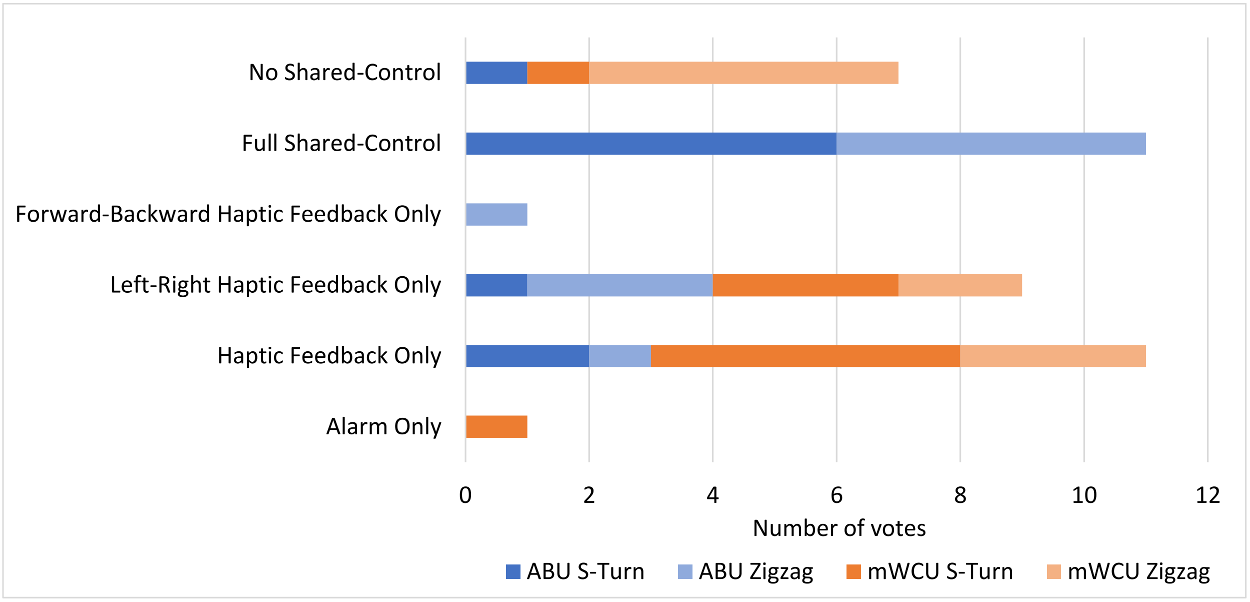


Fig. 4.17. Most preferred shared-control paradigm for each test course and subject group.

confidence. The participants' preference for haptic feedback over audio alarm is consistent with the Post-Study Questionnaire findings (Fig .4.16), further highlighting their preferences toward implicit communication methods. Specifically, mWCUs expressed a strong preference against having any audio alarm. This aversion to audio alarms among mWCUs might be attributed to their tendency to avoid drawing attention to themselves.

Overall, these results highlight the diverse preferences of participants when it comes to shared-control paradigms, with varying preferences for the presence of shared-control, type of feedback (haptic vs. audio), and the direction of haptic feedback. It is difficult to conclude which shared-control paradigm is the clear winner. This observation aligns with similar findings reported in [173]. It is evident that individuals have diverse preferences when it comes to shared-control designs, making it difficult to identify a one-size-fits-all solution. The studies in [170], [173] have emphasized the importance of offering multiple control options to the users to accommodate their individual needs in different scenarios.

4.7 LIMITATIONS AND FUTURE WORK

Despite the positive outcomes observed in terms of improved performance and reduced operational effort in the HRI study, this research is not without limitations and calls for further improvements. First, in terms of the hardware design, the single-beam LiDAR used by the current prototype cannot capture obstacle height information in the posterior direction. Therefore, it is preferable to upgrade these sensors to multi-beam LiDAR sensors or RGB-D cameras in the future. Similarly, the current prototype used a single-channel speaker. Therefore, the direction from which the alarm is heard does not indicate the direction of the imminent collision. Future study could look into a multi-channel alarm system and compare its efficacy in a similar study. Second, the static nature of the proposed test courses restricted their representation of dynamic and high-speed navigation challenges. Future research should encompass more comprehensive experiments to evaluate the performance of the proposed shared-control scheme in dynamic settings. Third, there is a need to explore a more analytically sound passive shared-control architecture and conduct a thorough analysis of its passivity property similar to the work in [174]. Lastly, the implementation of alternative reactive shared-control paradigms, such as DWA (Dynamic Window Approach) [55] and deliberative shared-control approaches[49]–[51], [53], on the PURE platform would provide benchmarking opportunities and deeper insights into the field of assistive mobility.

4.8 CONCLUSION

A shared-control approach utilizing the Passive Artificial Potential Field (PAPF) was introduced as a means to enable safe and intuitive navigation of the riding self-balancing device, PURE. This study included two evaluation experiments. 1) A robotic system evaluation that focused on assessing the deceleration performance of PURE and its shared-control system. 2) A human-robot interaction evaluation, which involved ABUs and mWCUs ridig PURE in two test courses that simulated challenging daily navigation scenarios, where the goal was to evaluate the performance improvement and operational effort reduction achieved using the shared-control. The HRI evaluation results suggested that the addition of shared-control led to an increase in performance metrics, as indicated by the reduction in the number of collisions. Second, the operational effort, as measured by the NASA TLX scores, demonstrated a significant decrease in effort with the use of shared-control. Third, the study revealed no statistically significant differences between ABUs and mWCUs in terms of completion time and collision-related performance. However, ABUs and mWCUs had distinct preferences towards specific shared-control designs. The insights gained from the study provide valuable guidance for the design and enhancement of shared-control systems, helping to improve safety, performance, and user experience in the field of assistive mobility devices.

CHAPTER 5

CONCLUSIONS

5.1. Contributions

This dissertation delved into the challenges of robot perception and safeguarding in the context of mobile robots, with a particular focus on the self-balancing mobile robot system known as the ballbot. The complexity and under-actuated dynamics of the ballbot, and its intended use as a riding mobility device, necessitated the development of advanced perception techniques and control systems. The dissertation centered on the creation of perception and shared-control systems applicable to a physical ballbot prototype, while adhering to size-weight-and-power constraints (SWaP-C), and generalizable to other mobile robots with similar characteristics.

Chapter 1 set the stage for the dissertation. It underscored the need for refined perception techniques and more effective control systems in the ballbot and ridable self-balancing devices. It described robot perception requirements and the design of the ballbot. Furthermore, the chapter also introduced the novel mobility device PURE and shed light on the existing efforts in assistive mobility devices utilizing vision-based sensor fusion and shared-control.

Chapter 2 centered on the successful development of a proprioception-based terrain classification method. This method harnessed a convolutional neural network (CNN) model and signals from standard on-board unmanned ground vehicle (UGV) sensors. Unlike existing methods, which perform optimally under strict motion conditions, such as driving straight at a constant speed, the proposed method managed to overcome these restrictions, achieving an accuracy of over 89% under various motion maneuvers. The fusion of the proprioception-based model with a vision-based CNN further enhanced the accuracy, showing strong performance (over 93%) across varying lighting conditions and motion maneuvers.

Chapter 3 focused on the development of a rapid, motion-robust RGB-D Drivable Area Segmentation (DAS) framework for indoor robot navigation. This framework integrated a ground plane segmentation algorithm with a self-supervised one-shot texture segmentation model known as MOSTS (Miniature One-Shot Texture Segmentation). The MOSTS model was trained using an innovative collage synthesis technique and did not require dense manually labeled training data. Furthermore, a validation dataset ISOD (Indoor Small Object Dataset) was built to evaluate MOSTS's capability for texture segmentation in various real-world scenarios, where it effectively identified small low-lying objects previously undetectable by depth measurements. MOSTS's performance was also compared with two state-of-the-art (SOTA) indoor semantic segmentation models, both quantitatively and qualitatively. MOSTS proved to have comparable accuracy while offering up to eight times faster inference speed in indoor drivable area segmentation. The proposed framework was deployable on low-power single-board computers and demonstrated adaptability for applications beyond drivable area segmentation.

