

AMERICAN UNIVERSITY OF BEIRUT

HUMAN-ROBOT INTERACTION THROUGH
MIXED-INITIATIVE SUPERVISED AUTONOMY

by

ABBAS MOHAMAD SIDAQUI

A dissertation
submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
to the Department of Electrical and Computer Engineering
of the Maroun Semaan Faculty of Engineering and Architecture
at the American University of Beirut

Beirut, Lebanon
November 2021

AMERICAN UNIVERSITY OF BEIRUT

HUMAN-ROBOT INTERACTION THROUGH
MIXED-INITIATIVE SUPERVISED
AUTONOMY

by
ABBAS MOHAMAD SIDAOU

Approved by:

Dr. Naseem Daher, Assistant Professor

Advisor

Electrical and Computer Engineering, Mechanical Engineering



Dr. Daniel Asmar, Associate Professor

Co-Advisor

Mechanical Engineering



Dr. Riad Chedid, Professor

Chairperson of Committee

Electrical and Computer Engineering



Dr. Rouwaida Kanj, Associate Professor

Member of Committee

Electrical and Computer Engineering

Rouwaida
Kanj

Digitally signed by Rouwaida
Kanj
DN: cn=Rouwaida Kanj, o, ou,
email=rk105@aub.edu.lb, c=US
Date: 2022.02.04 15:53:36
+02'00'

Dr. Hadi Kanaan, Professor

Member of Committee

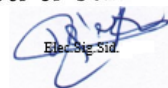
Electrical and Mechanical Engineering, Saint Joseph University of Beirut



Dr. Maher El Rafei, Associate Professor

Member of Committee

Electrical and Electronics Engineering, Lebanese University



Dr. Daniele Pucci, Researcher Tenure Track-PI

Member of Committee

Artificial and Mechanical Intelligence, Italian Institute of Technology



Date of thesis defense: November 4, 2021

AMERICAN UNIVERSITY OF BEIRUT

THESIS RELEASE FORM

Student Name: Sidaoui Abbas Mohamad
Last First Middle

I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of my thesis; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes

- As of the date of submission of my thesis
- After 1 year from the date of submission of my thesis .
- After 2 years from the date of submission of my thesis .
- After 3 years from the date of submission of my thesis .

Abbas Sidaoui



5 - Feb - 2022

Signature

Date

Acknowledgements

It is a genuine pleasure to express my thanks and gratitude to my advisors, Prof. Naseem Daher and Prof. Daniel Asmar for their continuous guidance and support throughout the PhD journey. You guys have been more than advisors, and I am proud to consider you as lifetime mentors. I thank my dear friend and colleague, Mohammad Kassem Zein, for his valuable contribution and support in implementing and testing the work presented in Chapter 3. I would also like to thank all the committee members, Prof. Chedid, Prof. Kanj, Prof. Kanaan, Prof. Rafei, and Prof. Pucci for their valuable time, feedback, and advice. Your comments and suggestions had a significant contribution in completing my thesis objectives. A special thanks to AUB University Research Board for funding and supporting this research. Finally, I would like to acknowledge Prof. Imad H. Elhajj for his input during the first three years of my PhD journey.

I consider myself lucky to have crossed paths with amazing friends and colleagues during my graduate journey at AUB. Noel Maalouf, you were my first friend at AUB, we published together, and travelled together. I wish you all the success in life. Mohammad Kassem Zein, our friendship started at VRL, became stronger through all the experiences we've shared, and I hope it lasts for life. Thank you for always pushing me to strive for more and dream big. Trishia Chemaly, thank you for showing me that no matter how big our dreams are, we can always achieve them through hard work and dedication. Sevag, Carine, Nadine, Maya, Waddah, and Zeina, thank you for all the good times we shared. No matter how far our lives take us apart I hope we stay close friends.

To my family, relatives, and childhood friends, thank you for always celebrating my achievements and having my back during tough times. Your unconditional love and support made me what I am today. Wassim Elhariri, my college best friend and business partner, thank you for all the success and failures we've shared together, for pushing me forward everyday, and for inspiring me to apply my research experience through entrepreneurship. I hope that, hand by hand, we can achieve our dreams in changing the world. Prof. Nadim Diab, you have been my academic role model since my Masters studies at RHU. I hope our paths cross again.

An Abstract of the Dissertation of

Abbas Mohamad Sidaoui for Doctor of Philosophy
Major: Electrical and Computer Engineering

Title: Human-Robot Interaction Through Mixed-Initiative Supervised Autonomy

Achieving full autonomy in robotic tasks is the ultimate goal for researchers and designers. This goal is motivated by the rapid advancements in automation algorithms and the decreasing cost of robotic hardware. However, full autonomy is still faced by challenges due to unplanned changes and unexpected situations that can occur anytime in the robot's workspace. Due to various factors, robots might not be able to fully perceive their environments, which increases the possibility of errors and decreases the efficiency of performing the tasks. To overcome such challenges, this thesis proposes to combine the unique and complementary traits of both humans and robots by allowing a human to supervise autonomy tasks and intervene when needed. This intervention is based on the human Situation Awareness (SA) and judgment, or upon requests that are triggered by the robot's 'self-confidence' (SC) in being able to successfully perform the task. In short, the aim of this research is to propose a Mixed-Initiative Supervised Autonomy (MISA) framework and validate its applicability in different robotics applications.

To evaluate the applicability and utility of MISA, it is implemented in three different proof-of-concept (POC) applications: grid-based collaborative simultaneous localization and mapping (SLAM), automated jigsaw puzzle reconstruction, and autonomous robot grasping. Augmented Reality (AR) (for SLAM) and two-dimensional graphical user interfaces (GUI) (for puzzle reconstruction and autonomous robot grasping) are custom-designed to enhance human SA and allow intuitive interaction between the human and the agents.

The superiority of the MISA framework is demonstrated through experiments. In SLAM, the superior maps produced by MISA preclude the need for post processing of any SLAM stock maps; furthermore, MISA reduces the required mapping time by approximately 50% versus the traditional baseline approach. In automated puzzle reconstruction, the MISA framework outperforms both fully autonomous solutions, as well as those resulting from on-demand human intervention prompted by the agent. In robot grasping, applying MISA results in a marked performance improvement by increasing the end-to-end success rate of the learning-based baseline approach from 35.0% to 87.6% .

Contents

Acknowledgements	v
Abstract	vi
List of Figures	ix
List of Tables	xiii
1 Introduction	1
1.1 Background and Rationale of the Thesis	1
1.2 Contributions	3
1.3 Thesis Related Work	4
2 Proposed Methodology	8
3 POC 1: MISA in collaborative SLAM	12
3.1 Introduction	12
3.2 Related Work	14
3.3 System Overview	15
3.3.1 Baseline Approach: OpenSLAM Gmapping	16
3.3.2 Applying MISA in Collaborative SLAM	17
3.4 Implementation and Experiments	22
3.4.1 Experimental Setup	22
3.4.2 Experimental Testing and Results	27
4 POC 2: MISA In Automated Puzzle Reconstruction	35

4.1	Introduction	35
4.2	Related Work	36
4.3	System Overview	38
4.3.1	Baseline Approach: Automated Puzzle Solver	38
4.3.2	Applying MISA in Automated Puzzle Reconstruction	40
4.4	Implementation and Experiments	43
4.4.1	Experimental Setup	43
4.4.2	Experimental Testing and Results	47
5	POC 3: MISA in Robot Grasping	52
5.1	Introduction	52
5.2	Related Work	54
5.3	System Overview	55
5.3.1	Baseline Approach: Autonomous Grasping	55
5.3.2	Applying MISA in Robot Grasping	56
5.4	Implementation and Experiments	59
5.4.1	Experimental Setup	59
5.4.2	Experimental Testing and Results	62
6	Conclusion and future work	66
A	Abbreviations	68
	Bibliography	69

List of Figures

1.1	The proposed MISA framework is motivated by needs from three areas: autonomous robots, human psychology, and human-robot interaction. The grey block in the middle shows an illustration of the proposed MISA system. The autonomous agent(s) perform the tasks autonomously and send the human supervisor their internal states, tasks results, and SC-based assistance requests through a user interface. The supervisor can assist the robot or intervene either directly or through the user interface.	2
2.1	Guideline to apply the proposed MISA framework in autonomous applications with four main modules.	9
2.2	Generic diagram of the proposed MISA framework with its main blocks and modules.	11
3.1	Conditions that affect map representation	13
3.2	The Map obtained from traditional SLAM (a) versus the map that reflects the real environment more accurately (b).	14
3.3	Flowchart representing the different modules in OpenSLAM gmapping baseline.	16
3.4	Block diagrams of MISA in collaborative SLAM.	19
3.5	Illustration of the <i>raycasting</i> unit in the proposed method. (a) shows the map and AR-HMD frames alongside the operator’s heading vector. (b) shows how the virtual rays are casted from the AR-HMD. (c) shows a top-view of the ray sets where each set is deviated from the other by θ_1 and from the heading vector by α . (d) shows the rays contained in each set, where each ray is rotated by angle β around the horizontal axis of the corresponding set; also shown are the nearest hit point and Horizontal_distance.	23
3.6	Flowchart presenting the implementation logic of MISA in collaborative SLAM.	24
3.7	The developed AR Interface. (a) shows the menu items, (b) shows the “tap” gesture, (c) and (d) show samples of map augmentation.	25
3.8	ARI components and the various reference frames.	26

3.9	Testing area at the American University of Beirut. The line in red represents the path that should be followed by the robot while conducting the experiments. Region 1 contains glass walls that are not detectable by the LiDAR, and Region 2 is an outdoor terrace.	27
3.10	Maps obtained using Set A (low laser range). The best map obtained by OpenSLAM gmapping is shown in (a), the worst map is shown in (b), (c) shows the map obtained using the proposed MISA framework, and (d) overlays the ground truth map on top of (c) to demonstrate the effectiveness of the MISA framework.	28
3.11	Maps obtained using Set B (medium laser range). The best map obtained by OpenSLAM gmapping is shown in (a), the worst map is shown in (b), (c) shows the map obtained using the proposed MISA framework, and (d) overlays the ground truth map on top of (c) to demonstrate the effectiveness of the MISA framework.	28
3.12	Maps obtained using Set C (high laser range). The best map obtained by OpenSLAM gmapping is shown in (a), the worst map is shown in (b), (c) shows the map obtained using the proposed MISA framework, and (d) overlays the ground truth map on top of (c) to demonstrate the effectiveness of the MISA framework.	29
3.13	The three scenarios discussed in N_{eff} evaluation. Red dots represent the particles.	30
3.14	Blueprint of the testing arena and photos of the experimental setup. Region I contains obstacles higher/lower than the LiDAR's detection plane, glass walls, and reflective objects. Region II is a corridor with poorly textured walls, and Region III has two glass walls facing each other. The robot starts in Region I and cannot traverse to Regions II and III since the exit is narrow.	31
3.15	Produced maps during the experiments: (a) shows the map obtained by the robot where it was not able to detect the objects numbered 1 to 5, which is overlaid on the blueprint in (b). (c) shows the automatic real-time updates performed by the HL on the augmented map (blue color) merged with the robot's map, which is overlaid on the blueprint in (d).	32
3.16	(a) Map produced by the AR-HMD Map Builder module in Region II . (b) the merged map of both Region II and Region III ; the red circle shows a mapping error that occurred because the HL failed to perform tracking in this location.	33
3.17	The final map produced by the proposed collaborative augmented SLAM system.	33
3.18	Final map with human augmented edits (a), and without any human edits (b)	34
4.1	Example of an image puzzle with square pieces of unknown orientations and locations (right), and the result of successful reconstruction of the puzzle pieces (left).	36
4.2	Flowchart representing the different modules in Automated Puzzle Solver baseline.	38

4.3	Block diagrams of MISA in automated puzzle reconstruction.	42
4.4	Flowchart presenting the implementation logic of MISA in automated puzzle reconstruction.	44
4.5	The MISA-based UI developed for automated puzzle reconstruction: Reconstruction phase.	46
4.6	The MISA-based UI developed for automated puzzle reconstruction: Trimming phase. The suggested trim frame is visualized in red over the resultant cluster image.	47
4.7	Demonstration of the cluster merging process. The figure shows how clusters are being merged (rotation and translation) to form a final combined cluster image.	47
4.8	Two sample results of puzzle reconstruction through the fully autonomous baseline approach.	48
4.9	The effect of the supervisor's assistance when the system's confidence drops. (a) shows the proposed merge with the corresponding clusters, (b) shows how the clusters are growing separately after the supervisor declines the proposed merge, (c) shows the final reconstructed cluster, and (d) shows how the reconstruction would end if the human had approved the proposed merge.	49
4.10	The effect of the supervisor's SA-based assistance. (a) shows a cluster with a misplaced piece, (b) shows how the final reconstruction looks like if the human did not select the misplaced piece to be deleted, and (c) shows the final reconstructed cluster considering the human assistance in detecting the misplaced piece.	50
4.11	Visual results of sample experiments. The left column shows results using the baseline approach, the middle column shows the results obtained through running the proposed system but the supervisor was allowed only to approve/decline merge options upon request, and the column to the right shows the results obtained by the proposed MISA in automated puzzle reconstruction.	51
5.1	Sub-tasks of a robotic grasping task. Object detection and grasp pose identification (top) belong to the <i>sensing</i> primitive of the task, arm trajectory planning (middle) belongs to the <i>planning</i> primitive, and plan execution (bottom) belongs to the <i>acting</i> primitive.	53
5.2	Flowchart representing the different modules in Autonomous robot grasping baseline.	55
5.3	Block diagrams of MISA in robot grasping.	58
5.4	Flowchart presenting the implementation logic of MISA in robot grasping.	60

5.5	The GUI developed for MISA in robot grasping. The snapshot shows an example of a wrongly proposed pose (rectangle with green and red sides), after the supervisor had pressed EDIT To adjust the proposed grasp pose, the supervisor clicks on the correct object (dry erase marker) location in the image and edits the pose, shown by the overlaid red line. The orientation of the pose can be adjusted through the Rotate Left/Right buttons, which change the orientation of the pose by 5° increments/decrements.	61
5.6	The 10 different backgrounds used in the MISA in robot grasping experiments.	63
5.7	Sample results visualizing the dispersion of the estimated grasp poses in cases of: successful grasp pose (lower row), and wrong grasp pose (upper row). . . .	64
5.8	A sample result of the execution error experiments. Red dots represents grasp locations that were selected by the supervisor through the GUI, these dots were drawn on the robot's work space at the beginning of each experiment. Blue dots represent the actual executed grasp location, these dots were obtained through fixing a blue marker at the center of the gripper. Green lines visualize the offset between the target poses and actual poses.	65

List of Tables

3.1	MISA attributes in collaborative SLAM.	17
3.2	Experimentation sets and their parameters.	27
4.1	MISA attributes in automated puzzle reconstruction.	41
4.2	Comparison of experimental results from MISA in automated puzzle reconstruction, and four state-of-the-art approaches.	51
5.1	MISA attributes in robotic grasping applications.	57
5.2	Experimental Results for MISA in robot grasping.	65

Dedication

In memory of my father.

Though you left this world early, you're in every chapter of my journey. I never had the chance to tell how thankful I am for what you've did to put me on the path towards success. I promise, I will always make you proud when looking down at me.

*To my **mother**, the strongest woman I know. You planted a seed, and got to watch it grow with love; now your seed is blossoming. Yesterday, today, and forever, I owe my success to you, your sacrifices, and your prayers. I love you Mama.*

*To my uncle, **Ahmad**. You believed in me since day one, treated me as your son, and helped me overcome every challenge and obstacle along the way. I cannot thank you enough for all the support and love you have given me.*

*To **Hasan and Mariam**. I never would have made it here without your encouragement and support. I promise, I will always be by your side.*

*To my grandparents, **Ali and Naila**. Thanks for doing anything and everything to help me be where I am today.*

*To **Aya**. No words can describe how grateful I am to have you in my life. You've always been my go-to person in low moments, and the first one I share my achievements and success with. Thanks for your unconditional support and love.*

*To my best friends, **Dina, Wassim, Hadi, and Mohammad**. You had my back through it all; I couldn't have asked for better friends.*

*To **Bassam**. You've been an advisor, friend, and big brother. Thank you for always pushing me to continue to strive to achieve more and be a better version of myself everyday.*

*To my advisors, **Daniel and Naseem**. I will never be able to express my sincere appreciation to both of you. I will never forget that in the lowest moment of this journey, you were there to hold my hand and put me back on track. Thank you for not giving up on me, you both have been pillars of support and guidance.*

Chapter 1

Introduction

1.1 Background and Rationale of the Thesis

When automation was first introduced, its use was limited to well-structured and controlled environments in which each automated agent performed a very specific task [1]. Nowadays, with the rapid advancements in automation and the decreasing cost of hardware, robots are being used to automate more tasks in unstructured and dynamic environments. However, as the tasks become more generic and the environments become more unstructured, full autonomy is challenged by the limited perception, cognition, and execution capabilities of robots [2]. Furthermore, when robots operate in full autonomy, the noise in the real world increases the possibility of making mistakes [3], which could be due to certain decisions or situations that the robot is uncertain about, or due to limitations in the software/hardware abilities. In some cases, the robot is able to flag that there might be a problem (*e.g.*, high uncertainty in object detection or inability to reach a manipulation goal); while in other cases, the robot may not even be aware that errors exist (*e.g.*, LiDAR not detecting a transparent obstacle).

In recent years, human-robot interaction has been gaining more attention from robotics researchers [4]. First, robots are being deployed to automate more tasks in close proximity with humans [5, 6]; second, the challenges facing autonomous robots may not perplex humans, owing to their superior cognition, reasoning, and problem-solving skills. This has lead researchers in the HRI community to include humans in the decision-making loop to assist and provide help, as long as autonomous systems are not fully capable of handling all types of contingencies [7–9]. Including a human in the loop can mitigate certain challenges and limitations facing autonomy, by complementing the strength, endurance, productivity, and precision that a robot offers on one hand, with the cognition, flexibility, and problem-solving abilities of humans on the other. In fact, studies have shown that combining human and robots skills can increase the capabilities, reliability, and robustness of robots [10]. Thus, creating successful approaches to human-robot interaction and human-autonomy teaming is considered crucial for developing effective autonomous systems [11, 12], and deploying robots outside of controlled and structured environments [13, 14].

One approach to human-autonomy teaming is having a human who helps the autonomous

agents when necessary. In fact, seeking and providing help to overcome challenges in performing tasks has been studied by human-psychology researchers for many decades. When uncertain about a situation, people tend to gather missing information, assess different alternatives, and avoid mistakes by seeking help from more knowledgeable persons [15]; for example, sometimes junior developers consult senior colleagues to help in detecting hidden bugs in a malfunctioning code, or students ask teachers to check an assignment answer that they are not certain about.

Help is also considered indispensable to solve problems and achieve goals that are beyond a person’s capabilities [15]. When aware of their own limitations, people seek help to overcome their disadvantaged situations; for instance, a shorter employee may ask a taller colleague to hand them an object that is placed at a high shelf. In some cases, people might not be aware of the situation nor their limitations; here, another person might have to intervene and provide help to avoid problems or unwanted outcomes. For example, a supervisor at work may step in to correct a mistake that an intern is not aware of, or when a passer-by helps a blind person avoid an obstacle on the sidewalk. Moreover, exchanging help has been shown to enhance the performance and effectiveness of human teams [16–18], especially in teams with members having different backgrounds, experiences, and skill sets.

With the above in mind, Figure 1.1 summarizes the motivation behind the proposed MISA framework from various perspectives. First, the human psychology literature indicates that although humans think of themselves as being autonomous [19], they always seek and receive help to achieve their goals and overcome uncertain situations and limited capabilities. Second, robotics literature states that autonomous robots are still facing challenges due to limitations in hardware, software, and cognitive abilities. These challenges lead to mistakes and problems; some of which can be detected and flagged by the robot, while others may go undetected by the robot. Third, research in HRI and human-autonomy interaction illustrates that having a human to interact with and assist autonomous agents is a must.