Chapter 4 introduced the use of a Passive Artificial Potential Field (PAPF) shared-control approach for enabling safe and intuitive navigation of a rideable self-balancing device (a.k.a. PURE). A human-robot interaction test was conducted with Able-Body Users (ABUs) and manual Wheelchair Users (mWCUs) to evaluate the improvement in performance and reduction in operational effort achieved with shared-control. Experimental results and statistical analysis suggested that shared-control effectively reduced collisions without compromising travel speed. shared-control also reduced mental demand, physical demand, effort, and frustration, while improving perceived performance. No significant differences were found in preferred driving speeds and collision likelihood between ABUs and mWCUs. Post-study questionnaires revealed that both ABUs and mWCUs found the shared-control to be intuitive, natural, and safe, with minimal perceived aggressiveness. Haptic feedback, particularly in the left-right direction, was deemed important and favored over an audio alarm. The study underlined the preferences and perceptions of ABUs and mWCUs regarding the shared-control design, emphasizing the necessity of providing multiple control options to cater to individual needs in various scenarios.

In summary, this dissertation contributed to the field of mobile robots, tackling the challenges of perception and shared-control in the ballbot and offering innovative solutions. Each chapter focused on a distinct aspect, from developing a robust terrain classification method and a rapid drivable area segmentation framework to integrating the PURE mobility platform with shared-control and evaluating its performance through human-robot interaction tests. These findings enriched the current body of knowledge in mobile robotics and provided tangible implications for assistive mobility devices. This dissertation hopes to pave the groundwork for emerging research and technological advancements in the realm of robotics.

5.2. Limitations and Future Work

This dissertation sets the stage for various future research avenues, drawing on the areas of exploration identified in each chapter.

The terrain classification method developed in Chapter 2 opens several areas for future research. Although the current fusion method combines proprioceptive signals and vision information to make comprehensive terrain predictions, it is unclear if the model truly comprehends the underlying associations between these signals and the corresponding terrain images. More precise data association techniques, such as SLAM, could be considered for future data collection and development. Rather than simply associating the first image frame with a set of prior proprioceptive signals, SLAM can overlay past proprioceptive experiences (robot paths) onto a 3D map of the environment. This could facilitate the resampling of the most accurate terrain image associated with a particular set of proprioceptive signals from this 3D map.

In its current form, the method in Chapter 2 achieves image-level terrain classification, predicting an overall terrain class label for an entire image. However, with more precise data association, it may be possible to achieve pixel-level terrain segmentation. Resampling and rewinding proprioceptive signals could offer a simple yet effective approach to augmenting existing datasets and synthesizing new training data. Moreover, a feature importance analysis could be used to further reduce the required signals and model size.

Given its vehicle-specific nature, the current method also presents portability and scalability challenges. It does not support direct model transfer between vehicle configurations without retraining the proprioception network. Although one can use the vision network to generate pseudo labels for the proprioception network under ideal lighting conditions, the training process still consumes computational resources. Future studies might consider a model-based learning approach, where the deep neural network attempts to learn specific model parameters (system identification) and uses this model to infer frequency responses under different vehicle motions and road profiles. Such an approach could make the method more explainable and require minimal fine-tuning when adapted to a different vehicle model. Contact-based robot simulation is a good starting point for developing such a model.

Lastly, the current model can only predict a fixed number of possible terrain classes. A more practical setup might follow the approach of [131], which generates pseudo labels for novel terrain classes that show significant differences (distances) in the model's latent space. This direction could further expand the versatility and adaptability of the terrain classification method.

Building on the achievements of Chapter 3, future studies should seek to overcome several identified limitations and explore new applications of the proposed framework. Currently, the framework performs optimally when a single general floor texture type is present in the scene. However, it struggles with mixed texture types. One simple solution could involve pre-encoding a set of potential reference images into texture tokens and storing them in the robot's memory. The robot could then iterate through this token bank to determine the most effective reference token. This pre-encoding approach would simplify the model's Siamese encoder structure and further conserve computational resources during model deployment. Future research could also explore replacing the reference image with text tokens using the CLIP model [175].

The potential applications of the MOSTS model extend beyond refining DAS. As Chapter 3 demonstrated, MOSTS can effectively detect trash on outdoor terrains and identify cracks on well-maintained asphalt. These findings suggest that the model could prove valuable for applications such as autonomous cleaning robots and robot runway patrols. Additionally, exploring medical image one-shot segmentation using MOSTS or a similar method could be a significant contribution. Given a reference image of a cancer cell cluster, an ideal one-shot model might be able to highlight cancer tissues in medical images based on texture contrast.

Looking forward, researchers could consider few-shot and zero-shot implementations of the proposed work. Instead of relying on a single image, the model could be trained to extract effective visual cues from a group of reference images, achieving a more comprehensive understanding of the targeted texture class. Alternatively, by coupling the MOSTS model with superpixel models like [176], a zero-shot texture segmentation model could be developed. This model could be trained using contrastive learning techniques like [177], eliminating the need for any reference image and further expanding the adaptability and applicability of the framework.

While Chapter 4 presented positive outcomes, several areas for future investigation and enhancement remain. First, a design iteration on the current prototype (e.g., upgrading single-beam LiDARs to RGB-D camera systems or multi-beam LiDARs, upgrading the single channel speaker to multi-channel, etc.) is highly recommended. Second, the static nature of the proposed test courses restricts their capacity to reflect the challenges of dynamic and high-speed navigation. To fully assess the performance of the proposed shared-control system, future researchers should design more comprehensive tests that incorporate dynamic obstacles and high-speed maneuvers. If conducting such tests in physical environments proves challenging, researchers could consider virtual reality simulations, similar to those in [41], as an alternative.

Additionally, the exploration of a more robust passive shared-control architecture warrants further study. A thorough analysis of its passivity property, akin to the work in [45], is desired. Currently, the implementation of the PAPF shared-control is algorithmic. A more rigorous, model-based velocity field implementation could improve the robustness of the shared-control design and achieve true passivity - ensuring that the controller cannot generate more energy than it receives from external sources.

Moreover, implementing alternative reactive shared-control paradigms, such as the Dynamic Window Approach (DWA) [55] and deliberative shared-control approaches [49]–[51], [53], on the

PURE platform could provide valuable benchmarking opportunities and deeper insights into the field of assistive mobility. For deliberative shared control, methods like the Koopman operator theory and GPU parallel processing could be utilized to facilitate real-time performance [178]. Such studies could expand the boundaries of shared control design and guide the development of more adaptive, responsive, and effective assistive mobility solutions.

Lastly, exploring the addition of fully autonomous features such as self-homing and self-docking could yield practical benefits. Future users would certainly appreciate situations where the PURE system autonomously navigates to its charging station after bedtime and then returns to pick up the user by the bed in the morning. These task-based autonomous features could significantly enhance the user experience and the overall utility of the system.

REFERENCES

[1] “Mobile robot.” Accessed: Jun. 12, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Mobile_robot#cite_note-tvt2-1

[2] “Roomba.” Accessed: Jun. 12, 2022. [Online]. Available: https://www.irobot.com/

[3] “‘The first self-balancing, electricpowered transportation device.’” Accessed: Jun. 12, 2022. [Online]. Available: https://www.segway.com/

[4] “Boston Dynamic Spot.” Accessed: Jun. 12, 2022. [Online]. Available: https://www.bostondynamics.com/products/spot

[5] A. Elfes, “Using Occupancy Grids for Mobile Robot Perception and Navigation,” *Computer (Long Beach Calif)*, vol. 22, no. 6, pp. 46–57, 1989, doi: 10.1109/2.30720.

[6] R. Siegwart, *Intro to Autonomous Mobile Robots*. 2004.

[7] U. Nagarajan, G. Kantor, and R. Hollis, “The ballbot: An omnidirectional balancing mobile robot,” *International Journal of Robotics Research*, vol. 33, no. 6, pp. 917–930, 2014, doi: 10.1177/0278364913509126.

[8] M. Kumagai and T. Ochiai, “Development of a robot balancing on a ball,” *2008 International Conference on Control, Automation and Systems, ICCAS 2008*, pp. 433–438, 2008, doi: 10.1109/ICCAS.2008.4694680.

[9] P. Fankhauser and C. Gwerder, “Modeling and Control of a Ballbot,” 2010. doi: 10.3929/ethz-a-010056685.

[10] C. Cai, J. Lu, and Z. Li, “Kinematic Analysis and Control Algorithm for the Ballbot,” *IEEE Access*, vol. 7, pp. 38314–38321, 2019, doi: 10.1109/ACCESS.2019.2902219.

[11] Y. Ech-Choudany, R. Grasse, R. Stock, O. Horn, and G. Bourhis, “Traded and combined cooperative control of a smart wheelchair,” *Robotica*, vol. 40, no. 8, pp. 2630–2650, Aug. 2022, doi: 10.1017/S0263574721001855.

[12] A. Simpkins, “Underactuated mechanical systems,” *IEEE Robot Autom Mag*, vol. 21, no. 1, 2014, doi: 10.1109/MRA.2014.2299502.

[13] P. Oryschuk, A. Salerno, A. M. Al-husseini, and J. Angeles, “Experimental Validation of an Underactuated Two-Wheeled Mobile Robot,” *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 27–27, 2009, doi: 10.1109/vissof.2009.5336427.

[14] X. Chen, A. S. Vempati, and P. Beardsley, “StreetMap - Mapping and Localization on Ground Planes using a Downward Facing Camera,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 1672–1679, 2018, doi: 10.1109/IROS.2018.8594157.

[15] “What is SWaP-C.” Accessed: Jun. 12, 2022. [Online]. Available: https://www.baesystems.com/en-us/definition/what-is-swap-c

[16] D. P. Miller, *Assistive robotics: an overview." Assistive technology and artificial intelligence*. 1998.

[17] U. Borgolte, H. Hoyer, C. Bühler, H. Heck, and R. Hoelper, “Architectural Concepts of a Semi-autonomous Wheelchair,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 22, no. 3–4, pp. 233–253, 1998, doi: 10.1023/a:1007944531532.

[18] T. Röfer and A. Lankenau, “Architecture and applications of the Bremen autonomous wheelchair,” *Inf Sci (N Y)*, vol. 126, no. 1, pp. 1–20, 2000, doi: 10.1016/S0020-0255(00)00020-7.

[19] H. Grewal, A. Matthews, R. Tea, and K. George, “LIDAR-based autonomous wheelchair,” *SAS 2017 - 2017 IEEE Sensors Applications Symposium, Proceedings*, 2017, doi: 10.1109/SAS.2017.7894082.

[20] C. Mandel, K. Huebner, T. Vierhuff, and M. Christian, “Towards an Autonomous Wheelchair: Cognitive Aspects in Service Robotics,” *Towards Autonomous Robotic Systems*, pp. 165–172, 2005.

[21] G. Pires, N. Honorio, C. Lopes, U. Nunes, and A. T. Almeida, “Autonomous wheelchair for disabled people,” *IEEE International Symposium on Industrial Electronics*, vol. 3, pp. 797–801, 1997, doi: 10.1109/isie.1997.648641.

[22] B. F. Wu, C. L. Jen, W. F. Li, T. Y. Tsou, P. Y. Tseng, and K. T. Hsiao, “RGB-D sensor based SLAM and human tracking with Bayesian framework for wheelchair robots,” *2013 International Conference on Advanced Robotics and Intelligent Systems, ARIS 2013 - Conference Proceedings*, pp. 110–115, 2013, doi: 10.1109/ARIS.2013.6573544.

[23] H. Xiao, Z. Li, C. Yang, W. Yuan, and L. Wang, “RGB-D sensor-based visual target detection and tracking for an intelligent wheelchair robot in indoors environments,” *Int J Control Autom Syst*, vol. 13, no. 3, pp. 521–529, 2015, doi: 10.1007/s12555-014-0353-4.

[24] Z. Li, Y. Xiong, and L. Zhou, “ROS-based indoor autonomous exploration and navigation wheelchair,” *Proceedings - 2017 10th International Symposium on Computational Intelligence and Design, ISCID 2017*, vol. 2, pp. 132–135, 2018, doi: 10.1109/ISCID.2017.55.

[25] H. Wang, Y. Sun, and M. Liu, “Self-Supervised Drivable Area and Road Anomaly Segmentation Using RGB-D Data for Robotic Wheelchairs,” *IEEE Robot Autom Lett*, vol. 4, no. 4, pp. 4386–4393, 2019, doi: 10.1109/LRA.2019.2932874.

[26] H. Wang, Y. Sun, R. Fan, and M. Liu, “S2P2: Self-Supervised Goal-Directed Path Planning Using RGB-D Data for Robotic Wheelchairs,” in *IEEE International Conference on Robotics and Automation (ICRA),* 2021, pp. 11422–11428. doi: 10.1109/icra48506.2021.9561314.

[27] B. Kim and J. Pineau, “Socially Adaptive Path Planning in Human Environments Using Inverse Reinforcement Learning,” *Int J Soc Robot*, vol. 8, no. 1, pp. 51–66, 2016, doi: 10.1007/s12369-015-0310-2.

[28] A. Erdogan and B. D. Argall, “The effect of robotic wheelchair control paradigm and interface on user performance, effort and preference: An experimental assessment,” *Rob Auton Syst*, vol. 94, pp. 282–297, 2017, doi: 10.1016/j.robot.2017.04.013.

[29] B. Zhang, C. Holloway, and T. Carlson, “A hierarchical design for shared-control wheelchair navigation in dynamic environments,” *Conf Proc IEEE Int Conf Syst Man Cybern*, vol. 2020-Octob, pp. 4439–4446, 2020, doi: 10.1109/SMC42975.2020.9282838.

[30] A. Erdogan and B. D. Argall, “Prediction of user preference over shared-control paradigms for a robotic wheelchair,” *IEEE International Conference on Rehabilitation Robotics*, pp. 1106–1111, 2017, doi: 10.1109/ICORR.2017.8009397.

[31] D. J. Gonon, D. Paez-Granados, and A. Billard, “Reactive Navigation in Crowds for Non-Holonomic Robots with Convex Bounding Shape,” *IEEE Robot Autom Lett*, vol. 6, no. 3, pp. 4728–4735, 2021, doi: 10.1109/LRA.2021.3068660.

[32] R. Tomari, Y. Kobayashi, and Y. Kuno, “Socially acceptable smart wheelchair navigation from head orientation observation,” *International journal on smart sensing and intelligent systems,* vol. 7, no. 2, pp. 630–643, 2014.

[33] A. C. Plascencia and J. Rozman, “Towards a user-wheelchair shared control paradigm for individuals with severe motor impairments,” *ICINCO 2016 - Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics*, vol. 2, no. Icinco, pp. 246–253, 2016, doi: 10.5220/0005973302460253.

[34] T. V. How, R. H. Wang, and A. Mihailidis, “Evaluation of an intelligent wheelchair system for older adults with cognitive impairments.,” *J Neuroeng Rehabil*, vol. 10, p. 90, 2013, doi: 10.1186/1743-0003-10-90.

[35] M. M. Martins, C. P. Santos, A. Frizera-Neto, and R. Ceres, “Assistive mobility devices focusing on Smart Walkers: Classification and review,” *Rob Auton Syst*, vol. 60, no. 4, pp. 548–562, 2012, doi: 10.1016/j.robot.2011.11.015.

[36] S. Y. Jiang, C. Y. Lin, K. T. Huang, and K. T. Song, “Shared Control Design of a Walking-Assistant Robot,” *IEEE Transactions on Control Systems Technology*, vol. 25, no. 6, pp. 2143–2150, 2017, doi: 10.1109/TCST.2016.2638879.

[37] G. A. Mutiara, G. I. Hapsari, and R. Rijalul, “Smart guide extension for blind cane,” *2016 4th International Conference on Information and Communication Technology, ICoICT 2016*, vol. 4, no. c, pp. 1–6, 2016, doi: 10.1109/ICoICT.2016.7571896.

[38] Y. H. Lee and G. Medioni, “RGB-D camera Based Navigation for the Visually Impaired,” in *RSS 2011 Workshop on RGB-D Cameras*, 2011, pp. 1–6.

[39] P. Kolar, P. Benavidez, and M. Jamshidi, “Survey of datafusion techniques for laser and vision based sensor integration for autonomous navigation,” *Sensors (Switzerland)*, vol. 20, no. 8, pp. 1–49, 2020, doi: 10.3390/s20082180.

[40] P. Kolar, P. Benavidez, and M. Jamshidi, “Survey of data fusion techniques for laser and vision based sensor integration for autonomous navigation,” *Sensors (Switzerland)*, vol. 20, no. 8. MDPI AG, Apr. 02, 2020. doi: 10.3390/s20082180.

[41] D. Baek, Y. Chen, Chang, and J. Ramos, “A Study of Shared-Control with Force Feedback for Obstacle Avoidance in Whole-body Telelocomotion of a Wheeled Humanoid,” Sep. 2022, [Online]. Available: http://arxiv.org/abs/2209.03994

[42] T. Hoshino, S. Yokota, and T. Chino, “OmniRide: A Personal Vehicle with 3 DOF Mobility.”

[43] D. B. Pham, H. Kim, J. Kim, and S. G. Lee, “Balancing and Transferring Control of a Ball Segway Using a Double-Loop Approach [Applications of Control],” *IEEE Control Syst*, vol. 38, no. 2, pp. 15–37, Apr. 2018, doi: 10.1109/MCS.2017.2786444.