From the three different lines of research, it can be concluded that there is a need for a new HRI framework that takes advantage of the complementary skills of humans and robots on one hand, and adopts the general idea of help-seeking and helping behaviors from human-

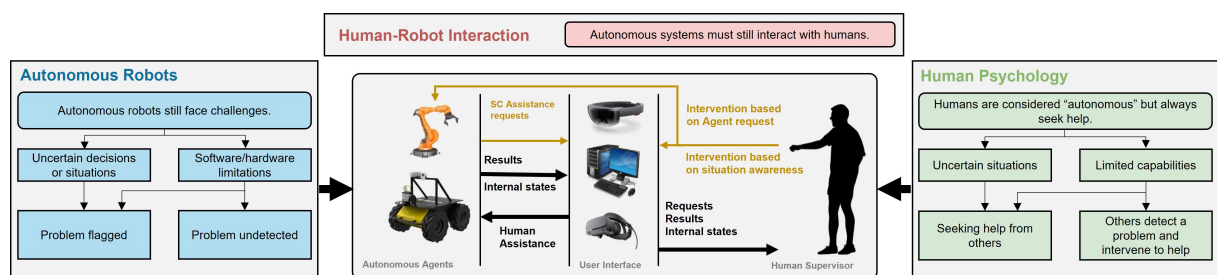


Figure 1.1: The proposed MISA framework is motivated by needs from three areas: autonomous robots, human psychology, and human-robot interaction. The grey block in the middle shows an illustration of the proposed MISA system. The autonomous agent(s) perform the tasks autonomously and send the human supervisor their internal states, tasks results, and SC-based assistance requests through a user interface. The supervisor can assist the robot or intervene either directly or through the user interface.

human interaction on the other hand. While state-of-the-art frameworks focus mainly on how to allocate and exchange tasks between humans and robots, there is no generalized framework, as far as I know, that considers including a human in the loop to support autonomous systems in which a human cannot completely take over the tasks execution. For this reason, MISA is proposed as a framework for HRI that allows a human to supervise autonomous tasks and provide help to avoid mistakes during autonomy, without a complete takeover of these tasks. Human assistance can be requested by the robot in tasks where it can reason about its SC. In tasks where the robot is not aware of its limitations, or does not have the means to measure its SC, human assistance is based on the human's SA. Moreover, the type, timing, and extent of interaction between the human supervisor and the robot are governed within MISA through a MISA-based user interface.

1.2 Contributions

The main contribution of this thesis is to propose and validate a *Mixed-Initiative Supervised Autonomy framework* for Human-Robot Interaction, which leverages the robot's SC and the human's SA in order to improve the performance of automated tasks and mitigate full autonomy limitations. Through the proposed framework, this thesis provides a guideline and generic flowchart to help researchers apply MISA in any automated application. To validate the proposed framework, it is tailored and applied to three different POC applications, as follows:

1. MISA in collaborative SLAM:

The proposed application of MISA in collaborative SLAM allows three co-located heterogeneous types of agents (robot, AR head mount device AR-HMD, and human) to contribute to the grid-based SLAM process.

Whenever an agent's self-confidence drops, it prompts the human supervisor to approve or correct its pose estimation. In addition, the human can view the map being built superposed on top of the real environment through a custom-developed AR interface, and apply edits to this map in real-time based on his/her SA. Moreover, when the robot is unable to reach a certain navigation goal, it prompts the human to assist in mapping the area corresponding to this goal, either through editing the map him/herself or through activating the AR-HMD collaborative mapping feature.

2. MISA in automated puzzle reconstruction:

This application of MISA in automated puzzle reconstruction addresses the challenges presented in solving image puzzles with square pieces of unknown locations and orientations, using the greedy approach.

Whenever the agent's self-confidence drops, it prompts the human supervisor to approve/decline the merging of puzzle pieces through a custom-developed GUI. Furthermore, the GUI allows the human supervisor to monitor the puzzle reconstruction results, and intervene to correct mismatched pieces and the final puzzle frame based on his/her SA.

3. MISA in robot grasping:

The application of MISA in robot grasping aims to address the challenges and limitations affecting end-to-end, learning-based robot grasping.

Whenever the agent’s self-confidence drops, it requests the supervisor’s assistance in approving/correcting the proposed grasp pose through a custom-developed GUI. In addition, the agent notifies the supervisor when the target object is out-of-reach. The human supervisor can also assist the robot in eliminating execution errors through defining the arm-camera calibration offset based on his/her SA.

Experimental validation provides empirical evidence that applying the proposed MISA framework outperforms full autonomy in the three POC applications. Moreover, this work presents the following additional contributions:

- Designing and implementing an intuitive MISA-based AR interface for SLAM applications. This interface allows a human supervisor to view the robot’s map augmented on top of the real environment and interact with it in real-time. The proposed AR interface could be easily integrated in any grid-based SLAM technique.
- Designing and implementing two intuitive MISA-based GUIs for HRI in puzzle reconstruction and robot grasping. The GUIs allows the human supervisor to monitor the autonomous agent, intervene when needed, and/or provide assistance upon requests.
- Defining self-confidence metrics for: robot pose estimation, clusters merging in puzzle reconstruction, and grasp pose estimation in data-driven robot grasping.
- Modeling of a virtual AR sensor to produce two dimensional (2D) occupancy grid maps from the mesh built by an AR-HMD.
- Online solution that allows merging of 2D occupancy grid maps produced by a robot, AR-HMD, and human supervisor.

1.3 Thesis Related Work

Researchers in the HRI community have been advocating bringing back humans to the automation loop through different approaches and methods. Below is a discussion of the most common approaches.

The term ‘Mixed-initiative (MI)’ was first introduced in 1970 in the context of computer-assisted instruction (CAI) systems [20]. This was followed by several definitions in the domain of Human-Computer Interaction (HCI) [21–23], where MI was proposed as a method to enable both agents and humans to contribute in planning and problem-solving based on their knowledge and skills. Motivated by the previous definitions, Jiang and Arkin [24] presented the first clear definition for mixed-initiative human-robot interaction (MI-HRI) as follows:

“A collaboration strategy for human-robot teams where humans and robots opportunistically seize (relinquish) initiative from (to) each other as a mission is being executed, where initiative is an element of the mission that can range from low-level motion control of the robot to high-level specification of mission goals, and the initiative is mixed only when each member is authorized to intervene and seize control of it.”

According to this definition, the term ‘initiative’ in MI-HRI systems refers to a task of the mission that could be performed and seized by either the robot or the human. This means that both agents (robot and human) may have tasks to perform within a mission, and both can completely take over tasks from each other. Unlike MI-HRI, in MISA the term ‘initiative’ is used to refer to ‘initiating the human assistance,’ and initiatives are considered mixed since this assistance is initiated either by the autonomous agent’s request or by the human supervisor. Another difference between MISA and MI-HRI is that all tasks within a MISA system are performed by the robot; albeit, human assistance is requested/provided to overcome certain challenges and limitations.

Another method that enables the contribution of both humans and agents in an automated task is ‘adjustable autonomy’ that was defined by Zieba *et al.* [25] as:

“The property of an autonomous system to change its level of autonomy while the system operates. The human operator, another system, or the autonomous system itself can adjust the autonomy level.”

Thus, a system with adjustable autonomy allows switching autonomy states between two extremes: full autonomy and manual control. This adjustment enables the user to help the system avoid some errors and failures through assisting in some tasks or taking control of the system in certain conditions. In literature, there are different approaches to applying adjustable autonomy, some of which are: teamwork-centered, supervised, degree-based, level-based, sliding, policy-based, and predictive approaches [26]. These approaches define different rules and conditions that govern the autonomy level change in a system. However, the common factor among them is that whenever the autonomy is adjusted, some tasks are re-allocated between the human and the agents, and certain conditions have to be met in order to re-adjust the system back to the full-autonomy state. In the hereby proposed framework, the tasks are not re-allocated between the human and the agent but rather the human is acting as a supervisor to assist the autonomous agent and intervene when needed. Since no additional tasks are re-allocated to the human in MISA, he/she can supervise multiple agents at once with, theoretically, less mental and physical strain as compared to re-assigning tasks to the human himself.

It is important to mention that MISA is not proposed to replace these previous approaches, but rather to expand the spectrum of HRI to autonomous systems where it is not feasible for a human to ‘completely takeover’ the mission tasks. Examples of such tasks are real-time localization of a mobile robot, motion planning of multiple degrees-of-freedom robotic manipulators, path planning of racing autonomous ground/aerial vehicles, among others. In fact, every HRI approach has use-cases where it performs better in. Adjustable autonomy and MI-HRI focus mainly on how to allocate functions between the robot and the human to enhance the performance of a collaborative mission. However, MISA is proposed to enhance the performance,

decrease uncertainty, and extend capabilities of robots that are initially fully autonomous by including a human in the loop to assist the robot without taking over any task execution.

Setting the Level of Robot Autonomy (LORA) was also considered by researchers as a method for HRI. The concept of *levels of autonomy* was first introduced by Sheridan and Verplank [27] in 1978, where 10 levels of machine autonomy were defined, ranging from “the human executes all actions” to “the computer executes all actions.” This was a general taxonomy that did not consider the inputs of the action functions such as information acquisition and analysis. In 2000, Parasuraman *et al.* [28] extended the taxonomy to be applied in 4 stages of automation: (1) information acquisition, (2) information analysis, (3) decision and action selection, and (4) action implementation. Most recently, and being inspired by the previous works, Beer *et al.* [29] proposed a new taxonomy for levels of autonomy that considers the perspective of HRI and the roles of both the human and the robot. This taxonomy consists of 10 Levels of Robot Autonomy (LORA) that are defined based on the allocation of functions between robot and human in each of the **sense**, **plan**, and **act** primitives. Among the continuum, there are two levels of relevance to the work in this thesis:

- Shared Control with Human Initiative: *“The robot autonomously senses the environment, develops plans/goals, and implements actions. However, the human monitors the robot’s progress, and may intervene and influence the robot with new goals/plans if the robot is having difficulty.”*
- Shared Control with Robot Initiative: *“Robot performs all aspects of the task (sense, plan, act). If the robot encounters difficulty, it can prompt the human for assistance in setting new goals/plans.”*

Although these two levels allow for human assistance, it is limited to the sense and plan primitives. In addition, the continuum does not contain a level which allows interaction based on both human judgment and requests from the robot. To mitigate the limitations of these two LORAs, the MISA framework proposes a new level labelled as ‘Mixed-Initiative Supervised Autonomy (MISA) Level,’ stated as follows:

The robot performs all three primitives (sense, plan, act) autonomously, but can ask for human assistance based on a pre-defined internal state. In addition, the human can intervene to influence the output of any primitive when s/he finds it necessary.

Moreover, and unlike LORA, MISA provides a clear framework for implementation.

In addition to the general approaches discussed above, several application-oriented methods were presented in which human help is exploited to overcome autonomy challenges and limitations. Certain researchers focused on the idea of having the human decide when to assist the robot or change its autonomy level. For instance, systems that allow the user to switch autonomy level in a navigation task were proposed in [30, 31]. Although these methods allow the human to switch between autonomy levels, the definition of the levels and the tasks allocated to the human in each level are different and mission-specific. In [32], a human operator helps the robot in enhancing its semantic knowledge by tagging objects and rooms via voice commands and a laser pointer. In such human-initiated assistance approaches, helping the robot and/or

changing the autonomy level is solely based on the human judgment; thus, the human has to closely monitor the performance in order to avoid failures or decaying performance, which results in additional mental/physical workload.

Other researchers investigated approaches where interaction and human help is based on the robot's request only. In [10], a framework that allows the robot to request human help based on the value of information (VOI) theory was proposed. In this framework, the human is requested to help the robot by providing perceptual input to overcome autonomy challenges. In [33], time and cost measures, instead of VOI, were used to decide on the need for interaction with the human. In [34] and [35], human assistance is requested whenever the robot faces difficulties or unplanned situations while performing navigation tasks. Instead of requesting human help in decision-making, [36] proposed a mechanism that allows the robot to switch between different autonomy levels based on the 'robot's self-confidence' and the modeled 'human trust in the robot.' In this approach, the robot decreases its autonomy level, and thus increases its dependency on the human operator, whenever its confidence in accurately performing the task drops. In these robot-initiated assistance approaches, the human interaction is requested either directly through an interface or by the agent changing its autonomy level; thus, any failure to detect the need for human assistance would lead to performance deterioration and potentially catastrophic consequences.

Other methods considered humans assisting robots based on both human judgment and robots requests. In [37], an interactive human-robot semantic sensing framework was proposed. Through this framework, the robot can pose questions to the operator, who acts as a human sensor, in order to increase its semantic knowledge or assure its observations in a target search mission. Moreover, the human operator can intervene at any moment to provide useful information to the robot. Here, human assistance is limited to the sensing primitive. In the context of adjustable autonomy, [38] proposed a remotely operated robot that changes its navigation autonomy level when a degradation in performance is detected. In addition, operators can choose to change autonomy levels based on their own judgment. In [39], a human can influence the robot autonomy via direct control or task assignment, albeit the proposed controller allows the robot to take actions in case of human error. Some researchers refer to MI as the ability of the robot to assist in or takeover the tasks that are initially performed by the human [40], or combine human inputs with automation recommendations to enhance performance [41]. Finally, in [42] the concept of MI is used to decide what task to be performed by which agent in a human-robot team. In contrast to such systems where the human is a teammate who performs tasks, the MISA framework proposes the human as a supervisor who intervenes to assist the autonomous agent as needed: either on-demand or based on SA.

Chapter 2

Proposed Methodology

In the proposed MISA framework, HRI systems consisting of a human supervisor and autonomous agent(s) are considered, where MISA is defined as:

”A Mixed-Initiative Supervised Autonomy framework that allows a robot to sense, plan, and act autonomously, but can ask for human assistance based on its internal self-confidence. In addition, the human supervisor can intervene and influence the robot sensing, planning, and acting at any given moment based on his/her SA.”

According to this definition, the robot operates autonomously but the human supervisor is allowed to help in reducing the robot’s uncertainty and increasing its capabilities through SC-based and SA-based assistance. The application of MISA in HRI systems relies on the following assumptions:

1. The human is aware of the application being performed and its automation limitations.
2. The autonomous agent has a defined self-confidence based mechanism that it can use to prompt the supervisor for help. Through this mechanism, the robot requests the human supervisor to approve, decline, or edit the decisions it is not certain about. Moreover, this mechanism is used to inform the supervisor if the robot is unable to perform a task.
3. When the autonomous agent seeks help, the human is both ready to assist and capable of providing the needed assistance.
4. The MISA system is equipped with an intuitive communication channel (MISA-based interface) that allows the agent to ask for help and the human to assist and intervene in the tasks.
5. The human has adequate expertise and SA, supported by the MISA-based interface, to intervene in the supervised task.

The proposed framework consists of a guideline and block diagram to help researchers apply MISA in any automated system. Figure 2.1 presents the MISA guideline, which is discussed below.

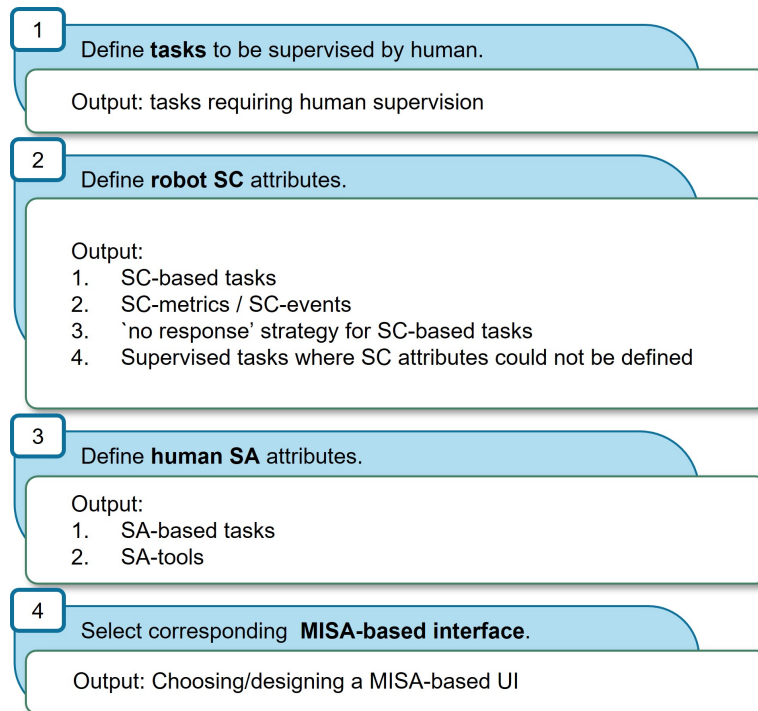


Figure 2.1: Guideline to apply the proposed MISA framework in autonomous applications with four main modules.

1. Define tasks to be supervised by human: the first step that is necessary to implement MISA is for one to define the tasks in which human supervision is needed based on the challenges and limitations of autonomy in the given system/application. These tasks depend on the robot’s hardware/software, mission type, and the environment in which the robot is operating.

To understand the challenges and limitations of autonomy in the given system/application, the MISA system designer can run the application in full autonomy mode, and compare the obtained performance and results with those that are expected. Another way is to find common challenges and limitations for similar systems/applications from the literature. After that, the MISA system designer has to define the tasks affected by these challenges and limitations, in which a human could help. For example, some challenges that face an autonomous mobile robot in a SLAM application can be present in obstacles that the robot cannot detect; these challenges affect the mapping task. while other challenges that affect the picking task in a pick-and-place application can be present in objects that are out of the manipulator’s reach.

The output of this step is a list of all the tasks that should be supervised by the human. It is important to mention here that not all tasks in the autonomous application need to be supervised, as some tasks result in better performance when being fully autonomous.

2. Define the robot’s self-confidence attributes: this step is concerned with the SC mechanism used by the robot to reason about when to ask the human for help and what happens if the human does not respond. Depending on the type of task, available internal-states of the robot, and full-autonomy experiments, the robot SC state could be toggled between ”confident” and ”not-confident” either through (1) detecting certain events (SC-events), and/or through (2)

monitoring certain metric/s (SC-metrics) within the supervised task and comparing them to a confidence threshold that is defined experimentally. Examples of events that could be used to set robot SC to "not-confident" are the loss of sensor data, inability to reach the goal, failure in generating motion plans; while examples of monitored metrics are performance, battery level, dispersion of particles in a particle filter, among others.

Depending on the task and how critical it is to get human assistance once requested, a 'no-response' strategy should be defined. Examples of such strategy are: wait, request again, wait for defined time before considering the decision approved/declined, put request on hold while proceeding with another task, among others.

For each task to be supervised, the MISA system designer has to determine whether or not it is possible to define events/metrics to be used to set the SC state. Thus, the outputs of this step include (1) SC-based tasks: tasks of which human assistance is requested through SC mechanism, (2) SC metrics and/or SC events, (3) 'no response' strategy for each SC-based task, and (4) supervised tasks that human assistance could not be requested through a SC mechanism.

3. Define the human situation awareness attributes: this step is crucial for supervised tasks in which robot SC attributes could not be defined. In such tasks, the assistance has to depend solely on the human SA. Given that on one hand the human is aware of the system's limitation, and on the other is endowed with superior situational and contextual awareness, s/he can determine when it is best to intervene at a task to attain better results where SC attributes could not be defined. However, the MISA system designer should define the needed tools to enhance human SA and help them better assess the situation. Examples of SA tools include showing certain results of the performed task, and communicating the status of the task planning/execution, to name a few.

The outputs of this step are (1) SA-based tasks: tasks in which assistance depends on human SA, (2) tools that would enhance human SA.

4. Choose the corresponding MISA-based user interface: after defining the tasks to be supervised and the SC/SA attributes, a major challenge lies in choosing the proper MISA-based UI that facilitates interaction between the human and the agent. The interface, along with the information to be communicated highly depend on the tasked mission, the hardware/software that are used, and the tasks to be supervised. Thus, the MISA-based UI must be well-designed in order to (1) help both agents communicate assistance requests/responses and human interventions efficiently, (2) maximize the human awareness through applying the SA tools, and (3) govern the type of intervention/assistance so that it does not negatively influence the entire process. MISA-based UIs include, but are not limited to graphical user interfaces, AR interfaces, projected lights, vocals, gestures, etc.

The desired output from this step is choosing/designing a MISA-based UI.

In addition to the discussed guideline, Figure 2.2 presents a block diagram to help apply MISA in any automated system. This diagram consists of two main blocks: the Autonomous System block and the MISA Coordinator (MISAC) block. The autonomous system block contains modules that correspond to the tasks performed by the autonomous agent; as mentioned in the guideline, some of these tasks are supervised, while others are fully autonomous.

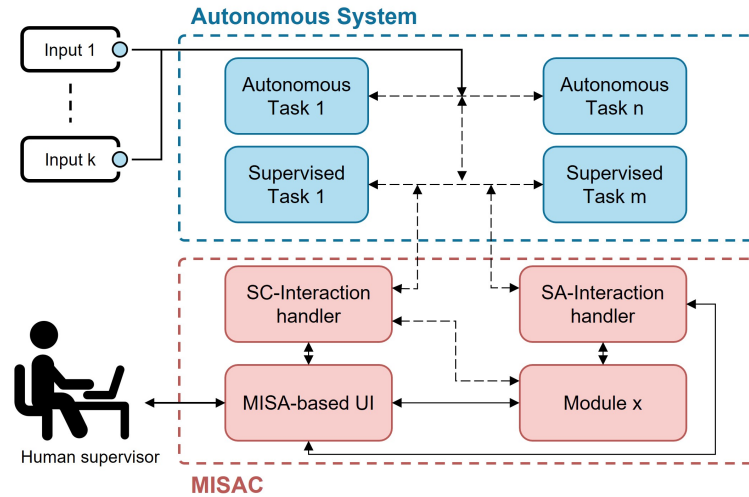


Figure 2.2: Generic diagram of the proposed MISA framework with its main blocks and modules.

The MISAC block contains three main modules: SC-Interaction handler, SA-Interaction handler, and the MISA-based UI. Interaction based on robot SC is handled through the SC-Interaction handler. This module monitors the SC-events/metrics defined through the ‘robot SC attributes’ and sends assistance requests to the human supervisor through the MISA-based UI. When the human responds to the request, SC-Interaction handler performs the needed operations and sends back the results to the corresponding supervised task. If no response from the human is received, the SC-Interaction handler applies the no-response strategy along with the needed operations.

Since an operator monitors the system through the SA tools provided in the MISA-based UI, s/he can intervene at any time based on SA. The SA-Interaction handler receives the human input/intervention, applies the needed operations, and sends the results to the corresponding supervised task. It is important to mention that both SC- and SA-Interaction handlers can be connected to several supervised tasks; and thus could handle different types of interactions and operations. Moreover, the MISAC block could contain other modules that handle some operations not directly related to the SC- SA- interaction handlers.

To validate the generality and utility of the proposed framework, it is applied to three autonomous applications as previously mentioned. For each application, the tasks to be supervised are defined through analyzing the performance of the fully autonomous system, and spotting common challenges from the literature. Then the robot SC attributes and the human SA attributes are defined through experimentation for each system separately. As for the MISA-based UI, custom-designed AR interfaces and 2D graphical interfaces are developed to enhance the human’s awareness and communicate information and requests efficiently.

Chapter 3

POC 1: MISA in collaborative SLAM

3.1 Introduction

Simultaneous localization and mapping (SLAM) plays an important role in a number of indoor and outdoor applications. For instance, industrial mobile robots heavily rely on SLAM to accomplish their tasks while autonomously navigating their environments. Moreover, SLAM is used for tasks such as autonomous aerial surveillance [43, 44], underwater reef monitoring [45, 46], 3D reconstruction of underwater morphologies [47], and underground mine exploration [48, 49]. SLAM can be implemented in 2D or 3D using sensors such as LiDARs or cameras. Even though the proposed framework is applied to 2D SLAM, the proposed method is extendable to 3D.

A main challenge of SLAM is that its estimates are affected by sensory errors, modeling errors, localization errors, and map representation errors. Localization errors could be caused by hardware faults, wheels slippage, inaccurate modeling, inaccurate map, and other causes related to the environment where the mission is being performed. When considering map representation errors, it is important to distinguish between two types of errors: those affecting the localization, and those affecting the real environment's representation. Noisy range measurements, changes in the environment being mapped, and the presence of dynamic obstacles are possible causes of map errors that could affect localization. Figure 3.1 shows an example of certain conditions that lead to errors affecting the real environment representation. Objects of **Type 1** are obstacles that are lower or higher than the LiDAR and thus cannot be detected. **Type 2** objects are transparent obstacles, such as glass panels. **Type 3** objects represent un-navigable areas such as stairs or holes in the ground. Although not detecting these types of obstacles has little impact on the localization part of SLAM, the robot might consider such areas as acceptable for navigation, and thus might plan a path through them. Finally, **Type 4** objects are obstacles that the user might want not to include in the output map. Figure 3.2 shows a comparison between the map obtained by a robot running SLAM in a sample environment (Figure 3.2a), and a map that provides more accurate representation of the real environment, in which the four mentioned error types are taken into account (Figure 3.2b). This sample scenario emphasizes the importance of evaluating and assessing the quality of the robot's map, especially when the

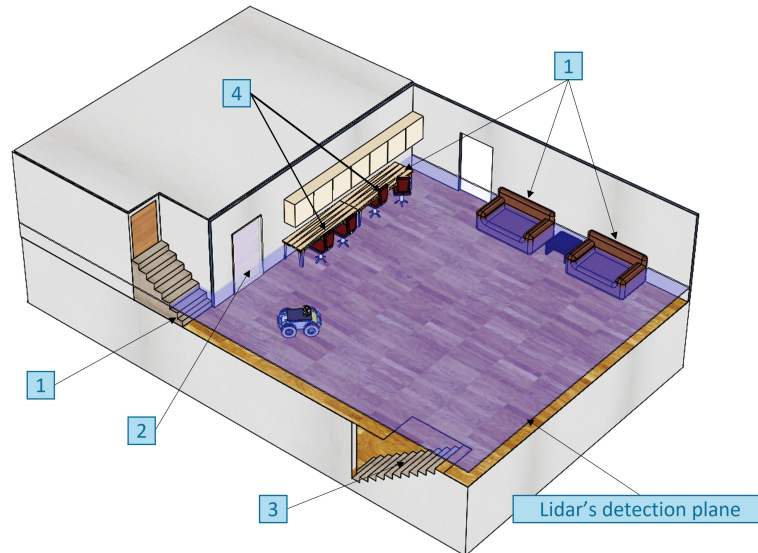


Figure 3.1: Conditions that affect map representation

robot is operating in an environment that is shared with humans.

To mitigate these challenges, most maps today are post-processed, however this is a long and dull process that is difficult to perform after the navigation is complete. Even worse, map edits usually require several trips to the mapped site for the sake of re-measuring and correcting dimensions, as well as marking forbidden areas that were not flagged by the robot during navigation. A much better approaches can entail having a human in-the-loop to assist the robot in SLAM, or even having multiple agents that contribute to the mapping process. Through collaborative mapping, global maps could be built by merging sub-maps obtained by different agents [50], or by directly fusing sensory data from these agents [51]. Moreover, one agent can localize itself in a map built by another [52]. In all cases, better mapping accuracy can be achieved through redundancy or through the fact that one agent can depend on the complementary beneficial properties of the other (being an autonomous agent or a human supervisor). Building on these concepts, the MISA framework is applied to collaborative SLAM.

Recent advances in AR have widened the HRI and human-in-the-loop scenarios where the operator can program [53], share control [54], or collaborate with a robot to enhance its performance [55]. Moreover, the latest AR-HMDs (such as Microsoft HoloLens HL) have built-in Visual SLAM algorithms that perform spatial mapping to produce 3D meshes of the operating environments [55]. Using AR-HMD in SLAM would allow for intuitive human supervision and interaction with the agent/s performing localization and mapping. Moreover, having a human in the loop, who can intervene to correct errors and supervise the collaboration between different agents based on superior awareness and judgment skills, would enhance the performance and help overcome autonomy challenges.

The proposed application of MISA in collaborative SLAM allows three co-located heterogeneous types of agents (robot, AR-HMD, and human) to contribute to the SLAM process. Applying MISA here allows for real-time pose and map correction. Whenever an agent's self-

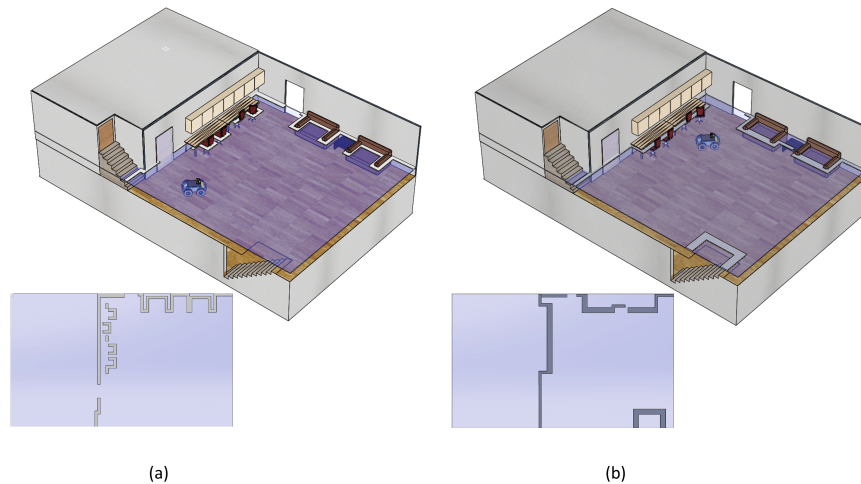


Figure 3.2: The Map obtained from traditional SLAM (a) versus the map that reflects the real environment more accurately (b).

confidence drops, it asks the human supervisor to approve/correct its pose estimation. In addition, the human can view the map being built superposed on the real environment through an AR interface that was specifically developed for this purpose, and apply edits to this map in real-time based on his/her SA. Moreover, when the robot is unable to reach a certain navigation goal, it prompts the human to assist in mapping the area corresponding to this goal; the human here is free to choose between mapping the area through editing the map him/herself or through activating the AR-HMD collaborative mapping feature. In the latter case, the AR-HMD contributes to the map building process by adding obstacles that are not detected by the robot and/or map areas that the robot cannot traverse.

The rest of the chapter is structured as follows: Section 3.2 presents previous related work. Section 3.3 gives an overview on the baseline SLAM approach and how the proposed framework is implemented in collaborative SLAM, while Section 3.4 describes the experimental setup, experiments, and results.

3.2 Related Work

Developing HRI systems where the efficiency and perception of humans aid the robot in SLAM has gained interest in literature. For instance, [56] showed that augmenting the map produced by a robot with human-labelled semantic information increases the total map accuracy. However, this method required post processing of these maps. A similar approach was developed by Diosi *et al.* [57], who presented interactive mapping, which uses human input to semi-autonomously create a labelled map of the environment. [58] proposed to correct scan alignments of 3D scanners through applying virtual forces by a human via a GUI. since this method lacks localization capability, it cannot produce large accurate maps. Sprute *et al.* [59]

proposed a system where a robot performs SLAM to map an area; when the robot is finished with mapping, the user can preview the environment through a tablet, equipped by an RGBD camera, and can manually add virtual borders to define forbidden areas. These borders are then added to the robot’s map as navigation-forbidden areas. Although these systems proved superior over full-autonomy methods by being more efficient in increasing map accuracy and navigation performance, they totally depend on human awareness and do not consider requesting help from the human in case of challenging situations.

The idea of collaborative SLAM is not new. An online pose-graph SLAM for multiple mobile robots using 3D LiDARS was proposed in [51]. This approach is based on a master-agent that receives odometry and scans matching factors from the robots in order to perform pose-graph optimization. Schmuk and Chli [50] presented a collaborative keyframe-based SLAM architecture consisting of multi UAV agents. In their work, each agent is performing short-memory SLAM, and a central station takes care of fusing sub-maps into a global optimized map and sending it back to the agents. Since these method adopts a centralized approach for map merging, any connectivity loss will cause major errors in SLAM estimates. [60] proposed a collaborative SLAM method that merges point clouds from a camera and rotating 2D LiDAR in order to localize a mobile robot in a map produced by a UAV, and Fankhauser *et al.* [52] localized a legged robot in a map produced by another robot. The problem with such approaches is that one agent fails to localize when navigating in an area that is not yet mapped by the second agent. In the proposed implementation of MISA framework in collaborative SLAM, delays and short communication losses are not a limitation since each agent runs SLAM on its own. Moreover, the proposed system benefits from one agent (e.g. AR-HMD) to correct the mapping errors of the other agent (e.g. robot) under the supervision of a human operator, who can also intervene based on the agents’ requests or his/her judgment.

Integrating AR technology in human-robot interaction is relatively new but is gaining more interest with the advancement of AR hardware. In [54], a HoloLens is used to control a robotic arm in a pick-and-place application. AR was also used in [61] to train a robot to open bottles via a virtual gripper. Moreover, an AR-HMD was used to program robots and view their plans in [53] and [62]. In the context of navigation, Zolotas *et al.* [63] proposed an AR system to help users control their wheelchairs. Finally, [64] proposed the use of AR in collaborative search-and-rescue missions. The system consists of a robot running SLAM and communicating with HL to display paths for the operator when the robot finds its goal. The implementation of MISA in SLAM makes use of AR-HMD to (1) visualize the map created by a robot performing SLAM aligned on the physical environment, (2) evaluate the map correctness, and (3) edit the map in real-time through intuitive gestures. Moreover, the system benefits from the visual SLAM capabilities of the HoloLens to collaborate with the robot in map building tasks.

3.3 System Overview

This section presents a brief description of OpenSLAM Gmapping, the SLAM baseline approach, followed by an overview of MISA’s implementation on top of this baseline.

3.3.1 Baseline Approach: OpenSLAM Gmapping

To establish a baseline, OpenSLAM gmapping [65] is adopted. OpenSLAM gmapping is a SLAM technique that applies Rao-Blackwellized particle filter (RBPF) to build grid maps from laser range measurements and odometry data [65]. RBPF was first introduced by Murphy *et al.* [66]. It consists of N particles in set R_t where each particle $R_t^{(i)} = (\zeta_t^{(i)}, \eta_t^{(i)}, w_t^{(i)})$ is presented by a proposed map $\eta_t^{(i)}$, pose $\zeta_t^{(i)}$ inside this map, and an importance weight $w_t^{(i)}$. OpenSLAM can be summarized by five main steps [65], shown in Figure 3.3 and briefly discussed below.

Pose Propagation: At every iteration, odometry u_t is calculated through the motion model and used to propose the new poses from the previous poses ζ_{t-1} . Each particle $R_t^{(i)}$ is sampled from $R_{t-1}^{(i)}$ by giving it an initial pose $\zeta_t^{\prime(i)} = \zeta_{t-1}^{(i)} \oplus u_t$, where \oplus is the standard pose compound operator [67].

Scan Matching: For every particle with pose $\zeta_t^{\prime(i)}$, scan-matching between the new observation z_t and the previous map $\eta_{t-1}^{(i)}$ is applied, and a score is given to this pose based on matching percentage. Each particle's pose is then adjusted within a predefined window until the maximum scan matching score is achieved for this single particle, resulting in a new corrected pose $\zeta_t^{(i)}$ for each particle.

Importance Weighting: The observation model $p(z|\zeta, \eta)$ and observations z_t are used to calculate an updated importance weight w_t^i for each particle according to the importance sampling principle given by equation (3.1).

$$w_t^{(i)} = w_{t-1}^{(i)} \cdot p(z_t | \zeta_t^{(i)}, \eta_{t-1}^{(i)}), \quad (3.1)$$

Adaptive Resampling: This step is important for replacing particles of low importance weights by those of high weights. First, dispersion of the particles' importance weights is measured through the effective sample size N_{eff} , which is given by equation 3.2. If N_{eff} drops below a

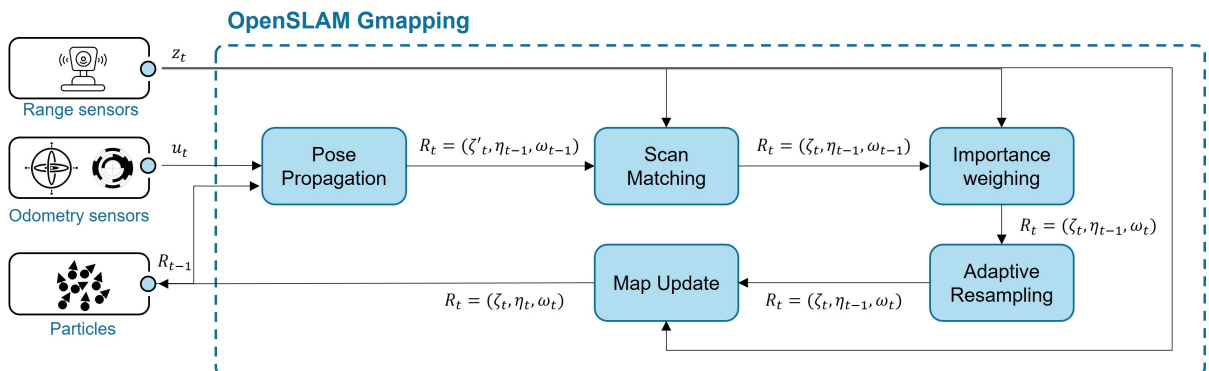


Figure 3.3: Flowchart representing the different modules in OpenSLAM gmapping baseline.

pre-defined threshold, resampling is done by replacing particles with $\tilde{w}_t < 0.5$ by duplicates of the particles with $\tilde{w}_t > 0.8$, where \tilde{w}_t is the normalized importance weight of particle $R_t^{(i)}$.

$$N_{eff} = \frac{1}{\sum_{i=1}^N (\tilde{w}_t^{(i)})^2}, \quad (3.2)$$

Map update: Each map $\eta_t^{(i)}$ is updated based on the particle’s new pose $\zeta_t^{(i)}$ and observation z_t . The map representation technique used is Occupancy Grid Mapping (OGM), which was first introduced first by Moravec and Elfes [68]. In OGM, maps are formed out of cells with assigned occupancy probabilities where a probability between 0 and the occupancy threshold th_{oc} means that the cell is Free, probability between th_{oc} and 1 means that it is Occupied, and a probability of -1 means that this cell is still Undiscovered.

The pose and map corresponding to the particle with the highest weight are considered as the robot pose P_R and robot map M_R in the remaining of this chapter.

3.3.2 Applying MISA in Collaborative SLAM

This section describes how the proposed MISA framework is applied in collaborative SLAM, following the generic methodology presented in Chapter (2). Table 3.1 summarizes the inputs form the SLAM problem, which are used to satisfy the MISA framework.

- **Tasks to be supervised by the human:**

Based on the surveyed literature and the full-autonomy experiments presented in Section 3.4.2, the main challenges facing SLAM are inaccurate map representation and inaccurate localization. In SLAM, mapping and localization are performed simultaneously, and thus any error that happens in one affects the other; similarly, increased accuracy in any of the two tasks enhances the accuracy of the other. Thus, the human should be able to supervise and intervene in the following tasks: (1) map building by the robot, (2) map building by the AR-HMD, (3) robot pose estimation, (4) AR-HMD pose estimation, (5) global map merging, and (5) the ability of the robot to acheive a given navigation goal.

Table 3.1: MISA attributes in collaborative SLAM.

Tasks to be supervised		Attributes		
SC-based tasks	SA-based tasks	SC Attributes	SA Attributes	MISA-based UI
<ul style="list-style-type: none"> • Robot pose estimation • AR-HMD pose estimation • Ability to reach navigation goal 	<ul style="list-style-type: none"> • Map building by robot • Map building by AR-HMD • Global map merging 	<ul style="list-style-type: none"> • Confidence in pose • Confidence in reaching navigation goals • SC-metric: N_{eff} • SC-event: HoloLens tracking feature 	<ul style="list-style-type: none"> • Maps quality • SA-tools: Augment the map on physical world 	<ul style="list-style-type: none"> • AR interface • 2D GUI

- **Robot SC attributes:**

The robot's SC in the pose estimation is calculated through the N_{eff} metric which reflects the dispersion of the particles' importance weights. This metric serves as an estimation of how well the particles represent the true pose of the robot [65]; thus, N_{eff} is employed as SC-metric to reason about when the robot should prompt the human for assistance in localizing itself. The robot prompts the human to approve or correct the estimated pose whenever N_{eff} drops below a threshold that was obtained experimentally. As for the HoloLens, the human supervisor is requested to help in the pose estimation based on a built-in feature that flags the HoloLens localization error event. Since any error in localization affects the whole map accuracy, the selected 'no response' strategy forces the robot to pause its operation until human response is received. Moreover, when the robot is given a navigation goal that it is not able to find a valid path to, it prompts the human supervisor to help in mapping the area corresponding to this goal. For this task, the robot proceeds into its next goal even if the human does not respond to the request directly.

- **Human SA attributes:**

Since SC attributes could not be defined in the map building task (for both robot and HoloLens), the applied MISA system benefits from the human perception and SA in assessing the quality of maps built by both agents. To help the human supervisor compare the map built by the agents with the real physical environment and apply edits, the SA tool proposed is augmenting the map on the physical world. The human is allowed to edit the robot's map by toggling the status of cells between 'Occupied' and 'Free'. Moreover, the human can choose to activate the collaboration feature to assist the robot in mapping whenever he finds it necessary.

- **MISA-based user interface:**

To maximize the human's SA and provide an intuitive way of interaction, an augmented reality interface (ARI) running on HoloLens is developed. Through this interface, humans can see the augmented map superposed over the real environment and can interact with it through their hands. Moreover, and upon request, humans can influence the robot's pose estimation through a GUI.