[44] P. D. Nisbet, “Who’s intelligent? Wheelchair, driver or both?,” in *In Proceedings of the International Conference on Control Applications IEEE*, 2002, pp. 760–765.

[45] P. W. Rushton, B. W. Mortenson, P. Viswanathan, R. H. Wang, W. C. Miller, and L. Hurd Clarke, “Intelligent power wheelchair use in long-term care: potential users’ experiences and perceptions,” *Disabil Rehabil Assist Technol*, vol. 12, no. 7, pp. 740–746, Oct. 2017, doi: 10.1080/17483107.2016.1260653.

[46] C. Urdiales *et al.*, “An Adaptive Scheme for Wheelchair Navigation Collaborative Control.” [Online]. Available: www.aaai.org

[47] C. Urdiales *et al.*, “A new multi-criteria optimization strategy for shared control in wheelchair assisted navigation,” *Auton Robots*, vol. 30, no. 2, pp. 179–197, Feb. 2011, doi: 10.1007/s10514-010-9211-2.

[48] C. X. Miller, T. Gebrekristos, M. Young, E. Montague, and B. Argall, “An Analysis of Human-Robot Information Streams to Inform Dynamic Autonomy Allocation,” in *IEEE International Conference on Intelligent Robots and Systems*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 1872–1878. doi: 10.1109/IROS51168.2021.9636637.

[49] V. K. Narayanan, A. Spalanzani, and M. Babel, “A semi-autonomous framework for human-aware and user intention driven wheelchair mobility assistance,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4700–4707.

[50] J. G. Storms and D. M. Tilbury, “Blending of human and obstacle avoidance control for a high speed mobile robot,” in *Proceedings of the American Control Conference*, Institute of Electrical and Electronics Engineers Inc., 2014, pp. 3488–3493. doi: 10.1109/ACC.2014.6859352.

[51] L. Devigne, V. K. Narayanan, F. Pasteau, and M. Babel, “Low complex sensor-based shared control for power wheelchair navigation,” in *IEEE International Conference on Intelligent Robots and Systems*, Institute of Electrical and Electronics Engineers Inc., Nov. 2016, pp. 5434–5439. doi: 10.1109/IROS.2016.7759799.

[52] X. Deng, Z. Liang Yu, C. Lin, Z. Gu, and Y. Li, “Self-adaptive shared control with brain state evaluation network for human-wheelchair cooperation,” *J Neural Eng*, vol. 17, no. 4, Aug. 2020, doi: 10.1088/1741-2552/ab937e.

[53] E. Demeester, A. Hüntemann, D. Vanhooydonck, G. Vanacker, H. Van Brussel, and M. Nuttin, “User-adapted plan recognition and user-adapted shared control: A Bayesian approach to semi-autonomous wheelchair driving,” *Auton Robots*, vol. 24, no. 2, pp. 193–211, 2008, doi: 10.1007/s10514-007-9064-5.

[54] M. A. Hadj-Abdelkader, G. Bourhis, and B. Cherki, “Haptic feedback control of a smart wheelchair,” *Appl Bionics Biomech*, vol. 9, no. 2, pp. 181–192, 2012, doi: 10.3233/ABB-2012-0067.

[55] H. Grewal, A. Matthews, R. Tea, and K. George, “LIDAR-based autonomous wheelchair,” in *SAS 2017 - 2017 IEEE Sensors Applications Symposium, Proceedings*, Institute of Electrical and Electronics Engineers Inc., Apr. 2017. doi: 10.1109/SAS.2017.7894082.

[56] M. R. Petry, A. P. Moreira, R. A. Braga, and L. P. Reis, “Shared Control for Obstacle Avoidance in Intelligent Wheelchairs,” in *IEEE Conference on Robotics, Automation and Mechatronics*, 2010, pp. 182–187.

[57] Z. Li, S. Zhao, J. Duan, C. Y. Su, C. Yang, and X. Zhao, “Human Cooperative Wheelchair With Brain-Machine Interaction Based on Shared Control Strategy,” *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 1, pp. 185–195, Feb. 2017, doi: 10.1109/TMECH.2016.2606642.

[58] Y. Kondo, Takanori Miyoshi, Kazuhiko Terashima, and Hideo Kitagawa, “Navigation guidance control using haptic feedback for obstacle avoidance of omni-directional wheelchair,” in *Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, IEEE Xplore, 2008, pp. 437–444.

[59] Vander Poorten *et al.*, "Powered Wheelchair Navigation Assistance through Kinematically Correct Environmental Haptic Feedback," in *IEEE International Conference on Robotics and Automation*, [IEEE], 2012, pp. 3706–3712.

[60] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation," *Auton Robots*, vol. 18, no. 1, pp. 81–102, 2005.

[61] I. Halatci, C. A. Brooks, and K. Iagnemma, "Terrain classification and classifier fusion for planetary exploration rovers," *IEEE Aerospace Conference Proceedings*, 2007, doi: 10.1109/AERO.2007.352692.

[62] J. C. Andersen, M. R. Blas, O. Ravn, N. A. Andersen, and M. Blanke, "Traversable terrain classification for outdoor autonomous robots using single 2D laser scans," *Integr Comput Aided Eng*, vol. 13, no. 3, pp. 223–232, 2006, doi: 10.3233/ica-2006-13303.

[63] K. M. Wurm, R. Kümmerle, C. Stachniss, and W. Burgard, "Improving robot navigation in structured outdoor environments by identifying vegetation from laser data," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pp. 1217–1222, 2009, doi: 10.1109/IROS.2009.5354530.

[64] L. Ojeda, J. Borenstein, G. Witus, and Robert Karlsen, "Terrain Characterization and Classification with a Mobile Robot," *J Field Robot*, vol. 23, no. 2, pp. 103–122, 2006, doi: 10.1002/rob.

[65] A. Valada and W. Burgard, "Deep spatiotemporal models for robust proprioceptive terrain classification," *International Journal of Robotics Research*, vol. 36, no. 13–14, pp. 1521–1539, 2017, doi: 10.1177/0278364917727062.

[66] D. Tick, T. Rahman, C. Busso, and N. Gans, "Indoor robotic terrain classification via angular velocity based hierarchical classifier selection," *Proc IEEE Int Conf Robot Autom*, pp. 3594–3600, 2012, doi: 10.1109/ICRA.2012.6225128.

[67] C. Spiteri, S. Al-Milli, Y. Gao, and A. Sarrionandia De León, "Real-time visual sinkage detection for planetary rovers," *Rob Auton Syst*, vol. 72, pp. 307–317, 2015, doi: 10.1016/j.robot.2015.06.009.

[68] C. Bai, J. Guo, and H. Zheng, "Three-Dimensional Vibration-Based Terrain Classification for Mobile Robots," *IEEE Access*, vol. 7, pp. 63485–63492, 2019, doi: 10.1109/ACCESS.2019.2916480.

[69] E. G. Collins and E. J. Coyle, "Vibration-based terrain classification using surface profile input frequency responses," *Proc IEEE Int Conf Robot Autom*, pp. 3276–3283, 2008, doi: 10.1109/ROBOT.2008.4543710.

[70] D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, and H. M. Gross, "Efficient RGB-D Semantic Segmentation for Indoor Scene Analysis," *Proc IEEE Int Conf Robot Autom*, vol. 2021-May, no. Icra, pp. 13525–13531, 2021, doi: 10.1109/ICRA48506.2021.9561675.

[71] S. P. Parikh, “A framework for shared motion control: Human -robot augmentation with applications to assistive technology,” 2005.

[72] Q. Li, W. Chen, and J. Wang, “Dynamic shared control for human-wheelchair cooperation,” *Proc IEEE Int Conf Robot Autom*, pp. 4278–4283, 2011, doi: 10.1109/ICRA.2011.5980055.

[73] J. Shen, J. Ibañez-Guzmán, T. C. Ng, and B. S. Chew, “A collaborative-shared control system with safe obstacle avoidance capability,” *2004 IEEE Conference on Robotics, Automation and Mechatronics*, pp. 119–123, 2004, doi: 10.1109/ramech.2004.1438902.

[74] Y. Chen, C. Rastogi, and W. R. Norris, “A CNN based vision-proprioception fusion method for robust UGV Terrain classification,” *IEEE Robot Autom Lett*, vol. 6, no. 4, pp. 7965–7972, 2021, doi: 10.1109/LRA.2021.3101866.

[75] Y. Chen, C. Rastogi, Z. Zhou, and W. R. Norris, “A Self-Supervised Miniature One-Shot Texture Segmentation (MOSTS) Model for Real-Time Robot Navigation and Embedded Applications,” *arXiv preprint arXiv:2306.08814*, 2023, Accessed: Jun. 30, 2023. [Online]. Available: https://arxiv.org/abs/2306.08814

[76] S. Otte, C. Weiss, T. Scherer, and A. Zell, “Recurrent neural networks for fast and robust vibration-based ground classification on mobile robots,” *Proc IEEE Int Conf Robot Autom*, vol. 2016-June, pp. 5603–5608, 2016, doi: 10.1109/ICRA.2016.7487778.

[77] C. Weiss, H. Fröhlich, and A. Zell, “Vibration-based terrain classification using support vector machines,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 4429–4434, 2006, doi: 10.1109/IROS.2006.282076.

[78] C. Weiss, H. Tamimi, and A. Zell, “A combination of vision- and vibration-based terrain classification,” *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pp. 2204–2209, 2008, doi: 10.1109/IROS.2008.4650678.

[79] G. Reina, A. Milella, and R. Galati, “Terrain assessment for precision agriculture using vehicle dynamic modelling,” *Biosyst Eng*, vol. 162, pp. 124–139, 2017, doi: 10.1016/j.biosystemseng.2017.06.025.

[80] E. Coyle and E. G. Collins, “A Comparison of Classifier Performance for Vibration-Based Terrain Classification,” *Army Science Conference*, pp. 1–4, 2008.

[81] R. Gonzalez and K. Iagnemma, “Deep Terrame chanics: Terrain classification and slip estimation for ground robots via deep learning,” *ArXiv*, 2018.

[82] G. Reina and R. Galati, “Slip-based terrain estimation with a skid-steer vehicle,” *Vehicle System Dynamics*, vol. 54, no. 10, pp. 1384–1404, 2016, doi: 10.1080/00423114.2016.1203961.

[83] E. Coyle *et al.*, “Vibration-based terrain classification for electric powered wheelchairs,” *Proceedings of the 4th IASTED International Conference on Telehealth and Assistive Technologies, Telehealth/AT 2008*, no. January 2014, pp. 139–144, 2008.

[84] E. G. Collins and E. J. Coyle, “Vibration-based terrain classification using surface profile input frequency responses,” *Proc IEEE Int Conf Robot Autom*, pp. 3276–3283, 2008, doi: 10.1109/ROBOT.2008.4543710.

[85] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, “Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks,” *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2015-Augus, pp. 4580–4584, 2015, doi: 10.1109/ICASSP.2015.7178838.

[86] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo, “Convolutional LSTM network: A machine learning approach for precipitation nowcasting,” *Adv Neural Inf Process Syst*, vol. 2015-Janua, pp. 802–810, 2015.

[87] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative study of CNN and RNN for natural language processing,” *ArXiv*, 2017.

[88] Y. Gao, L. A. Hendricks, K. J. Kuchenbecker, and T. Darrell, “Deep learning for tactile understanding from visual and haptic data,” *Proc IEEE Int Conf Robot Autom*, vol. 2016-June, pp. 536–543, 2016, doi: 10.1109/ICRA.2016.7487176.

[89] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, “MobileNetV2: Inverted residuals and linear bottlenecks,” *ArXiv*, pp. 4510–4520, 2018.

[90] K. Sentz and S. Ferson, “Combination of Evidence in Dempster- Shafer Theory,” *Contract*, no. April, p. 96, 2002.

[91] Y. Su, K. Zhang, J. Wang, and K. Madani, “Environment sound classification using a two-stream CNN based on decision-level fusion,” *Sensors (Switzerland)*, vol. 19, no. 7, pp. 1–15, 2019, doi: 10.3390/s19071733.

[92] P. Oryschuk, A. Salerno, A. M. Al-Husseini, and J. Angeles, “Experimental validation of an underactuated two-wheeled mobile robot,” *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 252–257, 2009.

[93] Xu Chen, Anurag Sai Vempati, and Paul Beardsley, “StreetMap - Mapping and Localization on Ground Planes using aDownward Facing Camera,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 1672–1679.

[94] A. Kirillov *et al.*, “Segment Anything,” Apr. 2023, [Online]. Available: http://arxiv.org/abs/2304.02643

[95] K. Perlin, “An Image Synthesizer,” 1985.

[96] G. Gini and A. Marchi, “Indoor Robot Navigation with Single Camera Vision,” *PRIS*, vol. 2, pp. 67–76, 2002.

[97] F. G. Rodriguez-Telles, L. A. Torres-Mendez, and E. A. Martínez-García, “A fast floor segmentation algorithm for visual-based robot navigation,” in *Proceedings - 2013 International Conference on Computer and Robot Vision, CRV 2013*, 2013, pp. 167–173.

[98] A. Singhani, “Real-Time Freespace Segmentation on Autonomous Robots for Detection of Obstacles and Drop-Offs,” Feb. 2019.

[99] Z. Winger, “Free Space Detection and Trajectory Planning for Autonomous Robot,” 2020.

[100] D. Teso-Fz-Betoño, E. Zulueta, A. Sánchez-Chica, U. Fernandez-Gamiz, and A. Saenz-Aguirre, “Semantic segmentation to develop an indoor navigation system for an autonomous mobile robot,” *Mathematics*, vol. 8, no. 5, p. 855, May 2020.

[101] S. Kumar, M. Siva Karthik, and K. Madhava Krishna, “Markov Random Field based Small Obstacle Discovery over Images,” in *IEEE International Conference on Robotics and Automation (ICRA)* , 2014, pp. 494–500.

[102] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, “SLIC Superpixels,” 2010.

[103] D. Sherrington and S. Kirkpatrick, “Solvable model of a spin-glass,” *Phys Rev Lett*, vol. 35, no. 26, p. 1792, 1975.

[104] V. Kolmogorov and R. Zabih, “What Energy Functions Can Be Minimized via Graph Cuts?,” *IEEE Trans Pattern Anal Mach Intell*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[105] P. Pinggera, S. Ramos, S. Gehrig, U. Franke, C. Rother, and R. Mester, “Lost and Found: Detecting Small Road Hazards for Self-Driving Vehicles,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2016, pp. 1099–1106.

[106] M. Hua, Y. Nan, and S. Lian, “Small Obstacle Avoidance Based on RGB-D Semantic Segmentation,” in *IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[107] H. Wang, Y. Sun, and M. Liu, “Self-Supervised Drivable Area and Road Anomaly Segmentation Using RGB-D Data for Robotic Wheelchairs,” *IEEE Robot Autom Lett*, vol. 4, no. 4, pp. 4386–4393, Oct. 2019.

[108] H. Wang, R. Fan, Y. Sun, and M. Liu, “Applying surface normal information in drivable area and road anomaly detection for ground mobile robots,” in *IEEE International Conference on Intelligent Robots and Systems*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 2706–2711.

[109] L. Sun, K. Yang, X. Hu, W. Hu, and K. Wang, “Real-Time fusion network for rgb-d sementic segmentation incorporating unexpected obstacle detection,” *IEEE Robot Autom Lett*, vol. 5, no. 4, pp. 5558–5565, Oct. 2020.

[110] M. Himmelsbach, F. V. Hundelshausen, and H. J. Wuensche, “Fast segmentation of 3D point clouds for ground vehicles,” in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2010, pp. 560–565.

[111] B. Douillard *et al.*, “On the segmentation of 3D lidar point clouds,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 2798–2805.

[112] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, “Real-Time Plane Segmentation Using RGB-D Cameras,” in *Robot Soccer World Cup*, 2011, pp. 306–317.

[113] Z. Wang, H. Liu, Y. Qian, and T. Xu, “Real-Time Plane Segmentation and Obstacle Detection of 3D Point Clouds for Indoor Scenes,” in *European Conference on Computer Vision*, 2012, pp. 22–31.

[114] R. A. Zeineldin and N. A. El-Fishawy, “Fast and accurate ground plane detection for the visually impaired from 3D organized point clouds,” in *Proceedings of 2016 SAI Computing Conference, SAI 2016*, Institute of Electrical and Electronics Engineers Inc., Aug. 2016, pp. 373–379.

[115] D. Zermas, I. Izzat, and N. Papanikolopoulos, “Fast segmentation of 3D point clouds: A paradigm on LiDAR data for autonomous vehicle applications,” in *Proceedings - IEEE International Conference on Robotics and Automation*, Institute of Electrical and Electronics Engineers Inc., Jul. 2017, pp. 5067–5073.