Figure 3.4 presents the block diagram of MISA framework applied in Collaborative SLAM. Since **OpenSLAM gmapping** block is already described in section (3.3.1), the **MISAC** block only is discussed below.

3.3.2.1 SC-Interaction handler

When the HoloLens SC-event is triggered, this module prompts the human supervisor to re-localize the HoloLens manually or through an AR marker placed on the robot. When the robot is unable to generate a valid path for a navigation goal, the module notifies the supervisor to map it manually or through activating the AR-HMD Map Builder feature. As for the robot pose estimation, this module replaces the *Adaptive Resampling* module that was discussed in section (3.3.1). It represents the mechanism applied to check the robot's SC and prompt the

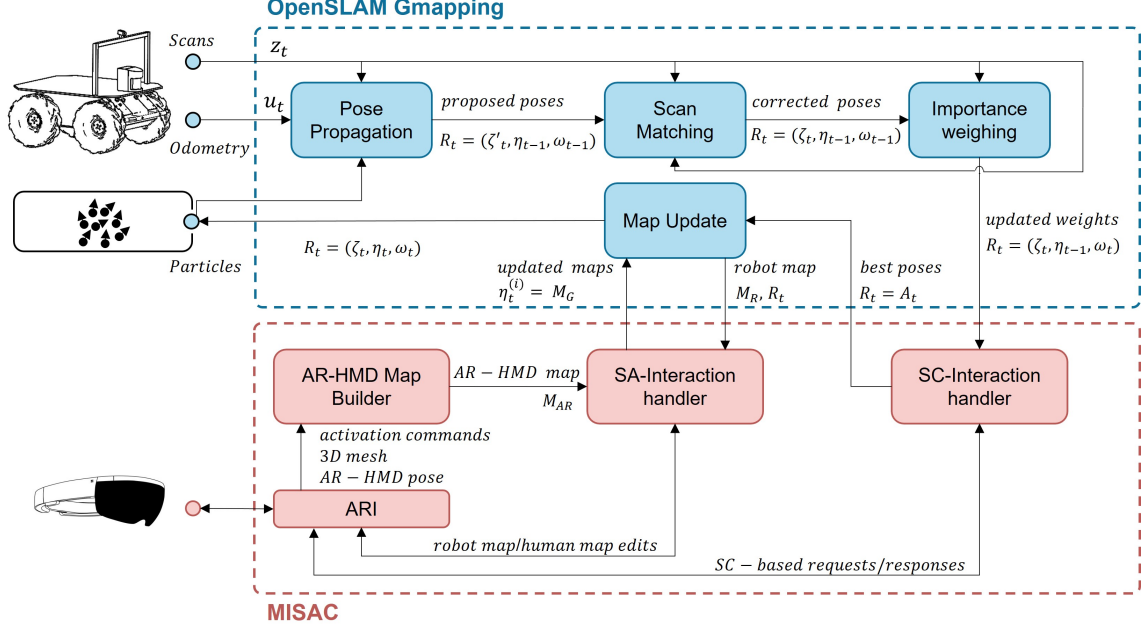


Figure 3.4: Block diagrams of MISA in collaborative SLAM.

human supervisor for help. The new ‘human augmented’ set of particles A_t , where $A_t^{(i)} = (a_t^{(i)}, \eta_t^{(i)}, w_t^{(i)})$, is sampled from R_t through the following steps:

1. *Self-confidence check:*

The robot’s self-confidence is based on the N_{eff} metric that is given in equation (3.2). At this step, if $N_{eff} > r_{th}$, where r_{th} is a resampling threshold obtained through empirical testing, no human assistance is requested; thus steps 2 to 5 are ignored and R_t is passed as-is to the OpenSLAM gmapping block ($A_t = R_t$). If $c_{th} < N_{eff} < r_{th}$, where c_{th} is a confidence threshold that is obtained experimentally, resampling is done as described in Section 3.3.1 and no human assistance is required.

2. *Acquiring human assistance:*

When $N_{eff} < c_{th}$, the human is requested to approve or correct the pose estimation through the GUI. When the new human-corrected pose is acquired, an augmented pose \acute{a}_t is calculated as:

$$\acute{a}_t = P_R \oplus \check{d}_t, \quad (3.3)$$

where \check{d}_t is the pose difference vector between the human-corrected pose and the robot pose P_R . A Gaussian approximation $a_t^* \sim \mathcal{N}(\mu, \Sigma)$ is then computed to propose the new particles distribution around the human augmented pose; resulting in a new sampled pose $a_t^{*(i)}$ for each particle. This approximation is applied here since the human correction itself may include errors as well. The mean of the distribution μ is equal to \acute{a}_t , and variance Σ is set to be $(0.1m, 0.1m, 0.08rad)$, which is obtained through experimentation.

3. *Scan matching:*

A scan-matcher based on the CARMEN Toolkit [69] is implemented to find the best pose

that maximizes the scan-matching score. For each particle, the algorithm searches for the best pose $a_t^{(i)}$ within a pre-defined window in map $\eta_{t-1}^{(i)}$ starting from the initial guess $a_t^{*(i)}$.

4. *Importance weighting:*

Same as described in section (3.3.1).

5. *Setting the new poses:*

In this step, the set of particles is updated with the corrected poses $a_t^{(i)}$ and sent back to the OpenSLAM gmapping block ($R_t = A_t$).

3.3.2.2 SA-Interaction handler

This module is responsible for handling interactions that are based on human SA; it maintains an updated global map M_G , that is used by all agents, through fusing the human edits and the AR-HMD map M_{HL} with the most recent robot map M_R through the following two functions:

1. **AR Map Merger:** when the human supervisor requests to check the map, this function activates, acquires the most recent M_R , and visualize it through the ARI as M_{merged} . Initially, $M_{merged} = M_R$, where M_R is a 1D occupancy grid of size Q :

$$M_R = \{m_1, m_2, m_3 \dots m_Q\}, \quad (3.4)$$

Every element m_i has a value equals to the occupancy probability of the corresponding map's cell.

When the supervisor edits the map or activates the AR-HMD Map Builder, this function updates M_{merged} through comparing the the corresponding cells of the acquired M_R with the human-edited cells and/or the updated cells from M_{HL} which is constructed through the AR-HMD Map Builder module. The function always prioritizes human edits over both the robot map and the AR-HMD map, and it prioritizes Occupied status over Free whenever the cell status is not consistent between the acquired robot map and the AR-HMD map.

2. **Global Map Updater:** when M_{merged} is received from the AR Map Merger, this module activates to fuse M_{merged} with the most recent M_R resulting in a global map M_G . This is done since the robot might update its map while the human is performing the edits. To avoid collision between the robot and undetected obstacles, cells that are set to be Occupied by the AR Map Merger, but found to be Free by the robot, are set as Occupied through this function.

The SA-Interaction handler module maintains same map size for all agents and replaces the maps associated with all particles with the updated global map, $\eta_t^{(i)} = M_G$.

3.3.2.3 AR-HMD Map Builder

This module is activated by the supervisor to produce an AR-HMD occupancy grid map M_{HL} in two steps: (1) *raycasting* to detect the obstacles, and (2) *map updating* to build and update the AR-HMD occupancy grid map. The produced M_{HL} has the same size and resolution as M_R , and its frame is aligned with the robot's map frame. The AR-HMD device is assumed to be able to create a 3D mesh or point cloud of the environment, and the relative transformation between the robot's map frame and the AR-HMD frame is known. Therefore, the pose of the human operator, p_a , and the heading vector in the robot's map frame, v_a , are known in real-time. All poses and distances are calculated in the map's frame.

Raycasting: is a well-known technique that is used to detect heading and distance to obstacles. By considering the AR-HMD as a virtual 3D LiDAR that transmits a series of rays originating from $p_a = (x_a, y_a, z_a)^T$ with defined directions, the proposed raycasting approach is presented in Algorithm 1 and illustrated in Figure 3.5. Each set of rays has a frame that is rotated by an angle α around the vertical axis of the AR-HMD. Then in each set, each ray's direction v_r is calculated by rotating the ray's frame by angle β around the set's frame. The direction of each ray can be calculated through the function (**RayDirection**) as:

$$v_r = R_{v,\alpha} \times R_{h,\beta} \times v_a, \quad (3.5)$$

where

$$R_{v,\alpha} = \begin{bmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (3.6)$$

$$R_{h,\beta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \beta & \sin \beta \\ 0 & -\sin \beta & \cos \beta \end{bmatrix}, \quad (3.7)$$

Each ray is then sent through the function (**CastRays**) that returns a true value to the flag hit and the hit point $p_h = (x_h, y_h, z_h)^T$, if the ray intersects with the 3D mesh. The horizontal distance d between p_a and p_h is then calculated through (**Horizontal_distance**):

$$d = \sqrt{(x_h - x_a)^2 + (y_h - y_a)^2}, \quad (3.8)$$

Only the hit corresponding to the minimum **Horizontal_distance** d_{min} recorded for this specific set is registered in the hit points list H.L as 3D Cartesian point $P_{hit} = (x_{hit}, y_{hit}, z_{hit})^T$. In other words, only the nearest hit to the AR-HMD from each set is obtained. Finally, the registered points list H.L is sent to the *Map Updater* module.

Map Updater: this module handles updating the M_{HL} when it receives H.L. For every P_{hit_i} in H.L, the elevation component, z_{hit} , is checked against the predefined ground height, z_{ground} . This step is added to avoid detecting the ground as an obstacle. If z_{hit} is not equal to $z_{ground} \pm error\ tolerance$, i.e. the detected obstacle is not ground, P_{hit_i} is added to the obstacles list O.L, then each cell c in the map is updated according to:

$$l(c | z_{1:t}) = l(c | z_{1:t-1}) + l(c | z_{1:t}), \quad (3.9)$$

Algorithm 1: Raycasting: $H_L = \text{Raycasting}(p_a, \mathbf{v}_a)$

Result: H.L
Input: AR-HMD pose (p_a), Heading vector (\mathbf{v}_a);
initialization;
H.L to zeros;
 d_{min} to maximum ray length;
for $\alpha = \alpha_{min}$ *to* α_{max} *step* θ_1 **do**
 for $\beta = \beta_{min}$ *to* β_{max} *step* θ_2 **do**
 $\mathbf{v}_r = \text{RayDirection}(\alpha, \beta, \mathbf{v}_a)$;
 $\text{hit} = \text{CastRays}(\mathbf{V}_r).hit$;
 if hit **then**
 $p_h = \text{CastRays}(\mathbf{V}_r).point$;
 $d = \text{Horizontal_distance}(p_a, p_h)$;
 if $d < d_{min}$ **then**
 $d_{min} = d$;
 $P_{hit} = p_h$;
 else
 pass;
 end
 else
 pass;
 end
 end
end
add P_{hit} to H.L

where $l(c | z_t)$ is the inverse sensor model (ISM) defined as:

$$l(c | z_t) = \begin{cases} l_{occ}, & \text{for cells in } O_CL \\ l_{free}, & \text{for cells in } F_CL \end{cases} \quad (3.10)$$

with $l_{occ} \in]0, 1]$ and $l_{free} \in [-1, 0[$. Moreover, $l(c | z_t)$ is limited between minimum and maximum occupancy bounds to avoid over-confidence.

3.4 Implementation and Experiments

3.4.1 Experimental Setup

The proposed MISA framework is implemented using Robot Operating System (ROS), and the ARI is developed using Unity3D. The SLAM baseline approach is deployed through *Gmapping* ROS package, which implements OpenSLAM's Gmapping [70], and the *move_base* ROS package [71] is used for autonomous navigation. This package implements Dijkstra's algorithm

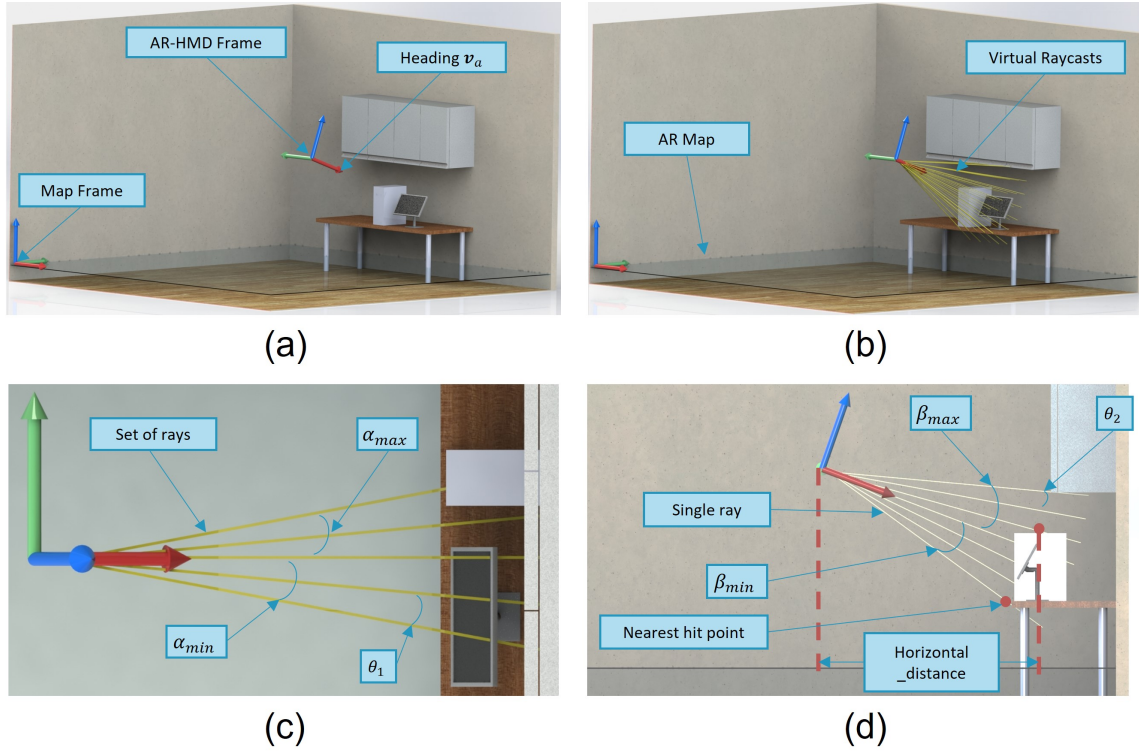


Figure 3.5: Illustration of the *raycasting* unit in the proposed method. (a) shows the map and AR-HMD frames alongside the operator’s heading vector. (b) shows how the virtual rays are casted from the AR-HMD. (c) shows a top-view of the ray sets where each set is deviated from the other by θ_1 and from the heading vector by α . (d) shows the rays contained in each set, where each ray is rotated by angle β around the horizontal axis of the corresponding set; also shown are the nearest hit point and Horizontal_distance.

to generate global plans for the robot navigation. The communication with the ARI is handled through *RosBridge_Suite* [72] ROS package and *HoloToolkit.Sharing* library [73]. The implementation and experiments were performed on a Microsoft HoloLens as an AR-HMD, and a Clearpath Husky A200 Explorer robot with installed 2D LiDAR and wheel encoders.

The implementation flowchart is shown in Figure 3.6. The human supervisor is equipped with a HoloLens running the ARI, and is co-located with the robot performing SLAM. When the system is started, the set of particles R_t is initialized, and the supervisor is requested to initialize the augmented map (M_{merged}) through looking at an AR marker placed on the robot. This is done to align the merged map augmented through the HoloLens with the robot’s map frame. If the path planner fails to generate a valid path for a given navigation goal, it notifies the supervisor to map the area manually or through activating the AR-HMD Map Builder. Moreover, when the HoloLens flags the SC-event, meaning that it has lost tracking, the supervisor assistance is requested to re-initialize M_{merged} by looking at the AR marker again or manually adjusting the augmented map through the ARI. Whenever the set of particles is updated, meaning that the robot is moving or the robot’s surrounding has changed, N_{eff} is calculated and compared against the robot’s pose confidence threshold r_{th} . If the robot is not confident about

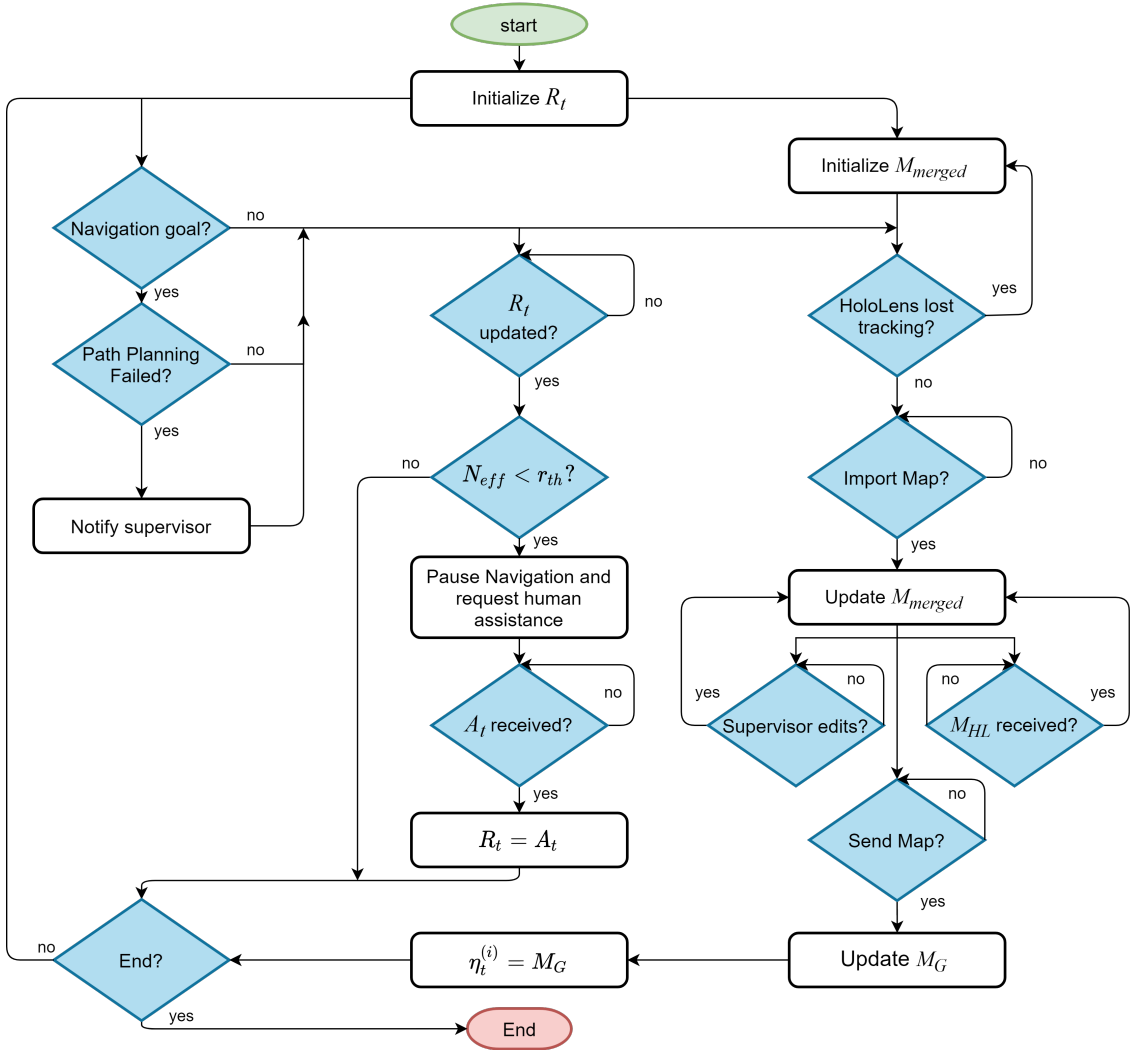


Figure 3.6: Flowchart presenting the implementation logic of MISA in collaborative SLAM.

its pose ($N_{eff} < r_{th}$), navigation is paused and the human is requested to correct P_R through Rviz [74] interface. Navigation is not resumed unless the human approves or modify the pose. Here, A_t is calculated and R_t is updated through the SC-Interaction handler. While SLAM is running, the human supervisor is allowed to import the current robot's map M_R at any time to evaluate it. Here, M_{merged} is set as a duplicate of M_R through the AR Map Merger function and is visualized through the ARI. The supervisor is allowed to edit the map manually or through activating the AR-HMD Map Builder to produce M_{HL} , and these edits are fused in M_{merged} in real-time. When the supervisor confirms the updated M_{merged} , the Global Map Updater function compares it with the most recent M_R , updates M_G , and replaces all particle maps $\eta_t^{(i)}$ with the updated M_G . This is done to make sure that any map updates performed by the robot while the human is editing are being fused in the global map, and that this global map is conveyed back to all particles.