[116] A. Paigwar, O. Erkent, D. Sierra-Gonzalez, and C. Laugier, “GndNet: Fast ground plane estimation and point cloud segmentation for autonomous vehicles,” in *IEEE International Conference on Intelligent Robots and Systems*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 2150–2156.

[117] J. Byun, K.-I. Na, B.-S. Seo, and M. Roh, “Drivable road detection with 3D Point Clouds based on the MRF for Intelligent Vehicle,” in *Field and Service Robotics*, 2015, pp. 49–60.

[118] A. Hacinecipoglu, E. I. Konukseven, and A. B. Koku, “Pose invariant people detection in point clouds for mobile robots,” *International Journal of Mechanical Engineering and Robotics Research*, vol. 9, no. 5, pp. 709–715, May 2020.

[119] S.-J. Park, K.-S. Hong, and S. Lee, “RDFNet: RGB-D Multi-level Residual Feature Fusion for Indoor Semantic Segmentation,” in *IEEE international conference on computer vision*, 2017, pp. 4980–4989.

[120] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, “FuseNet: Incorporating depth into semantic segmentation via fusion-based CNN architecture,” in *Asian conference on computer vision*, Springer Verlag, 2016, pp. 213–228.

[121] Y. Xing, J. Wang, X. Chen, and G. Zeng, “Coupling two-stream RGB-D semantic segmentation network by idempotent mappings,” in *IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 1850–1854.

[122] X. Hu, K. Yang, L. Fei, and K. Wang, "Acnet: Attention based network to exploit complementary features for rgbd semantic segmentation," in *IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 1440–1444.

[123] F. Fooladgar and S. Kasaei, "Multi-Modal Attention-based Fusion Model for Semantic Segmentation of RGB-Depth Images," Dec. 2019.

[124] A. Valada, R. Mohan, and W. Burgard, "Self-Supervised Model Adaptation for Multimodal Semantic Segmentation," *Int J Comput Vis*, vol. 128, no. 5, pp. 1239–1285, May 2020.

[125] X. Chen *et al.*, "Bi-directional Cross-Modality Feature Propagation with Separation-and-Aggregation Gate for RGB-D Semantic Segmentation," in *European Conference on Computer Vision*, Jul. 2020, pp. 561–577.

[126] J. Jiang, L. Zheng, F. Luo, and Z. Zhang, "RedNet: Residual Encoder-Decoder Network for indoor RGB-D Semantic Segmentation," Jun. 2018.

[127] D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, and H. M. Gross, "Efficient RGB-D Semantic Segmentation for Indoor Scene Analysis," in *IEEE International Conference on Robotics and Automation*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 13525–13531.

[128] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D Scene Understanding Benchmark Suite," in *IEEE conference on computer vision and pattern recognition*, 2015, pp. 567–576.

[129] I. Ustyuzhaninov, C. Michaelis, W. Brendel, and M. Bethge, "One-shot Texture Segmentation," Jul. 2018.

[130] K. Zhu, W. Zhai, Z.-J. Zha, and Y. Cao, "One-shot texture retrieval using global grouping metric," *IEEE Trans Multimedia*, vol. 23, pp. 3726–3737, 2020.

[131] J. Xue, H. Zhang, and K. Dana, "Deep Texture Manifold for Ground Terrain Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 558–567.

[132] Y. Huang, C. Qiu, X. Wang, S. Wang, and K. Yuan, "A compact convolutional neural network for surface defect inspection," *Sensors (Switzerland)*, vol. 20, no. 7, Apr. 2020, doi: 10.3390/s20071974.

[133] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013.

[134] R. G. Valenti, I. Dryanovski, and J. Xiao, "Keeping a good attitude: A quaternion-based orientation filter for IMUs and MARGs," *Sensors (Switzerland)*, vol. 15, no. 8, pp. 19302–19330, Aug. 2015.

[135] K. Xiao, L. Engstrom, A. Ilyas, and A. Madry, “Noise or Signal: The Role of Image Backgrounds in Object Recognition,” Jun. 2020.

[136] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *In Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[137] Nabila Abraham and Naimul Mefraz Khan, “A novel focal tversky loss function with improved attention u-net for lesion segmentation,” in *IEEE 16th international symposium on biomedical imaging (ISBI 2019)*, 2019, pp. 683–687.

[138] S. Xie and Z. Tu, “Holistically-nested edge detection,” in *In Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1395–1403.

[139] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso, “Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Verlag, 2017, pp. 240–248. doi: 10.1007/978-3-319-67558-9_28.

[140] O. Kodym, M. Španěl, and A. Herout, “Segmentation of Head and Neck Organs at Risk Using CNN with Batch Dice Loss,” in *German conference on pattern recognition*, Dec. 2018, pp. 105–114.

[141] S. A. Taghanaki *et al.*, “Combo Loss: Handling Input and Output Imbalance in Multi-Organ Segmentation,” *Computerized Medical Imaging and Graphics*, vol. 75, pp. 24–33, May 2019.

[142] S. Jadon, “A survey of loss functions for semantic segmentation,” in *IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, 2020, pp. 1–7.

[143] M. Cimpoi, S. Maji, I. Kokkinosécole, S. Mohamed, and A. Vedaldi, “Describing Textures in the Wild,” in *IEEE conference on computer vision and pattern recognition*, 2014, pp. 3606–3613.

[144] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[145] M. Tan and Q. v. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” in *International conference on machine learning*, May 2019, pp. 6105–6114.

[146] M. Tan and Q. v. Le, “EfficientNetV2: Smaller Models and Faster Training,” in *International Conference on Machine Learning*, Apr. 2021, pp. 10096–10106.

[147] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *IEEE conference on computer vision and pattern recognition*, Jan. 2018, pp. 4510–4520.

[148] A. Howard *et al.*, "Searching for MobileNetV3," in *IEEE/CVF international conference on computer vision*, May 2019, pp. 1314–1324.

[149] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *IEEE conference on computer vision and pattern recognition*, Institute of Electrical and Electronics Engineers (IEEE), Mar. 2009, pp. 248–255.

[150] X. Zhang, Y. Wei, Y. Yang, and T. Huang, "SG-One: Similarity Guidance Network for One-Shot Semantic Segmentation," Oct. 2018, [Online]. Available: http://arxiv.org/abs/1810.09091

[151] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," Jul. 2018, [Online]. Available: http://arxiv.org/abs/1807.06521

[152] C. Xiao, "Personal Unique Rolling Experience: Design, Modeling, and Control of a Riding Ballbot," Doctoral dissertation, University of Illinois Urbana-Champaign, Mechanical Science & Engineering, Urbana, IL, 2022.

[153] C. Xiao, M. Mansouri, D. Lam, J. Ramos, and E. T. Hsiao-Wecksler, "Design and Control of a Ballbot Drivetrain with High Agility, Minimal Footprint, and High Payload," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2023)*, Apr. 2023. [Online]. Available: http://arxiv.org/abs/2304.02887

[154] S. Y. Song *et al.*, "Design and Validation of a Torso-Dynamics Estimation System (TES) for Hands-Free Physical Human-Robot Interaction," in *32nd IEEE International Conference on Robot and Human Interactive Communication, IEEE RO-MAN 2023*, 2023. doi: 10.36227/techrxiv.22336843.v2.

[155] M. W. Spong, "Underactuated mechanical systems," in *Control Problems in Robotics and Automation*, London: Springer-Verlag, 1998, pp. 135–150. doi: 10.1007/BFb0015081.

[156] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *Int J Rob Res*, vol. 5, no. 1, pp. 90–98, 1986.

[157] R. Madarasz, L. Heiny, R. Cromp, and N. Mazur, "The design of an autonomous vehicle for the disabled," *IEEE Journal on Robotics and Automation*, vol. 2, no. 3, pp. 117–126, 1986, doi: 10.1109/JRA.1986.1087052.

[158] S. P. Levine *et al.*, "The NavChair Assistive Wheelchair Navigation System," 1999.

[159] J. S. Albus, "Outline for a Theory of Intelligence," 1991.

[160] Y. Hirata, A. Hara, and K. Kosuge, “Motion control of passive intelligent walker using servo brakes,” in *IEEE Transactions on Robotics*, Oct. 2007, pp. 981–990. doi: 10.1109/TRO.2007.906252.

[161] A. Gottardi, S. Tortora, E. Tosello, and E. Menegatti, “Shared Control in Robot Teleoperation With Improved Potential Fields,” *IEEE Trans Hum Mach Syst*, vol. 52, no. 3, pp. 410–422, Jun. 2022, doi: 10.1109/THMS.2022.3155716.

[162] H. Haddad, S. Lacroix, and R. Chatila, “Reactive Navigation in Outdoor Environments using Potential Fields,” in *In Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, 1998, pp. 1232–1237.

[163] A. Poncela, C. Urdiales, E. J. Pérez, and F. Sandoval, “A new efficiency-weighted strategy for continuous human/robot cooperation in navigation,” *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 39, no. 3, pp. 486–500, 2009, doi: 10.1109/TSMCA.2009.2013191.

[164] Y. Chen, C. Rastogi, Z. Zhou, and W. R. Norris, “A Self-Supervised Miniature One-Shot Texture Segmentation (MOSTS) Model for Real-Time Robot Navigation and Embedded Applications,” *arXiv preprint*, Jun. 2023, [Online]. Available: http://arxiv.org/abs/2306.08814

[165] paralympic.org, “CLASSIFICATION IN PARA ATHLETICS,” https://www.paralympic.org/athletics/classification.

[166] Uniform Federal Accessibility Standards Architecture and Transportation Barriers Compliance Board, “Uniform Federal Accessibility Standards (1984),” https://www.access-board.gov/aba/ufas.html.

[167] P. Hur, K. S. Rosengren, G. P. Horn, D. L. Smith, and E. T. Hsiao-Wecksler, “Effect of Protective Clothing and Fatigue on Functional Balance of Firefighters,” *Journal of Ergonomics*, vol. S2, no. 02, 2014, doi: 10.4172/2165-7556.s2-004.

[168] R. M. Kesler *et al.*, “Impact of SCBA size and firefighting work cycle on firefighter functional balance,” *Appl Ergon*, vol. 69, pp. 112–119, May 2018, doi: 10.1016/j.apergo.2018.01.006.

[169] S. G. Hart and L. E. Staveland, “Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research,” *Advances in psychology*, vol. 52, pp. 139–183, 1988.

[170] A. Erdogan and B. D. Argall, “Prediction of user preference over shared-control paradigms for a robotic wheelchair,” in *IEEE International Conference on Rehabilitation Robotics*, IEEE Computer Society, Aug. 2017, pp. 1106–1111. doi: 10.1109/ICORR.2017.8009397.

[171] R. W. Schutz and M. E. Gessaroli, “The analysis of repeated measures designs involving multiple dependent variables,” *Res Q Exerc Sport*, vol. 58, no. 2, pp. 132–149, 1987, doi: 10.1080/02701367.1987.10605437.

[172] Q. Li, W. Chen, and J. Wang, “Dynamic shared control for human-wheelchair cooperation,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 4278–4283. doi: 10.1109/ICRA.2011.5980055.

[173] A. Erdogan and B. D. Argall, “The effect of robotic wheelchair control paradigm and interface on user performance, effort and preference: An experimental assessment,” *Rob Auton Syst*, vol. 94, pp. 282–297, Aug. 2017, doi: 10.1016/j.robot.2017.04.013.

[174] D. Lee and P. Y. Li, “Passive bilateral control and tool dynamics rendering for nonlinear mechanical teleoperators,” *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 936–951, Oct. 2005, doi: 10.1109/TRO.2005.852259.

[175] A. Radford *et al.*, “Learning Transferable Visual Models From Natural Language Supervision,” Feb. 2021, [Online]. Available: http://arxiv.org/abs/2103.00020

[176] F. Yang, Q. Sun, H. Jin, A. Research, and Z. Zhou, “Superpixel Segmentation with Fully Convolutional Networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.*, 2020, pp. 13964–13973.

[177] G. Li and Y. Yu, “Deep Contrast Learning for Salient Object Detection,” in *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 478–487.

[178] A. Broad, T. Murphey, and B. Argall, “Highly Parallelized Data-driven MPC for Minimal Intervention Shared Control,” Jun. 2019, [Online]. Available: http://arxiv.org/abs/1906.02318

APPENDIX A: Pseudocode of the ground plane segmentation node

https://github.com/mszuyx/pc_groundPlaneSegmentation/tree/main

Algorithm 3.1

```
Inputs:   $\mathbf{P}_{input}$: the raw input points
          $\theta_x, \theta_y, \theta_z$: the Euler angles estimated by [134]
          $\sigma$: the size of the voxel grid
          $\delta$: the size of the box crop
          $k_{min}$: the threshold for radius outlier removal
          $r$: the radius for the radius search
          $N$: the RANSAC iteration budget, default = 3
          $T_{up}$: the upper bound of ground plane thickness
          $T_{low}$: the lower bound of ground plane thickness
Outputs:  $\mathbf{P}_{ground}$: the ground points
          $\mathbf{P}_{obstacle}$: the obstacle/non-ground points
          $\vec{\mathbf{n}}_{corrected}$: the normal vector of the ground plane
          $d$: the orthogonal distance to the ground plane
1.  // Pre-processing input point cloud
2.  $\mathbf{R} = \mathbf{rotationFromEulerAngles}(\theta_y, \theta_x, \theta_z)$
3.  $\mathbf{P}_{aligned} = \mathbf{orientationAlignment}(\mathbf{P}_{input}, \mathbf{R})$
4.  $\mathbf{P}_{cropped} = \mathbf{boxCrop}(\mathbf{P}_{aligned}, \delta)$
5.  $\mathbf{P}_{voxel} = \mathbf{voxelGridFilter}(\mathbf{P}_{cropped}, \sigma)$
6.  $\mathbf{P}_{pass} = \mathbf{passThroughFilter}(\mathbf{P}_{voxel}, \sigma)$
7.  $\mathbf{P}_{candidate} = \mathbf{radiusOutlierRemoval}(\mathbf{P}_{pass})$
8.  // Obtain initial point candidates (seed) for RANSAC
9.  $\mathbf{P}_{sorted} = \mathbf{sort}(\mathbf{P}_{candidate},\ y_{axis}\ descending\ order)$
10. $d = \mathbf{findMean}(first\ 50\%\ of\ \mathbf{P}_{sorted})$
11. For each point $\mathbf{p}_i$ in $\mathbf{P}_{sorted}$ do:
12.     If $\mathbf{p}_{i,y} > d$:
13.         $\mathbf{p}_i \rightarrow \mathbf{P}_{seed}$
14.     End if
15. End for
16. // Run RANSAC iteratively to refine ground plane detection
17. $\mathbf{P}_{ground} = \mathbf{P}_{seed}$
18. For $j = 0;\ j < N;\ j^{++}$:
19.     // Estimate plane parameters based on the current candidate points
20.     $\vec{\mathbf{n}}, d = \mathbf{estimatePlane}(\mathbf{P}_{ground})$
21.     // Update candidate points (inliers) based on the plane parameters
22.     $\mathbf{P}_{ground} \rightarrow clear()$
23.     If $j < N - 1$:
24.         For $each\ point\ \mathbf{p}_j\ in\ \mathbf{P}_{sorted}\ do$:
25.             $dist = \vec{\mathbf{n}}\ \mathbf{p}_j$
26.             If $dist > d - T_{up}$:
27.                 $\mathbf{p}_j \rightarrow \mathbf{P}_{ground}$
28.     // At the last iteration, push all the non-ground plane points to "obstacle points"
29.     Else:
30.         $\mathbf{P}_{obstacle} \rightarrow clear()$
31.         For $each\ point\ \mathbf{p}_k\ in\ \mathbf{P}_{cropped}\ do$:
32.             $dist = \vec{\mathbf{n}}\ \mathbf{p}_k$
```

```
33.                 If $dist > d - T_{up}$ && $dist < d + T_{low}$:
34.                     $\mathbf{p}_k \rightarrow \mathbf{P}_{ground}$
35.                 Else:
36.                     $\mathbf{p}_k \rightarrow \mathbf{P}_{obstacle}$
37.                 End if
38.             End for
39.         End if
40. End for
41. // Recover the original orientation of the plane's normal vector
42. $\vec{\mathbf{n}}_{corrected} = \mathbf{R}^{\mathrm{T}}\,\vec{\mathbf{n}}$
```