Figure 3.7 shows real snaps of the ARI, which overlays M_{merged} on the LiDAR's plane of the robot, and menu items are always rendered to the left of the supervisor. In the augmented

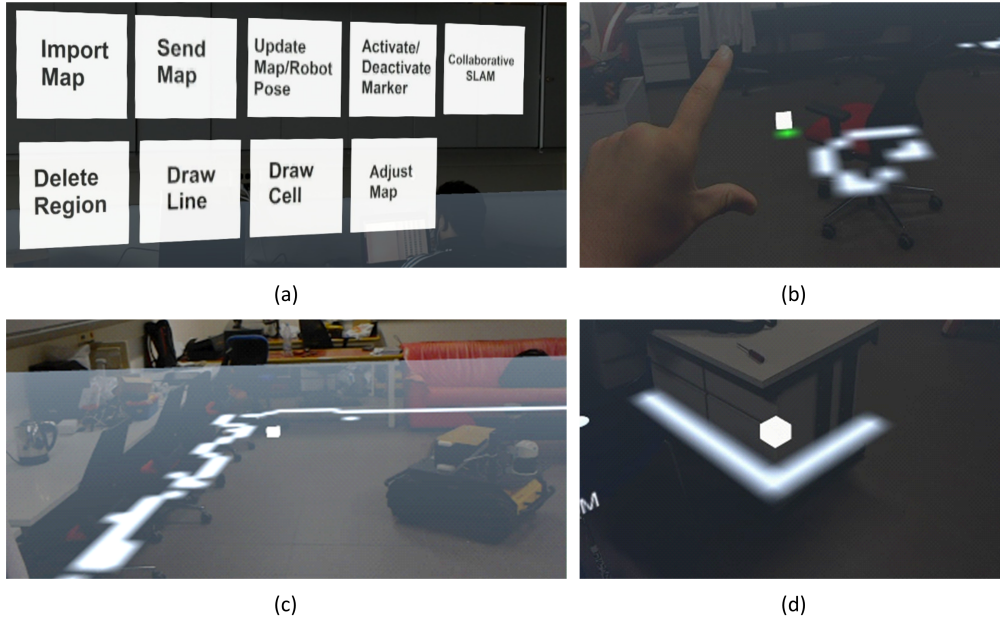


Figure 3.7: The developed AR Interface. (a) shows the menu items, (b) shows the “tap” gesture, (c) and (d) show samples of map augmentation.

M_{merged} , white cells are considered Occupied and gray cells are considered Free/Undiscovered. Figure 3.8 illustrates the used frames and poses:

- **Map Frame F_M** : Placed on the bottom-left corner of the augmented map.
- **World Frame F_W** : The HoloLens world frame, initialized and fixed by the HoloLens when the system runs.
- **Map pose P_M** : Pose of F_M in F_W , fixed at the initialization phase of the system.
- **Robot pose P_R** : Pose of the robot in F_M .
- **HoloLens pose P_H** : Pose of the HoloLens inside F_W , acquired through the HoloLens built-in visual SLAM.
- **Gaze point pose P_G** : the intersection between the viewing ray (gaze line) and the augmented objects. The gaze line vector is acquired from the HoloLens itself in real-time.

The ARI’s Menu items, which handle sending/receiving requests and messages to/from the robot, are discussed below:

- **Import Map**: Acquires the current M_R and duplicates it as M_{merged} , a 1D occupancy grid array of size Q . M_{merged} is converted to a 2D grid map of width W , height H , and cell’s resolution α_m . The map is visualized as an image plane of $(W \times H)$ pixels, where each pixel represents a cell. The plane’s dimension is $(W \times \alpha_m)$ by $(H \times \alpha_m)$.

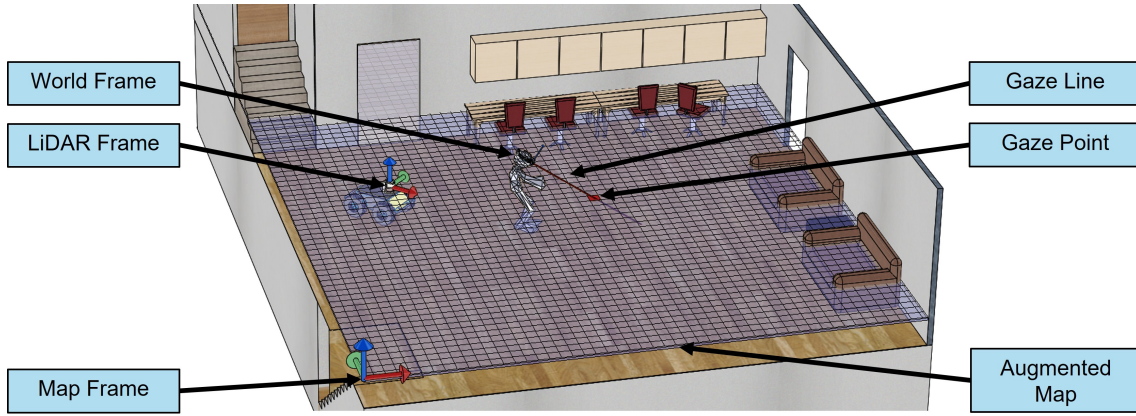


Figure 3.8: ARI components and the various reference frames.

- **Activate/Deactivate Marker:** Activates and deactivates AR marker recognition and tracking. A predefined AR marker is placed on the robot and used to initialize P_M . Vuforia SDK [75] for Unity is used for AR marker recognition and tracking. When the marker recognition is enabled, and the marker is in the view field of the HoloLens, the ARI acquires the most recent P_R and calculates P_M . To account for localization drifting or lost tracking in the HoloLens' visual SLAM, the supervisor can re-initialize the map's pose at any time. Marker deactivation capability is added to account for instability and drifting in the HoloLens visual odometry that is observed when keeping the marker tracking enabled.
- **Adjust Map:** allows for manual adjustment of the augmented M_{merged} 's pose by applying translations (orientation adjustment is done through the AR Marker). The supervisor has to click on a reference cell in the map, then click on the true position of that cell (considering the physical world).
- **Draw Line:** enables a set of cells that form a single straight line to be set as occupied or free. The supervisor only has to click on the starting and end points of the line.
- **Delete region:** a rectangular region could be selected by clicking on two cells that form the diagonal of the desired rectangle. All cells within this area would be set to free.
- **Send Map:** when the supervisor is satisfied with M_{merged} , this button should be clicked to proceed with updating M_G and the maps of the particles. The 2D pixels array of the image plane are converted back to 1D occupancy array before updating the maps of the particles.
- **Collaborative SLAM:** enables the AR-HMD Map Builder.

Interaction with the augmented objects is obtained through the HoloLens-supported “tap” gesture. When looking on the augmented menu items, a round cursor is rendered on the intersection point P_G . However, to facilitate the map editing procedure, if the gaze line happens to intersect with the map plane, P_G is rounded to the nearest cell's (pixel) center, and a cube is

Table 3.2: Experimentation sets and their parameters.

Set	A			B			C		
Lighting	Day, Night			Day, Night			Day, Night		
Laser range(m)	5.5-6.1			7.5-8.1			9.5-10.1		
Number of Particles	5	50	100	5	50	100	5	50	100

rendered inside that cell of interest, so that the supervisor can tap on the cell to toggle its status between Free and Occupied.

3.4.2 Experimental Testing and Results

Experiment-1: the first batch of experiments was conducted to test the performance of the baseline system in full autonomy mode in order to define the MISA framework parameters. The testing area was carefully selected in a way to introduce the common challenges that affect the performance of SLAM (*i.e.*, low semantic features, glass walls, object higher and others lower than the LiDAR’s detection plane). Table 3.2 presents the parameters used in the different sets of this batch, where each set of tests was run three times, resulting in 54 tests. In these sets, the operator had to tele-operate the robot, following a defined trajectory for a distance of about 170m, to map the area shown in Figure 3.9.

The laser range was the lowest in Set A. Scan Matching rapidly failed in region 2 where detection of indoor obstacles was unattainable. This increased the effect of odometry drift and thus most maps produced had huge errors. The best map obtained in this set (A) is shown in Figure 3.10a and the worst is shown in Figure 3.10b. In Sets B and C, the laser range was increased to mid-range and high-range, respectively. The higher range was chosen so that the robot detects the inner walls while navigating inside region 2. While Figure 3.11a and Figure 3.12a both show an improvement over the best map that could be achieved in Figure 3.10a, this improvement comes at a higher computational cost due to the increased laser range. The worst maps obtained using both ranges are also documented in Figures 3.11b and 3.12b.

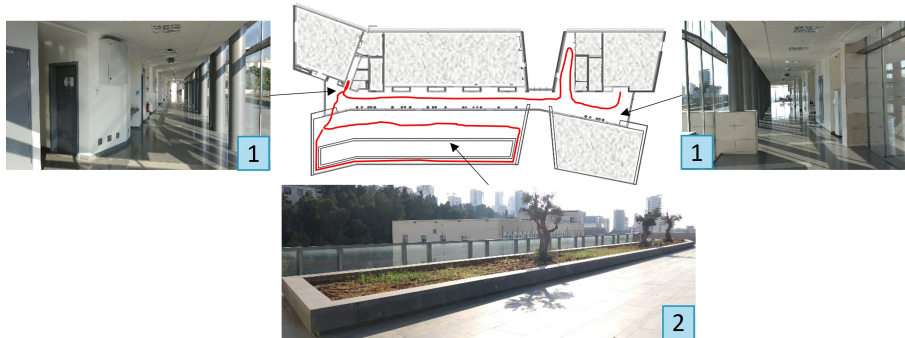


Figure 3.9: Testing area at the American University of Beirut. The line in red represents the path that should be followed by the robot while conducting the experiments. Region 1 contains glass walls that are not detectable by the LiDAR, and Region 2 is an outdoor terrace.

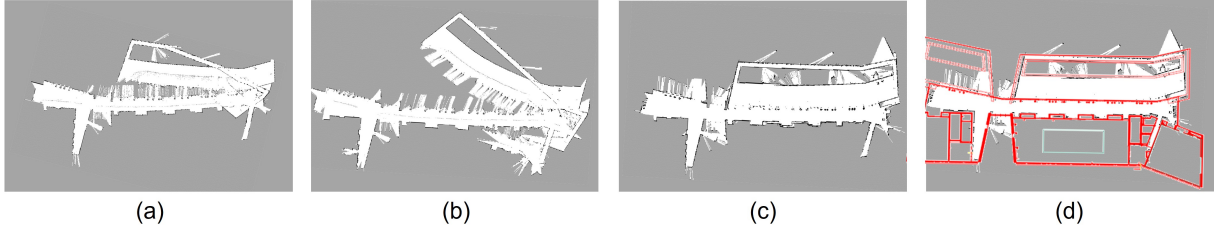


Figure 3.10: Maps obtained using Set A (low laser range). The best map obtained by OpenSLAM gmapping is shown in (a), the worst map is shown in (b), (c) shows the map obtained using the proposed MISA framework, and (d) overlays the ground truth map on top of (c) to demonstrate the effectiveness of the MISA framework.

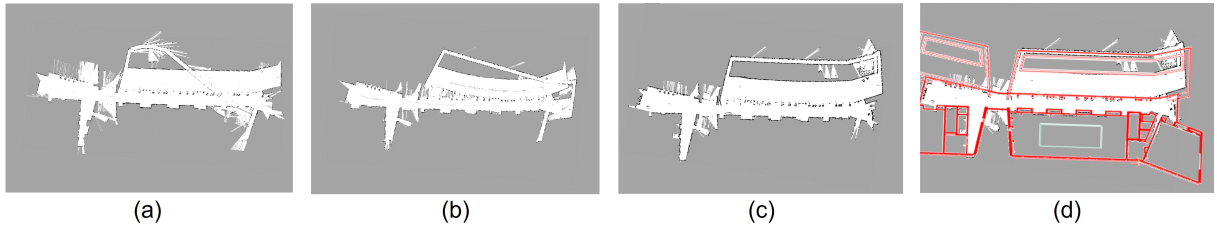


Figure 3.11: Maps obtained using Set B (medium laser range). The best map obtained by OpenSLAM gmapping is shown in (a), the worst map is shown in (b), (c) shows the map obtained using the proposed MISA framework, and (d) overlays the ground truth map on top of (c) to demonstrate the effectiveness of the MISA framework.

The obtained maps were analyzed by overlaying the ground truth map on the resulting map, visually evaluating the geometric matching, and manually labelling the map between ‘success’ and ‘fail’. Success is defined as the ability to get a robot’s map matching the ground truth map, up to a certain extent, with no areas overlaying on top of each other. In the performed tests, 31 out of 54 runs produced ‘successful’ maps, resulting in a 57.4% success rate. However, even successful maps requires post processing to fix some mapping errors such as undetected and/or wrongly mapped obstacles.

From literature, it is known that the effective sampling size N_{eff} is used to assess how well the particles’ poses are close to the target posterior [65]; thus, this metric was evaluated as a SC-metric for the robot’s pose estimation. Using results of the tests discussed above, the recorded N_{eff} , particles estimated poses ζ_t , and particles normalized weights \tilde{w}_t were utilized in this evaluation, and the findings are summarized in the following three scenarios:

1. When navigating in areas with non-symmetrical contours, Scan Matching can update the estimated particle poses (resulting from Pose Propagation) with high accuracy; thus, the dispersion of ζ_t and \tilde{w}_t is minimal, yielding to high N_{eff} score (see Figure 3.13a). Here, resampling of particles is not needed.
2. When navigating in areas with high symmetry levels, Scan Matching might yield incorrect pose updates, which results from the fact that certain contours in the map look identical to

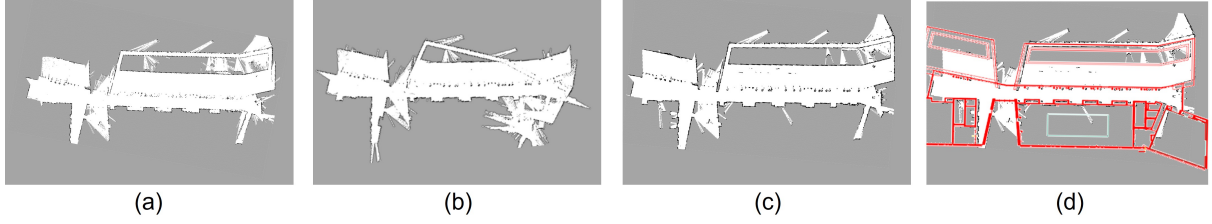


Figure 3.12: Maps obtained using Set C (high laser range). The best map obtained by OpenSLAM gmapping is shown in (a), the worst map is shown in (b), (c) shows the map obtained using the proposed MISA framework, and (d) overlays the ground truth map on top of (c) to demonstrate the effectiveness of the MISA framework.

the contours around the true pose; this leads to higher dispersion of ζ_t and \tilde{w}_t , and lower N_{eff} score as shown in Figure 3.13b. In this case, resampling replaces the particles with low weights (inaccurate poses) with those of high weights (accurate poses).

3. In cases where no enough contours are present (*i.e.*, open areas, transparent walls..), Scan Matching fails and the pose estimation depends mainly on the Pose Propagation step of Gmapping. Here, the particles pose estimation is highly affected by wheel slippage and odometry noise/drift. Since these estimations are not corrected by Scan Matching, the error in the estimated particle poses accumulates, resulting in high dispersion of ζ_t and \tilde{w}_t , and a very low N_{eff} score as shown in Figure 3.13c. In this case, resampling is not able to account for the accumulated error since even particles with $\tilde{w}_t^{(i)} > 0.8$ might have very low importance weights $w_t^{(i)}$. This scenario is responsible for the large drifts presented in the maps in Figures 3.10a, 3.10b, 3.11a, 3.11b, and 3.12b.

Based on this evaluation, two thresholds are defined: Resampling threshold r_{th} and confidence threshold c_{th} . When $c_{th} < N_{eff} < r_{th}$, resampling is performed as discussed in Section 3.3.1, and when $N_{eff} < c_{th}$, human assistance in the robot's pose estimation is requested. The numerical values of these thresholds were obtained experimentally as $r_{th} = 0.7$ and $c_{th} = 0.5$.

Additional tests were done to assess the HoloLens performance in spatial mapping. The challenges detected are: fast movements, walls or objects that are low-textured, and looking at the ground while moving. These scenarios result in drifting in HoloLens localization or/and tracking loss error. Normally, this leads to unpredicted transformations in the augmented Holograms. Thus, the proposed MISA framework solves this problem by means of the re-initializing feature, which relocates the map in the environment based on the robot's pose as discussed earlier.

Experiment-2: additional experiments were performed to test the effect of requesting human assistance in pose estimation on the performance of SLAM. One test was done on each set of Table 3.2, using 100 particles, while applying the SC-metrics discussed in Experiment-1. In these experiments, the robot stops and request human assistance when $N_{eff} < c_{th}$. Figures 3.10c, 3.11c, and 3.12c shows the maps obtained through these tests, which are overlaid by the blueprints of the area in Figures 3.10d, 3.11d, and 3.12d. The conducted experiments demonstrate how implementing MISA to request human assistance in pose estimation can result in

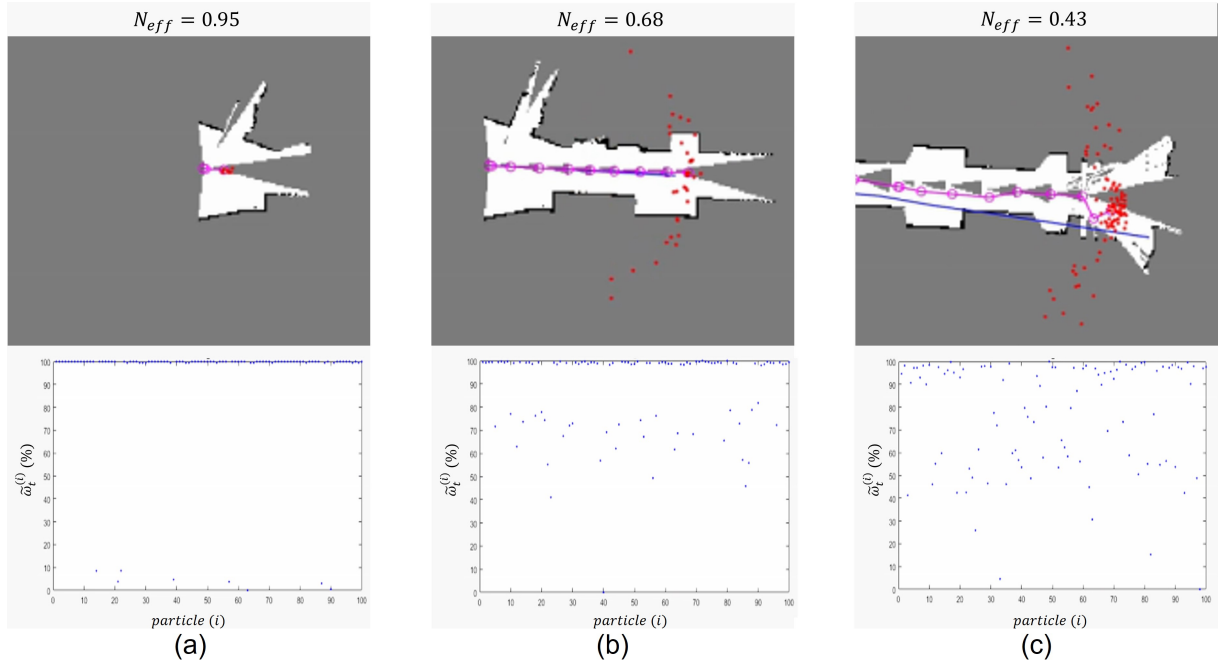


Figure 3.13: The three scenarios discussed in N_{eff} evaluation. Red dots represent the particles.

higher quality maps, especially in areas that are considered challenging. In addition to the qualitative assessment, below are some quantitative results to compare the proposed implementation of MISA framework in SLAM:

1. *Run time*: tests using the MISA framework took 35% – 50% less time than those performed with OpenSLAM gmapping. This is due to the fact that when applying pose correction throughout the run, there is no need to revisit areas and check for loop closures.
2. *Computational Costs*: since no computational complexity is added by the MISA system, no significant added computational costs were recorded.
3. *Success rate*: when using OpenSLAM gmapping, 31 out of 54 runs produced useful maps, all of which required post processing, resulting in a 57.4% success rate. Through the MISA framework, maps with no overlaying areas were produced in 100% of the tests. However, all of the maps required post-processing to account for minor mapping errors.
4. *Human workload*: Throughout the tests that involved pose correction, the operator needed to change the robot’s pose 9.5% of the time when the robot’s self-confidence dropped below the specified threshold. In 91.5% of the time, the supervisor only had to confirm the robot’s pose estimation.

Experiment-3: this experiment was conducted to test the performance of full MISA integration in collaborative SLAM. Figure 3.14 shows a blueprint of the testing site. **Region I** was staged to include: (1) objects higher and others lower than the LiDAR’s detection plane, (2)

glass walls, and (3) reflective objects. Moreover, it was separated from **Region II** by an exit that is too narrow for the robot to pass through. **Region II** is a narrow corridor with poorly textured walls, and region **Region III** features two glass walls facing each other. The resolution of both the robot's and augmented occupancy grid maps were set to $0.1m \times 0.1m$ with an initial size of $10m \times 10m$. The map's update rate is set to $0.5s$, the LiDAR's measurement range to $5m$, and the HoloLens raycasting rays length to $2m$. For the experiment, the human supervisor is asked to map the entire area in the fastest/most accurate manner. The supervisor was instructed to initialize the augmented map by looking at the AR-Marker placed on the robot, and then s/he is free to:

- move inside the entire test area,
- send navigation goals or tele-operate the robot using a joystick,
- import the most recent map from the robot at any time,
- edit any part of the map whenever s/he finds errors,
- send the merged map to the robot where it is used to update the global map,
- choose when to use AR-HMD mapping through the HoloLens,
- re-initialize the augmented map to align it with the physical environment in cases of drifting or when the HoloLens confidence in its pose estimate drops, and
- correct or approve the robot's pose when requested.



Figure 3.14: Blueprint of the testing arena and photos of the experimental setup. **Region I** contains obstacles higher/lower than the LiDAR's detection plane, glass walls, and reflective objects. **Region II** is a corridor with poorly textured walls, and **Region III** has two glass walls facing each other. The robot starts in **Region I** and cannot traverse to **Regions II** and **III** since the exit is narrow.

After initializing the map, the supervisor started from **Region I** where she tele-operated the robot in order to map the area. Figure 3.15a shows the resultant map and Figure 3.15b shows the map overlaid on the blueprint. Although the robot succeeded to map the walls accurately, it failed to correctly map the couches, glass, and the objects because they do not intersect with the LiDAR's 2D plane; objects 1 to 5 in Figure 3.15a are examples of obstacles that were not mapped. Given that the supervisor can see the map superposed over the physical environment, she can activate the HoloLens *AR-HMD Map Builder* module and walk around the unmapped objects to allow the HoloLens to improve the map. Figure 3.15c shows the automatic real-time updates performed by the HoloLens on the merged map (blue color), and Figure 3.15d shows the merged map overlaid on the blueprint.

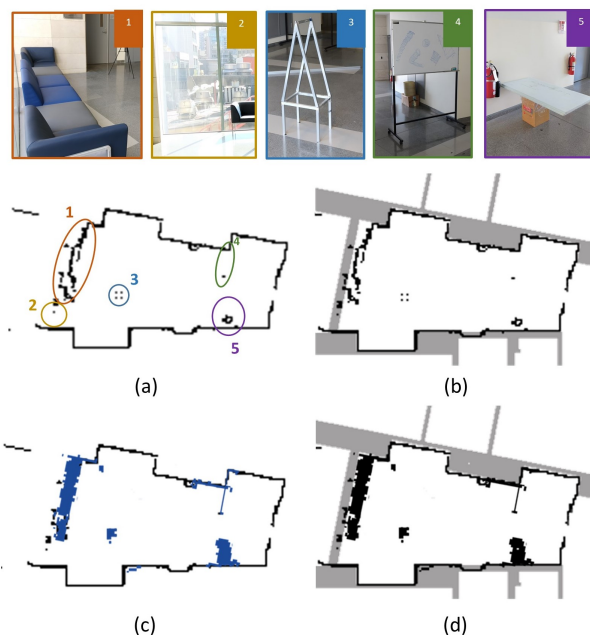


Figure 3.15: Produced maps during the experiments: (a) shows the map obtained by the robot where it was not able to detect the objects numbered 1 to 5, which is overlaid on the blueprint in (b). (c) shows the automatic real-time updates performed by the HL on the augmented map (blue color) merged with the robot's map, which is overlaid on the blueprint in (d).

Since the robot cannot traverse **Region II** and **Region III**, the supervisor walked around these regions while activating the AR-HMD Map Builder feature; the resulting AR-HMD map is shown in Figure 3.16a and the merged map is shown in Figure 3.16b. Here, the supervisor's judgment and SA lead to mapping the new area through the HoloLens. Glass walls are also invisible to the HoloLens, and the red circle in Figure 3.16b shows the largest HoloLens mapping error, which occurs because the HoloLens failed to perform tracking in this feature-deprived location. In such a case, human intervention is needed. In fact, the supervisor performed manual edits to correct for errors stemming from both the robot and the HoloLens. Figure 3.17a shows the edits performed by the supervisor being fused in the robot's map. To show the impact of the human edits, Figure 3.18 illustrates a comparison between the map built by the robot and the HoloLens on their own, and the same map being corrected in-situ by the human operator.

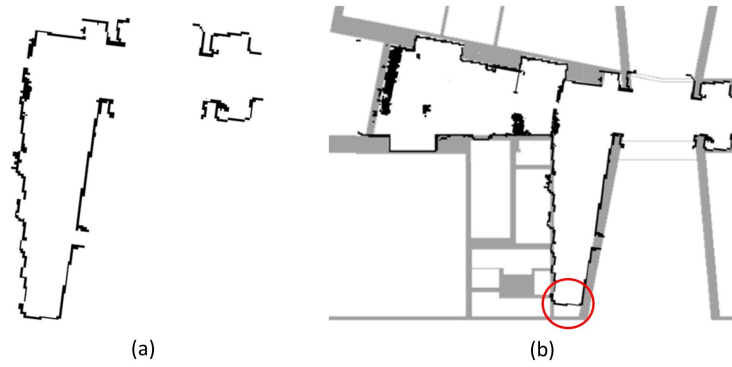


Figure 3.16: (a) Map produced by the AR-HMD Map Builder module in **Region II**. (b) the merged map of both **Region II** and **Region III**; the red circle shows a mapping error that occurred because the HL failed to perform tracking in this location.

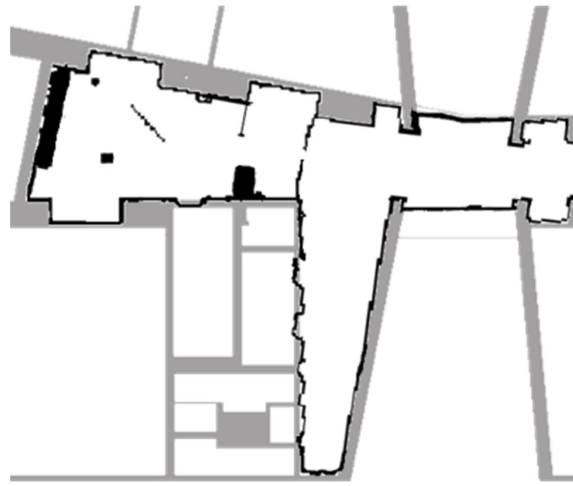


Figure 3.17: The final map produced by the proposed collaborative augmented SLAM system.

Quantitatively speaking, the entire experiment took **13 minutes and 50 seconds**: (1) mapping region **Region I** using the robot took 6 minutes and 15 seconds, 47 seconds of which the AR-HMD Map Builder was activated, (2) mapping **Region II** and **Region III** was completed in 3 minutes, and (3) the final human updates were performed in 4 minutes and 35 seconds. Moreover, all human edits and AR-HMD map updates are visualized instantly, while sending maps between the robot and the HoloLens took an average of 1.4 seconds, which is acceptable since the maps were sent/received based on the supervisor's request through the ARI. That said, performance can be improved by only sending the indices and values of the updated cells (instead of the whole map) between the agents.

To assess the above experiment's duration, another experiment was conducted in full autonomy, in which the robot was tele-operated to map **Region I**, then the robot was manually moved to map **Region II** and **Region III**. This procedure alone took 11 minutes 37 seconds, without accounting for the post-processing needed to add undetected obstacles and correct errors. The



Figure 3.18: Final map with human augmented edits (a), and without any human edits (b)

traditional method of tape-measuring the test area was used, and the map was edited offline using GIMP software. Measuring the test area took 8 minutes, and applying the edits in GIMP took 27 minutes since the supervisor has to measure pixels and apply edits. In summary, the traditional mapping and post-processing method required around **46 minutes**. Thus, the proposed MISA framework **eliminated post-processing** of the map and needed approximately **70% less time** than the traditional method. It is worth mentioning that all of the time measurements in the experiments were obtained by a timer that is set by the supervisor.

Although connectivity issues can pose a major challenge in collaborative SLAM, delays or short-loss of connectivity can be well tolerated by the MISA system. The reasons are: (1) the agents in the MISA system do not depend on each other to localize themselves, (2) the HoloLens is autonomous relative to keeping the map in place after being initialized, and (3) the maps are transferred between agents upon the supervisor's request.

Through the above experiments, the efficiency of the proposed MISA framework is demonstrated in collaborative SLAM systems by producing accurate maps in significantly less time. While the robot can produce accurate maps, it fails when obstacles do not intersect with the LiDAR's detection plane. Here, the HoloLens can add these obstacles owing to its spatial mapping capability. Moreover, having a human supervisor can mitigate the situations when both the robot and the HoloLens fail to accurately map undetected parts in the area of concern, and when either agent fails to localize in the environment.

Chapter 4

POC 2: MISA In Automated Puzzle Reconstruction

4.1 Introduction

The Jigsaw puzzle is a problem that requires reconstructing an image from non-overlapping pieces through examining both the shapes and/or color information of each piece. This 'game' was known since the 18th century; however, the first computation approach to solve a puzzle was introduced in 1964 [76]. Although solving puzzles might seem a trivial task, [77, 78] showed that this problem is NP-complete specially when determining adjacency through inter-piece compatibility. In addition to being an interesting and challenging game, the jigsaw puzzle problem could be extended to reconstructing objects that have been torn into smaller sets of fragments. For instance, researchers have applied puzzle solving techniques in different applications such as DNA/RNA modeling [79], speech de-scrambling [80], reassembling archaeological artifacts [81, 82], and reconstructing shredded documents, paints, and photos [83–86].

Jigsaw puzzles might come in different forms, some of which are square puzzles, interlocking puzzles, 3D puzzles, and others; regardless of its form, a jigsaw puzzle is generally solved depending on functions of colors, textures, shapes, and possible inter-piece correlations; thus, digital image processing and pattern recognition techniques are commonly used in this problem [87]. In recent years, most research on solving jigsaw puzzles has focused on image puzzles with square non-overlapping pieces (see Figure 4.1). Since all pieces are squares, solving these puzzles has to rely on the image information only without considering the shape. The difficulty of such puzzles varies depending on the type: pieces with unknown orientations, pieces with unknown locations, and pieces with unknown locations and orientations which is the hardest to solve.

Different approaches have been proposed to solve such jigsaw puzzles; however, the most common approaches are the global methods [88, 89] and greedy methods [76, 90]. Global methods, such as genetic algorithms, tend to maximize a global compatibility function to solve the puzzle; thus the performance of such methods highly depends on the choice of the function to be maximized. On the other hand, greedy methods starts by matching an initial pairwise of

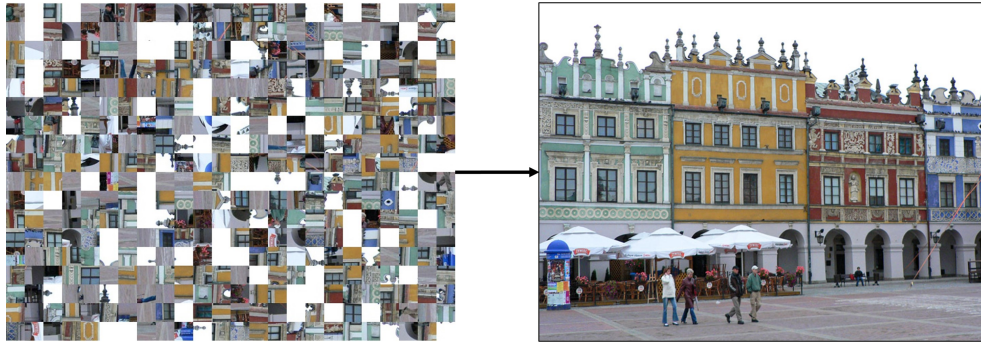


Figure 4.1: Example of an image puzzle with square pieces of unknown orientations and locations (right), and the result of successful reconstruction of the puzzle pieces (left).

pieces and then extending the matches till the puzzle is solved. In this method, the two fundamental steps are measuring pairwise compatibility scores for adjoining pairs of puzzle pieces, and setting the strategy of reconstruction [90]. Although compatibility scores were proven to be reliable and priority selection [90], they do not guaranty correct matching between pieces [91]. Moreover, most of greedy methods are prone to stuck in local minima since they do not have the possibility to move/rotate a piece that was already placed based on local compatibility, even if it showed to be globally misplaced. As for the reconstruction strategies, most of them are inspired from techniques that humans use. Some of these techniques are piece sorting, starting from edges, and reconstructing small clusters of similar features [92].

The proposed application of MISA in automated puzzle reconstruction addresses the challenges presented in solving image puzzles with square pieces of unknown locations and orientations, using the greedy approach. Through a custom-developed GUI, the human supervisor monitors the puzzle reconstruction results in real-time, intervenes based on his/her SA to correct global errors, and receives assistance requests from the agent when its self-confidence in local matches drops. In addition, the GUI communicates some internal states of the agent such as the reconstruction completion percentage, pieces that are being merged, and the location of the merged pieces in the reconstruction results. It is assumed that based on the SA, imagination, and cognitive power of a human, s/he can look at puzzle pieces or clusters and be able to detect errors, intervene, and assess matching options, thanks to their ability to see ‘the big picture’.

The rest of the chapter is structured as follows: Section 4.2 presents previous related work. Section 4.3 gives an overview on the baseline puzzle reconstruction approach and how the proposed framework is implemented in automated puzzle solving, while Section 4.4 describes the experimental setup, experiments, and results.

4.2 Related Work

The first algorithm to solve jigsaw puzzles was proposed by Freeman and Gardner [93] in 1964, where they focused on shape information in puzzle pieces. This was followed by several works that also considered only the shapes of pieces. Zisserman et. al [94] solved

puzzles through detecting "indents" and "outdents" in pieces and matching them based on bi-tangents and distinguished points. Another method for matching such pieces was proposed by [95], where they fitted ellipses to the indents and outdents and compared them to get the best matches. Authors in [96] applied the 'start with boundary' strategy in solving jigsaw puzzles by identifying the pieces belonging to the edges, and then forming the puzzle reconstruction as traveling salesman problem. All These approaches did not consider using color and image information in influencing the matching of pieces.

Using both edge shape and color information to influence adjoining puzzle pieces was first introduced in [97] where the authors compared the color similarity along matching edges of two pieces to decide on the reconstruction. In [98], a method that consists of extracting the edge features, calculating the histogram around the edge, and finally reconstruct the pieces through histogram intersection algorithm was proposed. Although this method resulted in more accurate reconstruction compared to the traditional histogram intersection methods, the experiments were limited to very small puzzles consisting of only 12 pieces. Other approaches proposed for such puzzles relied on isthmus global critical points [99], dividing puzzle pieces to groups of most likely 'interconnected' pieces [100], applying shape classification algorithms [101], and others.

Taking the challenge a step further, several researchers proposed algorithms to solve jigsaw puzzles of square pieces with unknown locations and orientations, thus the reconstruction of pieces depends only on image information. This type of puzzles was proved to be NP-complete problem [88], meaning that forming it as an optimization problem with a global energy function is very hard to achieve [102]. Shih et. al [103, 104] proposed an algorithm based on a modified Hausdorff distance method. In their method, they start by constructing a square of nine pieces. Then reconstruction is continued based on a square expansion method which keeps expanding each side alone till no more high confident matches are found for the side under reconstruction. To check the pairwise compatibility between neighbor pieces, [105] proposed to not only use edge color similarity, but also the content similarity. In [106], the authors proposed that the original image is given to the system, and they applied SIFT method to extract features from both the pieces and the original image itself and then match these two groups of features to re-assemble the pieces correctly. In [91], the authors proposed a method that considers the transmission relations between neighboring pieces, thus extending the compatibility check to four neighbor pieces instead of only two.

Cho et al. [88], proposed to solve square jigsaw puzzles through a graphical model and probabilistic approach that uses Markov Random Field and belief propagation. Through this approach, the authors were able to handle puzzles up to 432 pieces; however, anchor pieces should be given as prior knowledge to the system. This work claimed that most discriminative measurement for pairwise matching is the dissimilarity-based compatibility. Gallagher [106] also proposed a graphical model which solves puzzles through tree-based reassembly algorithm. They introduced a new compatibility metric which they called Mahalanobis Gradient Compatibility (MGC). This metric measures the local gradient near the puzzle piece edge, and was proved to have superior performance over previously proposed metrics. Zanoci and Anders [76], followed the lead of [90] and formulated the puzzle problem as search for minimum spanning tree (MST) in a graph. Moreover, they used Disjoint Set Forest (DSF) data structure

to guarantee fast reconstruction. One of the common limitations of these approaches is that the algorithm can't change the position of a piece that is misplaced early in the reconstruction process. Such mistakes lead to very poor performance of the algorithm.

There have been very few works that consider leveraging human capabilities to assist in puzzle solving; however, these methods focused on shredded documents reconstruction. For instance, [107] proposed a mixed-initiative approach in which the system shows a fragment to the human through a GUI. Based on his imagination, the human can draw what the neighboring piece might look like, then the system selects the most likely matches from the fragments set. This approach is not feasible for image-based puzzles consisting of complex shapes and patterns. In [108, 109], the system suggests the pieces to be matched and requests human approval. Here, the system asks for every match option, which increases the workload on the human supervisor.

4.3 System Overview

This section presents a brief description of the adopted baseline approach, which formulates the puzzle reconstruction problem as a search for MST, followed by an overview of MISA's implementation on top of this baseline.

4.3.1 Baseline Approach: Automated Puzzle Solver

To establish a baseline, the methods applied in [76, 89] are adopted. The baseline is referred to as Automated Puzzle Solver, where the problem is formed as a search for MST in a graph. The MGC metric is used for piecewise compatibility, and DSF is applied for reconstruction. This baseline can be summarized through the modules shown in Figure 4.2 and discussed below.

Pairwise Compatibility: starting from a set of square puzzle pieces with unknown positions and orientations, this module calculates the pairwise compatibility function for each possible

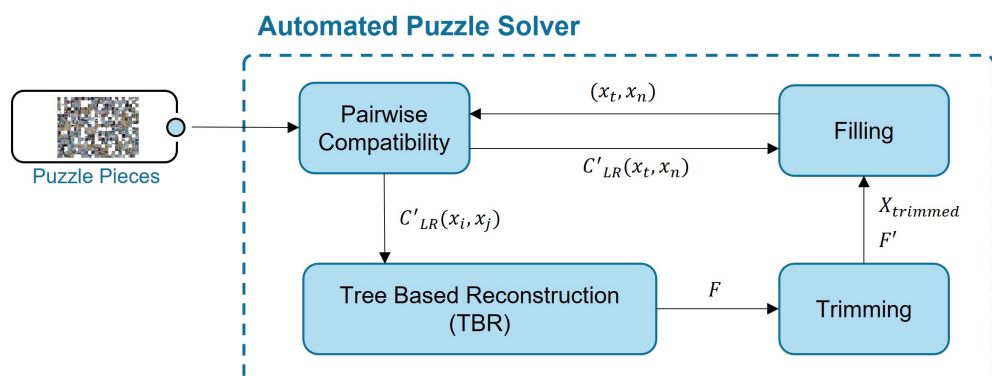


Figure 4.2: Flowchart representing the different modules in Automated Puzzle Solver baseline.

pair of pieces in the puzzle set. The MGC metric [76] is adopted to determine the similarity of the gradient distributions on the common boundary of the pair of pieces to be matched. Assuming that the compatibility measure $D_{LR}(x_i, x_j)$ of two puzzle pieces x_i and x_j , where x_j is on the right side of x_i , is to be calculated, the color gradients in each color channel (red, green, blue) are calculated near the right edge of x_i as follows:

$$G_{iL} = x_i(p, P, c) - x_i(p, P - 1, c), \quad (4.1)$$

where G_{iL} is the gradients array with 3 columns, c represents the three color channels, and P is the number of rows since the dimension of the puzzle piece (in pixels) is $P \times P$. Then, the mean distribution $\mu_{iL}(c)$ and the covariance S_{iL} of G_{iL} is calculated on the same side of the same piece x_i as:

$$\mu_{iL}(c) = \frac{1}{P} \sum_{p=1}^P G_{iL}(p, c), \quad (4.2)$$

The compatibility measure $D_{LR}(x_i, x_j)$, which calculates the gradient from the right edge of x_i to the left edge of x_j , is calculated as:

$$D_{LR}(x_i, x_j) = \sum_{p=1}^P (G_{ijLR}(p) - \mu_{iL}) S_{iL}^{-1} (G_{ijLR}(p) - \mu_{iL})^T, \quad (4.3)$$

where $G_{ijLR} = x_j(p, 1, c) - x_i(p, P, c)$. After that, the symmetric compatibility score $C_{LR}(x_i, x_j)$ is computed as:

$$C_{LR}(x_i, x_j) = D_{LR}(x_i, x_j) + D_{RL}(x_j, x_i), \quad (4.4)$$

where $D_{RL}(x_j, x_i)$ is calculated in a similar way to (4.3). These steps are applied to each possible pair of pieces, for all possible orientations of each piece ($0^\circ, 90^\circ, 180^\circ, 270^\circ$). The final step entails dividing each compatibility score $C_{LR}(x_i, x_j)$ with the second-smallest score corresponding to the same edge, resulting in the final compatibility score $C'_{LR}(x_i, x_j)$. This step ensures that significant matches of each edge has a score that is much smaller than 1, thus leading to re-prioritizing the compatibility scores in a way where pieces edges, which are more likely to be a ‘correct match,’ are chosen by the reconstruction algorithm in the very early steps.