APPENDIX B: Pseudocode of the user command velocity mapping

```
Algorithm 4.1
Inputs:   $\theta_x^{usr}, \theta_y^{usr}, \theta_z^{usr}$: the raw wearable imu readings (roll, pitch, yaw)
          $M_x^{usr}, M_y^{usr}$: the raw moment readings from FSS
Params:   $\alpha$:  the command fusion factor [0,1]
          $Range_{pitch\_roll}$:  the maximum allowed angle range for pitch and roll (user leaning)
          $Range_{yaw}$:  the maximum allowed angle range for yaw (user twisting)
          $Range_{moment}$:  the maximum allowed moment range for FSS
Outputs:  $v_x^{usr}, v_y^{usr}, \omega_z^{usr}$: the user velocity commands
1.  $v_x^{usr}$= (1 - $\alpha$) * ($\theta_y^{usr}$/ $Range_{pitch\_roll}$) +  $\alpha$ * ($M_y^{usr}$/ $Range_{moment}$)
2.  $v_y^{usr}$= -1 * ((1 - $\alpha$)* ($\theta_x^{usr}$/ $Range_{pitch\_roll}$) +  $\alpha$ * ($M_x^{usr}$/ $Range_{moment}$)
3.  $\omega_z^{usr}$= $\theta_z^{usr}$/ $Range_{yaw}$
```

APPENDIX C: Pseudocode of the passive artificial potential field algorithm

https://github.com/mszuyx/PURE_SMC

Algorithm 4.2

Inputs: P_{comb}: the raw input points from all sensors
$v_x^{usr}, v_y^{usr}, \omega_z^{usr}$: the raw user velocity commands
$v_x^{fb}, v_y^{fb}, \omega_z^{fb}$: the robot odometry feedbacks

Params: δ_{pad}: the padding offset for the front and back points
δ_{thre}: the distance threshold for APF
w_{FB}: the weight for forward-backward APF force
w_{LR}: the weight for left-right APF force
η: the APF function coefficient
ζ: the speed tracking coefficient
ε: the trajectory smoothing coefficient

Outputs: $v_x^{cmd}, v_y^{cmd}, \omega_z^{cmd}$: the final velocity commands

```
1.  // Collect artificial potential force from point cloud measurements
2.  If $P_{comb}$.size() > 0 do:
3.      For each point $P_i$ in $P_{comb}$ do:
4.          // Compute the angle and range for the current measurement
5.          $\alpha_{curr} = \alpha_{min} + (\alpha_{incr} * i)$
6.          $\delta_{curr} = P_i$.range
7.          // Add some padding offset for points in the front and the back of robot
8.          If $abs(\sin(\alpha_{curr}) * \delta_{curr}) \leq 0.2$ do:
9.              $\delta_{curr} -= \delta_{pad}$
10.         End If
11.         // Start calculating potential force if the range is below a threshold
12.         If $\delta_{curr} \leq \delta_{thre}$ do:
13.             // Collecting potential forces in the forward-backward direction if driving towards this point
14.             If $abs(\sin(\alpha_{curr}) * \delta_{curr}) \leq 0.2$ &&
15.             $\cos(\alpha_{curr}) * v_x^{usr} > 0$ do:
16.                 $mag_x = -w_{FB} * \eta * \left(\frac{1.0}{\delta_{curr}} - \frac{1.0}{\delta_{thre}}\right)^2$
17.                 $f_x += \cos(\alpha_{curr}) * mag_x$
18.                 $cnt_x += 1$
19.             End If
20.             // Collecting potential forces in the left-right direction if driving toward this point
21.             If $abs(\cos(\alpha_{curr}) * \delta_{curr}) \leq 0.5$ &&
22.             $\sin(\alpha_{curr}) * v_y^{usr} > 0$ do:
23.                 $mag_y = -w_{LR} * \eta * \left(\frac{1.0}{\delta_{curr}} - \frac{1.0}{\delta_{thre}}\right)^2$
24.                 $f_y += \sin(\alpha_{curr}) * mag_y$
25.                 $cnt_y += 1$
26.             End If
27.         End If
28.     End For
29. End If
```

```
30. // Saturate the overall (averaged) potential forces
31. $f_x = \min(\max(\frac{f_x}{cnt_x+1.0}, -1.0), 1.0);$
32. $f_y = \min(\max(\frac{f_y}{cnt_y+1.0}, -1.0), 1.0);$
33. // Scaled and merge with the user command velocity to form ideal velocity commands
34. $v_x^{ideal} = v_x^{usr} + (f_x * abs(v_x^{usr}))$
35. $v_y^{ideal} = v_y^{usr} + (f_y * abs(v_y^{usr}))$
36. // Compute gains for velocity compensation
37. $\zeta_x = w_{FB} * \zeta;\ \zeta_y = w_{LR} * \zeta$
38. // Set these gains to zero if the robot is already running slower than the ideal velocity
39. **If** $v_x^{fb} * v_x^{ideal} > 0$ **&&** $\mathrm{abs}(v_x^{fb}) - \mathrm{abs}(v_x^{ideal}) < 0$ **do**:
40.         $\zeta_x = 0.0$
41. **End If**
42. **If** $v_y^{fb} * v_y^{ideal} > 0$ **&&** $\mathrm{abs}(v_y^{fb}) - \mathrm{abs}(v_y^{ideal}) < 0$ **do**:
43.         $\zeta_y = 0.0$
44. **End If**
45. // Compute compensated velocity commands
46. $v_x^{comp} = v_x^{ideal} + \zeta_x * (v_x^{ideal} - v_x^{fb})$
47. $v_y^{comp} = v_y^{ideal} + \zeta_y * (v_y^{ideal} - v_y^{fb})$
48. // Smooth (filter) the velocity commands
49. $v_x^{cmd} = v_x^{comp} + \varepsilon * (v_x^{old} - v_x^{comp})$
50. $v_y^{cmd} = v_y^{comp} + \varepsilon * (v_y^{old} - v_y^{comp})$
51. $\omega_z^{cmd} = \omega_z^{usr}$
52. // Update the variables for the filter
53. $v_x^{old} = v_x^{cmd};\ v_y^{old} = v_y^{cmd}$
54. **Return** $[v_x^{cmd}, v_y^{cmd}, \omega_z^{cmd}]$
```

APPENDIX D: Post-study Questionnaire for PURE Shared-Control Study

Shared-Control mode

	Very little	Little	Much	Very much
How intuitive was the shared control method? (i.e., how quickly were you able to learn and use shared control to ride the PURE without consciously thinking about the controls?)	☐	☐	☐	☐
Comments (Optional):				
	Very little	Little	Much	Very much
How much did the shared control method positively contribute to keeping you from having collisions while moving <u>forwards/backwards</u>? (i.e., Did the shared control make the test course feel easier?)	☐	☐	☐	☐
• What did you **like**? (Optional) • What did you **dislike**? (Optional)				
	Very little	Little	Much	Very much
How much did the shared control method positively contribute to keeping you from collisions while moving <u>left/right</u>? (i.e., Did the shared control make the test course feel easier?)	☐	☐	☐	☐

- What did you **like**? (Optional)
- What did you **dislike**? (Optional)

	Very little	Little	Much	Very much
How natural was the ride for you when using the shared control? (i.e., Did you feel disturbed or confused by the shared control?)	☐	☐	☐	☐

- What did you **like**? (Optional)
- What did you **dislike**? (Optional)

	Very little	Little	Much	Very much
How would you evaluate this shared control's aggressiveness? (i.e., Did shared control's acceleration/deceleration feel too abrupt?)	☐	☐	☐	☐

- What did you **like**? (Optional)
- What did you **dislike**? (Optional)

	Very little	Little	Much	Very much
How much do you think the haptic resistance helped you to adjust your motions to minimize collisions?	☐	☐	☐	☐

- What did you **like**? (Optional)
- What did you **dislike**? (Optional)

	Very little	Little	Much	Very much
Do you think the haptic resistance will be equally helpful during daily usage?	☐	☐	☐	☐
Comments (Optional):				
	Very little	Little	Much	Very much
How much do you think the audio alarm helps to adjust your motion and minimize collisions in the test courses?	☐	☐	☐	☐
• What did you **like**? (Optional) • What did you **dislike**? (Optional)				
	Very little	Little	Much	Very much
Do you think the audio alarm will be equally helpful during daily usage?	☐	☐	☐	☐
Comments (Optional):				
	Very little	Little	Much	Very much
How safe did you feel when riding on the robot using the shared-control?	☐	☐	☐	☐
• What did you **like**? (Optional) • What did you **dislike**? (Optional)				

For the future development purpose, what aspects of the shared-control method (in general) would you like to improve for better control of the PURE device?

Shared-Control vs. Test Course

For each of the following test courses, please specify which Shared-Control method do you prefer to use (select the **best** method for each row).

	No Shared-Control	**Shared-Control (Haptic + Alarm)**	**Shared-Control (Forward-Backward haptic only)**	**Shared-Control (Left-Right haptic only)**	**Shared-Control (Haptic only)**	**Shared-Control (Alarm only)**
S-Turn	☐	☐	☐	☐	☐	☐
Zigzag	☐	☐	☐	☐	☐	☐