Tree-based Reconstruction (TBR): this module is responsible for reconstructing the puzzle pieces through finding a minimum spanning tree (MST) for a graph representation of the puzzle $G = (V, E)$ [76, 90]. For that, pieces are treated as vertices, and compatibility scores ($C'_{LR}(x_i, x_j)$) are treated as weights of edges (e_{ij}) in the graph. Each edge has a corresponding configuration between the two vertices (the orientation of each piece). To find the MST that represents a valid configuration of the puzzle, a modified Disjoint Set Forest (DSF) data structure is applied as follows:

The TBR module initializes with forests (clusters) equal to the number of puzzle pieces, and each forest having an individual vertex (v_i) corresponding to a puzzle piece. Each forest records the local coordinates and the orientation of each member puzzle piece (vertex). At the beginning of every iteration, the edge representing the lowest compatibility score (e_{min}) is selected in order to join the corresponding two vertices (v_i, v_j) in one forest. If the two vertices are already in the same forest (*i.e.*, pieces belong to same cluster), or matching the pieces leads

to collision in their corresponding clusters, the edge is discarded and appended to unused edges list (E_{unused}). This step is referred to as **Collision check** step. If Collision check is passed successfully, the two pieces are assembled, together with their corresponding clusters. Meaning that the two vertices are joined into a single forest, and e_{min} is removed from the main set of edges E and added to the set of edges in this forest; here, the local coordinates and orientations of each piece are updated. This step is referred to as **Merging** step. The process is repeated till all pieces are assembled in one cluster F ; meaning that all vertices belong to the same forest.

Trimming: since the TBR does not take into account the original image dimensions, the resulted image is not guaranteed to have a rectangular shape. To account for this error, trimming is applied to ensure a final image with same dimensions as the original one. In this module, a frame with size equal to the original image is moved over the assembled puzzle to determine the location of the portion with the highest number of pieces; this frame is tested for both portrait and landscape orientations. When the frame with most number of pieces is defined, all pieces outside this frame are trimmed from the image, resulting in an updated puzzle cluster F' . The trimmed pieces forms a set $X_{trimmed}$, which is sent along with F' to the Filling module.

Filling: here, pieces in $X_{trimmed}$ are used to fill all the holes in the reconstructed image F' . Filling starts from holes with highest number of neighbors and keeps going till all the holes are filled. For each hole, the compatibility score ($C'_{LR}(x_t, x_n)$) is calculated between each edge of each piece x_t in $X_{trimmed}$, and the edges of each of the hole neighbor pieces x_n . Then the holes are filled based on this score. In other words, each hole is filled by a piece whose edges has best compatibility score with the adjacent edges corresponding to the hole's neighbor pieces.

4.3.2 Applying MISA in Automated Puzzle Reconstruction

This section describes how the proposed MISA framework is applied in automated puzzle reconstruction, following the generic methodology presented in Chapter (2). Table 4.1 summarizes the inputs form the puzzle reconstruction problem, which are used to satisfy the MISA framework.

- **Tasks to be supervised by the human:**

Based on the surveyed literature and the full-autonomy experiments presented in Section 4.4.2, the main challenges that face the baseline greedy autonomous puzzle solver are: the high probability of incorrect matches when puzzle pieces are poorly-textured, the inability to asses the global consistency of the reconstructed puzzle, and the possibility of calculating wrong location/orientation of the trim frame. Thus, the human supervisor should be able to supervise and intervene in the following: (1) matching of two pieces with low image features (not enough texture or color variation in the piece), (2) global puzzle consistency and correctness (detecting globally misplaced pieces), and (3) trim frame location and orientation.

- **Robot SC attributes:**

As discussed later in Section 4.4.2, the possibility of having a false-positive match increases as the texture complexity in a puzzle piece decreases. This texture complexity is

Table 4.1: MISA attributes in automated puzzle reconstruction.

Tasks to be supervised		Attributes		
SC-based tasks	SA-based tasks	SC Attributes	SA Attributes	MISA-based UI
<ul style="list-style-type: none"> • Matching of poorly textured pieces 	<ul style="list-style-type: none"> • Global puzzle reconstruction • Trim frame location and orientation 	<ul style="list-style-type: none"> • Confidence in local pieces merging • SC-metric: Piece entropy 	<ul style="list-style-type: none"> • Global puzzle consistency • SA-tools: <ul style="list-style-type: none"> – Show reconstruction results – Show pieces responsible for the matching and their locations in the reconstructed cluster – Show the proposed trim frame 	<ul style="list-style-type: none"> • 2D GUI

measured through the *Entropy* metric. In short, when the complexity of the texture in the image increases, the *Entropy* value increases, and vice versa. Thus, the *Entropy* of a puzzle piece is used as SC-metric. When this SC-metric drops below a certain threshold that is defined experimentally, the agent asks the human supervisor to approve or decline the match decision. In case no response is received within a pre-defined time limit, the agent proceeds with the match. This ‘no-response’ strategy is selected because even if merging the clusters resulted in misplaced pieces, the human can detect the error and fix it through SA-based intervention at a later stage.

• **Human SA attributes:**

Since SC attributes could not be defined to capture globally misplaced pieces, the human supervisor should leverage his/her SA and judgment abilities to assess the global consistency in the reconstructed puzzle. The *SA tools* proposed to enhance human SA here are summarized in showing the supervisor the reconstruction results, highlighting the pieces responsible for merging in addition to their location in the resulted cluster, and showing the supervisor the proposed trim frame when reconstruction is done. These tools allow the supervisor to detect misplaced pieces and relay this information back to the agent. Moreover, the supervisor is allowed to re-locate or re-orient the trim frame if needed.

• **MISA-based user interface:**

A user-friendly GUI is developed to apply the SA tools discussed above. The GUI displays the progress of the reconstruction progress to the supervisor, communicates with him/her the system’s requests for assistance, and furnishes certain internal states that help the supervisor intervene in the process.

Figure 4.3 presents the block diagram of MISA framework applied in the automated puzzle solver baseline. Since **Automated Puzzle Solver** block is already described in section (4.3.1), the **MISAC** block only is discussed below.

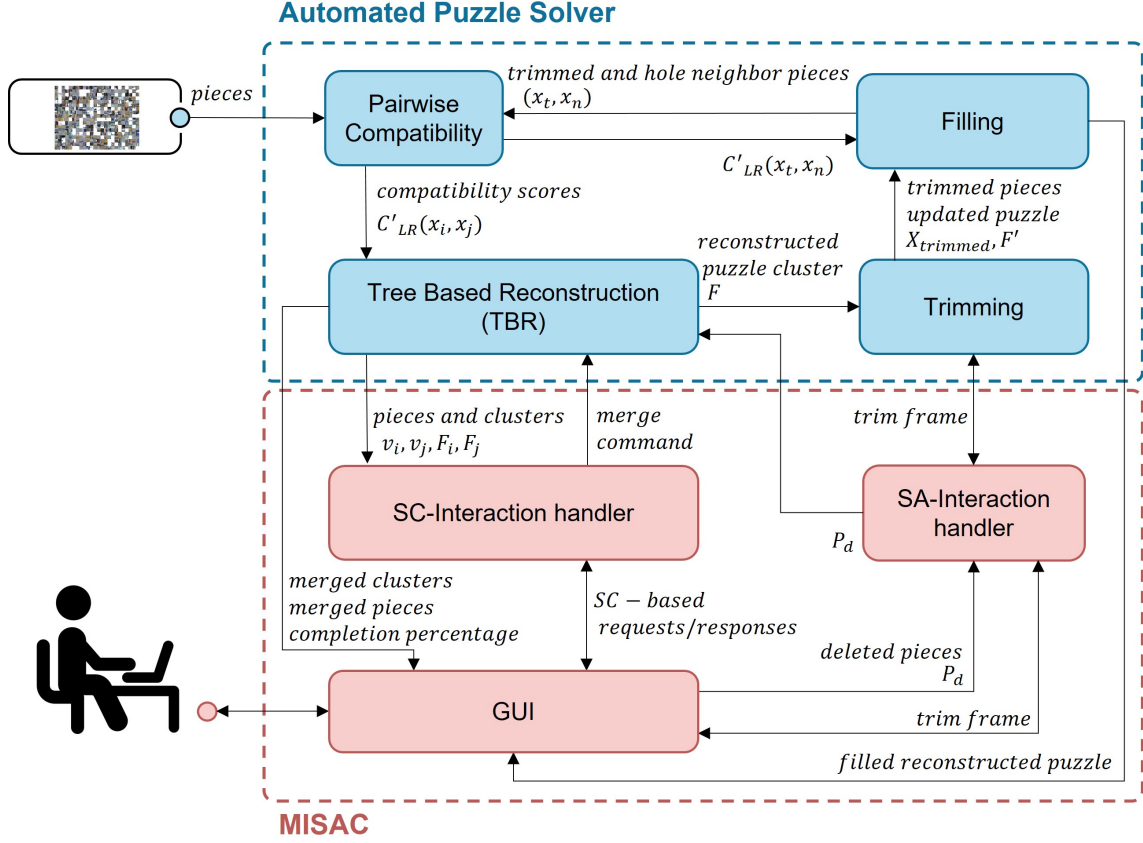


Figure 4.3: Block diagrams of MISA in automated puzzle reconstruction.

4.3.2.1 SC-Interaction handler

This module modifies the structure of TBR (discussed in Section 4.3.1) to allow for SC checking, requesting human assistance, and conveying the human response back to TBR. When the two pieces to be merged (v_i, v_j), along with their corresponding clusters (F_i, F_j), pass the Collision check, this module checks if either piece has an *Entropy* below the pre-defined confidence threshold $C_{th,p}$. *Entropy* is obtained from the Gray Level Co-Occurrence Matrix (*GLCM*), which extracts second order statistical texture features from images. *GLCM* is a square matrix where the number of rows and columns equals the number of gray levels in the image. Each element $GLCM(n, m)$ of the matrix is represented as a joint probability distribution function $P(n, m|d_{pixel}, \theta)$, which is the probability of appearance of two pixels with values k_n and k_m , separated by distance d_{pixel} at an orientation angle θ . After the *GLCM* is obtained, the entropy is calculated as follows:

$$Entropy = - \sum_{n=0}^{M-1} \sum_{m=0}^{M-1} P(n, m|d_{pixel}, \theta) \log_2 P(n, m|d_{pixel}, \theta) \quad (4.5)$$

where $P(n, m|d_{pixel}, \theta) = GLCM(n, m)$.

Since each piece of the puzzle can be placed in four different orientations, the module

calculates the entropy corresponding to the orientation of each piece where $\theta = 0^\circ, 90^\circ, 180^\circ$, or 270° ; $d_{pixel} = 1$ in all calculations.

If both pieces pass the SC check, the SC-Interaction handler commands TBR to proceed with merging the clusters. Otherwise, this module performs a temporary merge, shows the merging result (cluster image) to the supervisor through the GUI, and requests human assistance in approving the merge. After requesting assistance, the SC-interaction handler awaits human response for a defined period of time. If the human approves the temporary merge, TBR is commanded to proceed with merging. If the human declines the merge, merging is undone and the corresponding e_{min} is deleted from E in TBR. In case no response is received within this time limit, TBR is commanded to proceed with merging.

4.3.2.2 SA-Interaction handler

In addition to approving or declining the merge upon the agent's request, the supervisor can select pieces that s/he deems as misplaced in the resultant image based on his/her SA and judgment. When the supervisor selects these pieces through the GUI, SA-Interaction handler sends the deleted pieces list (P_d) to TBR in order to delete the corresponding vertices from the shown cluster image. Here, TBR deletes the vertices from the displayed forest and forms new forests, using corresponding edges from E_{unused} , where each forest contains one vertex corresponding to one deleted piece with a specific orientation; resulting in four vertices for every piece. The vertices of the newly formed forests are connected to the other vertices in the graph through edges corresponding to the compatibility scores as discussed before. Allowing the supervisor to inform the system about misplaced pieces is crucial since not all errors occur due to low entropy, as some pieces might have complex texture (high entropy) and still be misplaced by the system.

Moreover, this module shows the trim frame to the supervisor when reconstruction is done. If the supervisor modifies the location/orientation of the proposed trim frame, SA-Interaction handler sends the new frame to the Trimming module. Otherwise, it commands the trimming module to proceed with the proposed frame.

4.4 Implementation and Experiments

4.4.1 Experimental Setup

The proposed MISA framework is implemented in Python3, and the GUI is developed in pyQT. The implementation flowchart is shown in Figure 4.4. When the system is started, the compatibility score $C'_{LR}(x_i, x_j)$ is calculated for every possible pair of pieces, and the *Entropy* is calculated for every piece in the puzzle set. TBR initializes with clusters of single pieces, meaning that the number of clusters equals the number of pieces in the puzzle set. As long as number of clusters is higher than one, edge e_{min} with the smallest weight C'_{LR} is selected from E to check its corresponding pieces for merging. If the two pieces are already in the

same cluster, or merging them results in collision between their corresponding clusters, e_{min} is removed from E and appended to E_{unused} , a list containing all edges that are not used for merging. Otherwise, the *Entropy* of each piece, given its orientation, is checked against the confidence threshold $C_{th,p}$.

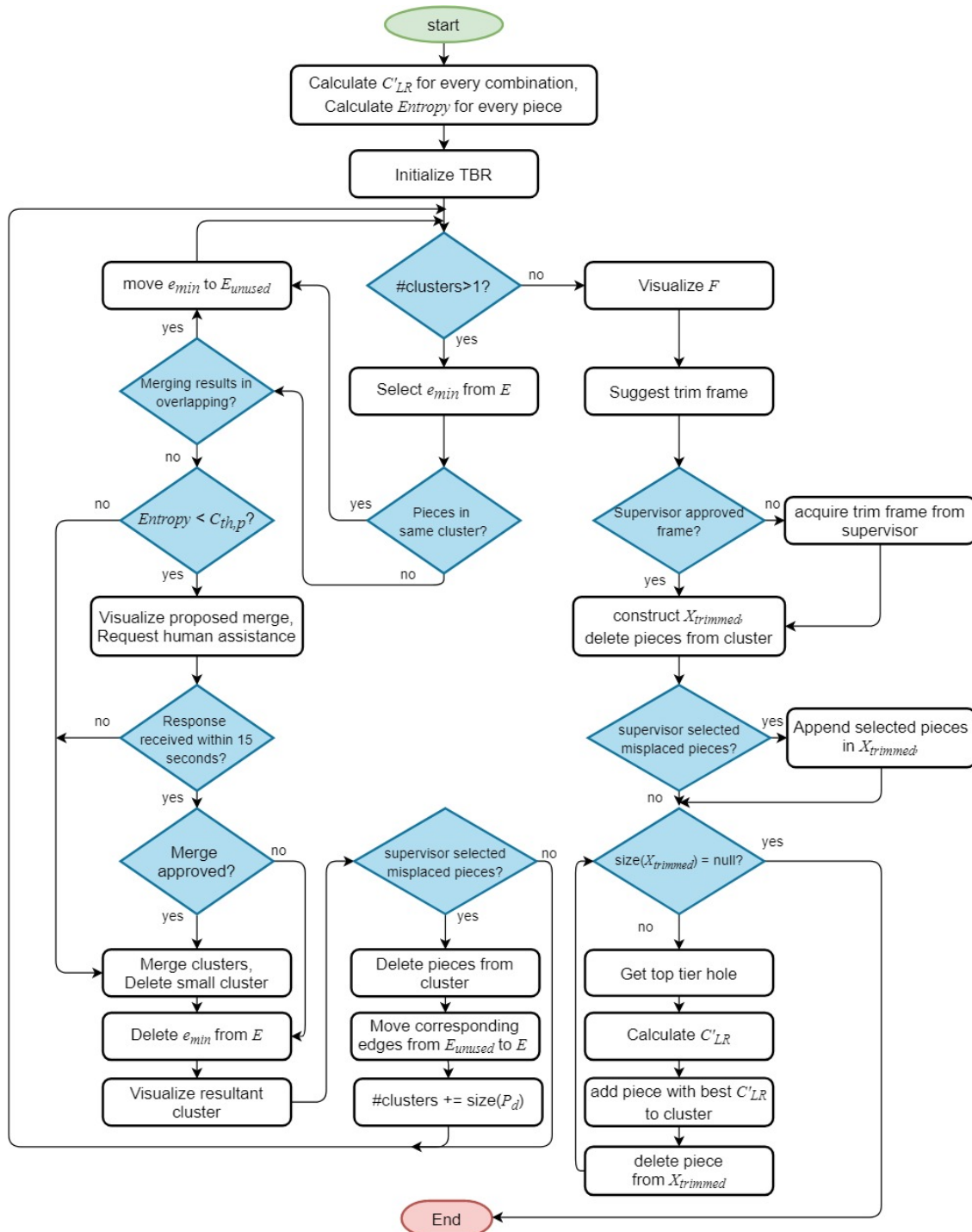


Figure 4.4: Flowchart presenting the implementation logic of MISA in automated puzzle reconstruction.

If the *Entropy* of both pieces is higher than $C_{th,p}$, the clusters of the corresponding pieces are merged, the smaller cluster is deleted, e_{min} is deleted from E , and the resultant cluster is visualized on the GUI. Through merging, pieces belonging to the small cluster are moved to the larger cluster, which might require changes in orientations of the pieces. In result, every merge decreases the number of clusters by one. If the *Entropy* of either piece is below $C_{th,p}$, meaning that SC in the match option is low, the proposed merge is visualized through the GUI, and the human supervisor is requested to approve or decline the merge. If the supervisor approves, the same merging procedure is applied. In case the supervisor declines the proposed merge, e_{min} is deleted from E and the proposed merge is not applied.

Whenever a cluster is visualized on the GUI, the supervisor can select pieces that are found to be misplaced based on his/her SA. The selected pieces are deleted from the visualized cluster and added again as single-piece clusters in TBR. Moreover, corresponding edges in E_{unused} are moved to E to be used in the reconstruction process again.

When the system arrives at one cluster only, meaning that all puzzle pieces are reconstructed in a single cluster image, the trim frame is calculated and visualized over the resultant cluster F through the GUI. The supervisor can either approve or re-locate/re-orient the suggested trim frame. Either way, all pieces outside the trim frame are deleted from F and appended in list $X_{trimmed}$. At this step, the supervisor is allowed to select misplaced pieces in the cluster image. These pieces are also deleted from the cluster and appended to $X_{trimmed}$. As long as $X_{trimmed}$ is not empty, the system searches the cluster for the hole with the highest number of occupied neighbors (top tier hole); then it calculates the compatibility score between all edges of each piece in $X_{trimmed}$ and the free edges of the neighbor pieces of the hole. In other words, the system checks all orientations of each trimmed piece in the selected hole. The piece configuration that results in the lowest compatibility score is placed in the hole and deleted from $X_{trimmed}$.

Figure 4.5, shows the GUI developed for MISA in automated puzzle reconstruction. The GUI displays the results of the reconstruction in real-time (a resultant cluster image is shown in the ‘output’ in the GUI), the percentage of completion of the reconstruction process, in addition to the logs/errors/warnings. Moreover, and to help the supervisor decide whether to decline the merge or delete misplaced pieces, the GUI shows the supervisor the two pieces that are responsible of merging, and the location of these two pieces in the resultant cluster image. Below is a description of the main components of the GUI:

- **Select image:** allows the supervisor to select an image to be divided into puzzle pieces and then reconstructed,
- **Start reconstruction:** starts the reconstruction process discussed earlier,
- **Rotate CW:** rotates the resultant cluster image by 90° ,
- **Zoom in:** zooms in the resulted cluster image ($\times 2$),
- **Zoom out:** zooms out the resulted cluster image ($\div 2$),
- **Show boxes:** shows boxes on the cluster image highlighting the pieces responsible for the merging,

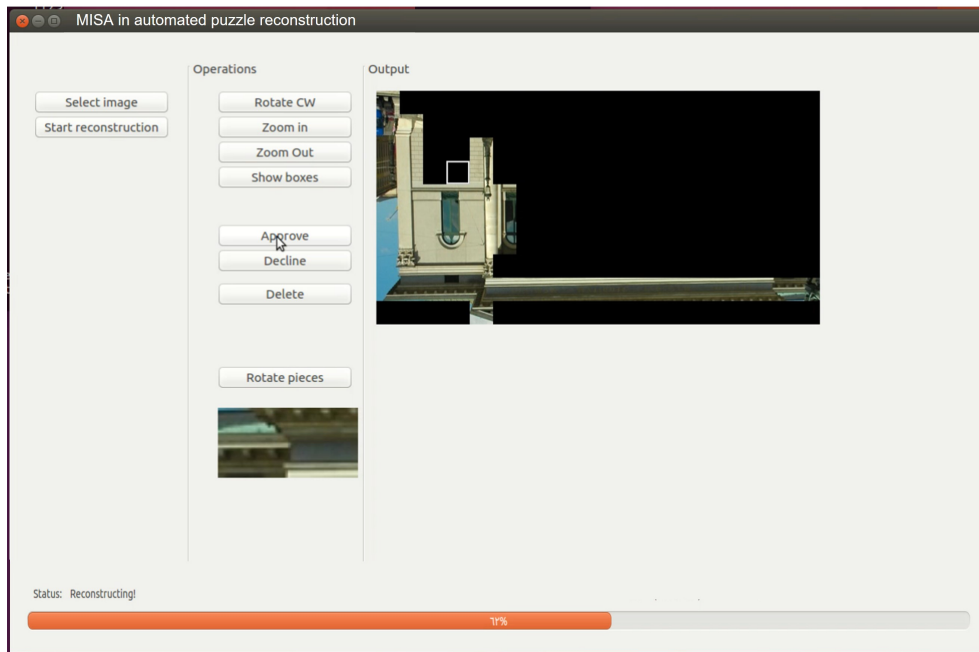


Figure 4.5: The MISA-based UI developed for automated puzzle reconstruction: Reconstruction phase.

- **Approve:** allows the supervisor to approve the proposed merge when requested,
- **Decline:** allows the supervisor to decline the proposed merge when requested,
- **Delete:** when pressed, it allows the supervisor to select misplaced pieces by moving the square cursor on the cluster image and clicking on the pieces,
- **Rotate pieces:** rotates the image placed below the button by 90° clockwise. This image shows only the two pieces that were responsible of merging the clusters. These pieces are shown to help the supervisor decide whether to decline the merge or delete misplaced pieces,
- **Status:** communicates the current stage of reconstruction in addition to some logs, errors, and warnings,
- **Status bar:** shows the percentage of reconstruction completion based on the number of existing clusters in TBR.

When the final cluster is obtained, the GUI displays the proposed trimming frame to be approved or re-located/re-oriented by the supervisor, as shown in Figure 4.6. Here the **Approve** button is replaced by **Approve frame** and the **Decline** button is replaced by **Edit frame**. The supervisor can either approve the trim frame, or re-locate/re-orient it within the cluster image.

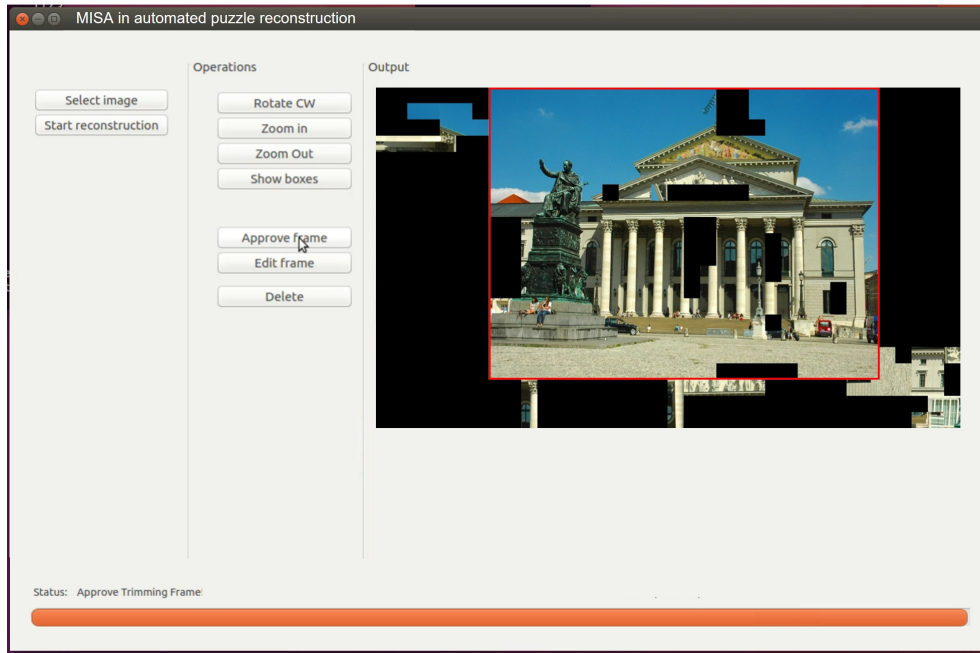


Figure 4.6: The MISA-based UI developed for automated puzzle reconstruction: Trimming phase. The suggested trim frame is visualized in red over the resultant cluster image.

4.4.2 Experimental Testing and Results

Experiment-1: the first set of experiments was conducted to test the performance of the baseline system in full-autonomy mode in order to define the MISA framework parameters. Tests were performed using the MIT dataset published in [88]. This dataset consists of 20 images and is used as a benchmark in most of the available references in the related literature. In these tests, every image in the set is divided to 432 pieces with 28×28 pixels each, the pieces are given random locations and orientations, and the fully-autonomous reconstruction process is started. Figure 4.7 shows an example of how the clusters are merged: the reconstruction initializes with 6 single-piece clusters. In the first iteration, C6 is rotated and merged with C1, then it is deleted from the puzzle pieces set (forest). In the second iteration, C4 and C5 are

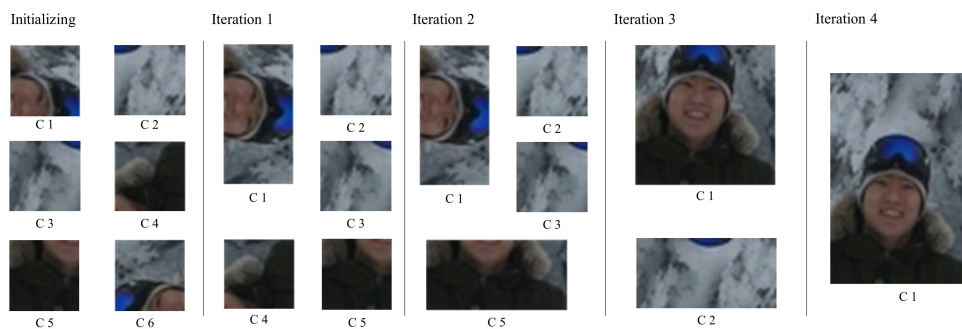


Figure 4.7: Demonstration of the cluster merging process. The figure shows how clusters are being merged (rotation and translation) to form a final combined cluster image.

merged, resulting in two clusters containing two pieces and two clusters with one piece each. In iteration 3, C1 and C5 are merged together, resulting in C1 with four pieces; in this iteration C1 is rotated to fit with C5. Finally, in iteration 4 the two remaining clusters are merged in one final cluster.

To assess the reconstruction results, two types of accuracy measures are calculated. The **direct comparison metric** (DCM) compares the exact position and orientation of each piece with the original image, while the **neighbor comparison metric** (NCM) compares the number of times that two pieces, which are adjacent (with the same orientation) in the original image, are adjacent in the final reconstructed image. To better understand these metrics, Figure 4.8 presents sample results for two image puzzle reconstructions. These results shows that DCM reports very low scores if the reconstructed image has a wrong orientation (Figure 4.8a), or if its pieces slightly shifted in a certain direction (Figure 4.8b). In contrast, NCM shows more robustness to such conditions. It is important to mention here that the wrong orientation in the reconstructed image in Figure 4.8a is due to a wrong orientation of the trim frame that was calculated autonomously. Experiment-1 row in Table 4.2 presents the average NCM and DCM scores for the 20 images tested in this experiment.

Through carefully evaluating the baseline reconstruction performance, three main points were concluded:

1. In cases where the pieces to be merged are low in texture and have highly similar colors at the edges, the pairwise compatibility score will be low and TBR would merge the two pieces although they are not adjacent in the original image. We refer to this case as "false-positive" match, and it leads to local minima that affects the accuracy as the puzzle is being reconstructed.
2. If two pieces were matched based on low pairwise compatibility, the system does not have the tendency to detect if these pieces are globally misplaced, and thus it is unable to move or rotate the misplaced pieces. In such cases, the error accumulates as the puzzle is reconstructed.
3. Since the puzzle is reconstructed without dimension/orientation constraints, the trim frame



Figure 4.8: Two sample results of puzzle reconstruction through the fully autonomous baseline approach.

calculated by the system might have wrong location/orientation based on the accuracy of the reconstructed image. Thus, the final result after re-filling the puzzle might not match the original image.

For conclusion points (2) and (3), the system does not have the means to measure its SC in the global puzzle consistency nor in the trim frame correctness because the baseline approach does not have access to the original image; while point (1) indicates that the confidence in a successful merging of puzzle pieces can be captured through measuring the texture complexity of these pieces. *Entropy* of an image is a metric that measures the disorder or complexity of textures in it; thus, *Entropy* was evaluated as a SC-metric. For this, 20 reconstruction test were done using same conditions as Experiment-1. For each iteration in every test, the *Entropy* of the pieces merged was recorded, and a ‘merge success’ label was given by an expert supervisor. After collecting the data, the confidence threshold $C_{th,p}$ was varied between the lowest and highest recorded *Entropy* values, and the False Positive (FP) and False Negative (FN) rates were calculated. It was found that setting $C_{th,p} = 1.2$ resulted in the lowest FP and FN rates, which were 14.6% and 17.9% respectively. As a result, *Entropy* proved to be a valid SC-metric, and $C_{th,p}$ is set to 1.2.

Experiment-2: this set of experiments was conducted to test the performance of MISA in automated puzzle reconstruction. Tests were performed using the MIT dataset [88]. In each test, the supervisor selects an image from the set through the GUI, then the reconstruction process is started. The supervisor was told not to approve any proposed merge unless he’s totally confident that the match is correct. Moreover, he was instructed to select the misplaced pieces whenever detected. Here it is important to mention that the supervisor has detailed knowledge about the GUI and how the algorithm works. Three runs were performed on each image from the set, resulting in 60 tests. Figure 4.9, shows an example of the effect of acquiring human assistance on the accuracy of puzzle reconstruction. Figure 4.9a presents two clusters that the system is suggesting to merge based on low compatibility score between two pieces, each from one cluster. However, the two pieces have $Entropy < C_{th,p}$, so the supervisor

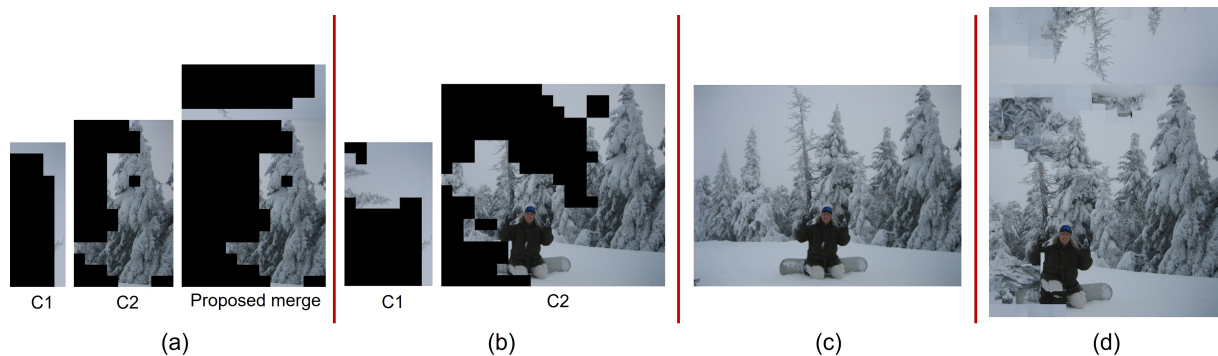


Figure 4.9: The effect of the supervisor’s assistance when the system’s confidence drops. (a) shows the proposed merge with the corresponding clusters, (b) shows how the clusters are growing separately after the supervisor declines the proposed merge, (c) shows the final reconstructed cluster, and (d) shows how the reconstruction would end if the human had approved the proposed merge.

approval is requested. Figure 4.9b shows how every cluster is growing on its own after the supervisor declined the merge. Figure 4.9c shows the final result of reconstruction, while Figure 4.9d shows how the final result would be if the supervisor was to approve the merge. Another example that shows the effect of human SA-based assistance is presented in Figure 4.10. Figure 4.10a shows a cluster with a misplaced piece, Figure 4.10b shows the final reconstruction result if the supervisor did not notify the system about the misplaced piece, and Figure 4.10c shows the final result considering the human assistance in detecting the piece. Experiment-2 row in Table 4.2 shows that the average NCM score obtained in this set of experiments is increased by 6.4%, while the average DCM score is enhanced by 11.6%.

To further show the effect of having MISA, four experiments were conducted in which the supervisor is only allowed to approve or decline the merge result when the system’s SC drops. This means that the supervisor can only intervene upon the system’s request; so the system cannot benefit from the supervisor’s better SA and judgment to correct errors that it cannot detect. Visual results are shown in Fig. 4.11. The first column (left) shows results of reconstructions from Experiment-1, the second column (middle) shows results of the supervisor **only** approving/declining upon the system’s request, while the third column (right) shows results of Experiment-2.

The presented results demonstrate that integrating MISA in automated puzzle reconstruction, where the supervisor can intervene based on his/her SA **and** the system’s Sc-based requests, results in superior performance as compared to a fully autonomous puzzle reconstruction or a system that only asks for human assistance based on its self-confidence. Moreover, the results in Table 4.2 shows the superior performance of the proposed MISA system compared to results from other approaches in the literature (where the same MIT dataset [88] is used).



Figure 4.10: The effect of the supervisor’s SA-based assistance. (a) shows a cluster with a misplaced piece, (b) shows how the final reconstruction looks like if the human did not select the misplaced piece to be deleted, and (c) shows the final reconstructed cluster considering the human assistance in detecting the misplaced piece .

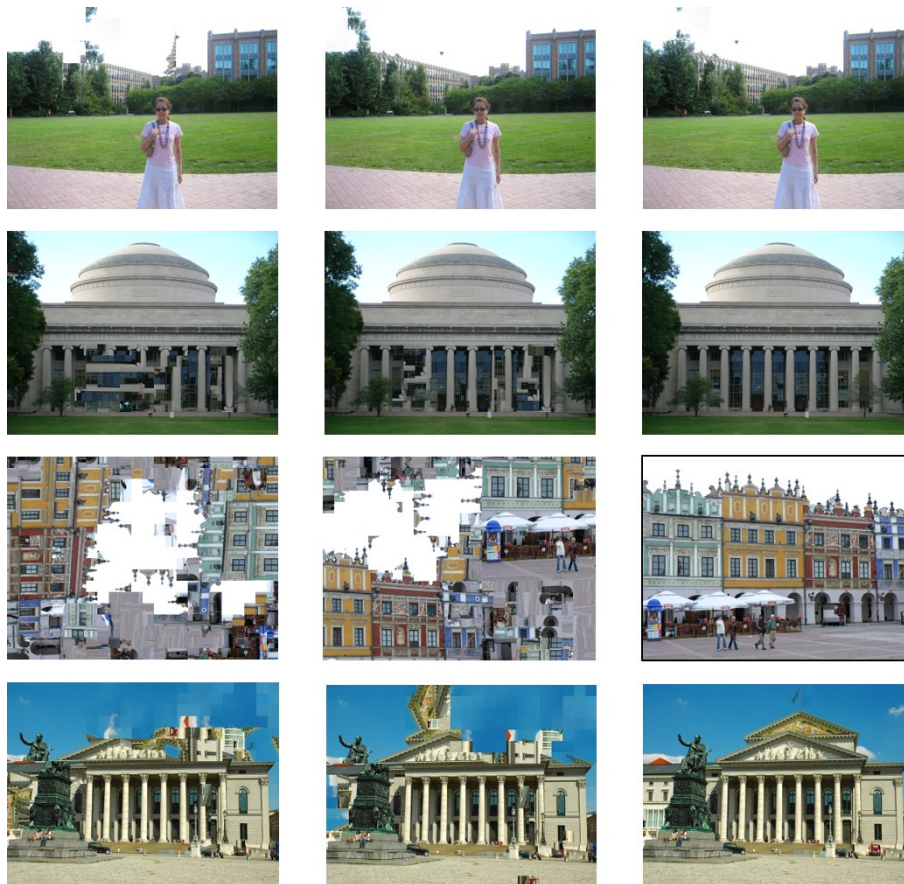


Figure 4.11: Visual results of sample experiments. The left column shows results using the baseline approach, the middle column shows the results obtained through running the proposed system but the supervisor was allowed only to approve/decline merge options upon request, and the column to the right shows the results obtained by the proposed MISA in automated puzzle reconstruction.

Table 4.2: Comparison of experimental results from MISA in automated puzzle reconstruction, and four state-of-the-art approaches.

Experiment / Approach	Year	DCM	NCM
Experiment-1	2021	84.9%	90.8%
Experiment-2	2021	96.5%	97.2%
MTS with DSF [76]	2016	88.5%	92.9%
Linear Programming [89]	2015	95.3%	95.6%
Loop Constraints [110]	2014	94.7%	94.9%
Constrained MST [90]	2012	82.2%	90.4%

Chapter 5

POC 3: MISA in Robot Grasping

5.1 Introduction

Robot grasping is a fundamental task in a wide variety of applications such as pick-and-place [111], bin-picking [112], stacking [113], assembly [114], and others. A grasping task (Figure 5.1) usually consists of four main sub-tasks in the sense-plan-act paradigm: a) detecting the target object, b) identifying the optimal grasping pose, c) generating the manipulator arm trajectory plans, and d) executing the grasp. Although different methods have been developed for each sub-task, end-to-end robot grasping is still considered a complex and challenging task in robot manipulation [115, 116], due to the fact that every sub-task is prone to some errors and limitations.

Detecting the target object accurately and identifying the best grasp pose depend on the robot's ability to understand the object information (*e.g.*, pose and shape) through perception. Various grasping approaches assume that perception works perfectly or with little uncertainty [117, 118]; however, machine perception is prone to errors and might not generalize to novel objects that have not been seen before [119, 120]. Errors in detecting the object and identifying the grasp pose occur due to complex backgrounds, partially observed objects, illumination conditions, object/background material properties, etc. Moreover, the ability of a robot to generate a valid approach-and-grasp plan, and execute it, is hindered by several challenges and limitations such as out-of-reach objects, presence of obstacles, kinematics inaccuracy, and camera-arm calibration errors, among others.

Robotic manipulator suppliers minimize these challenges and limitations by using high-precision actuators and sensors, advanced perception systems, and precise machining of components. However, such measures increase the cost of robots and affect their adaptability outside industrial settings. To increase robotic manipulators' affordability and deployment, researchers are aiming to develop low-cost robots using lower quality actuators and sensors [121–124]. Although learning-based approaches are being developed to enhance their performance and accuracy [124, 125], low-cost manipulators still suffer from perceptual and calibration errors, inaccurate kinematics, and manipulation execution errors [124]. These are mainly caused by inherited system errors like manufacturing and assembly faults, backlash in gears, and compo-

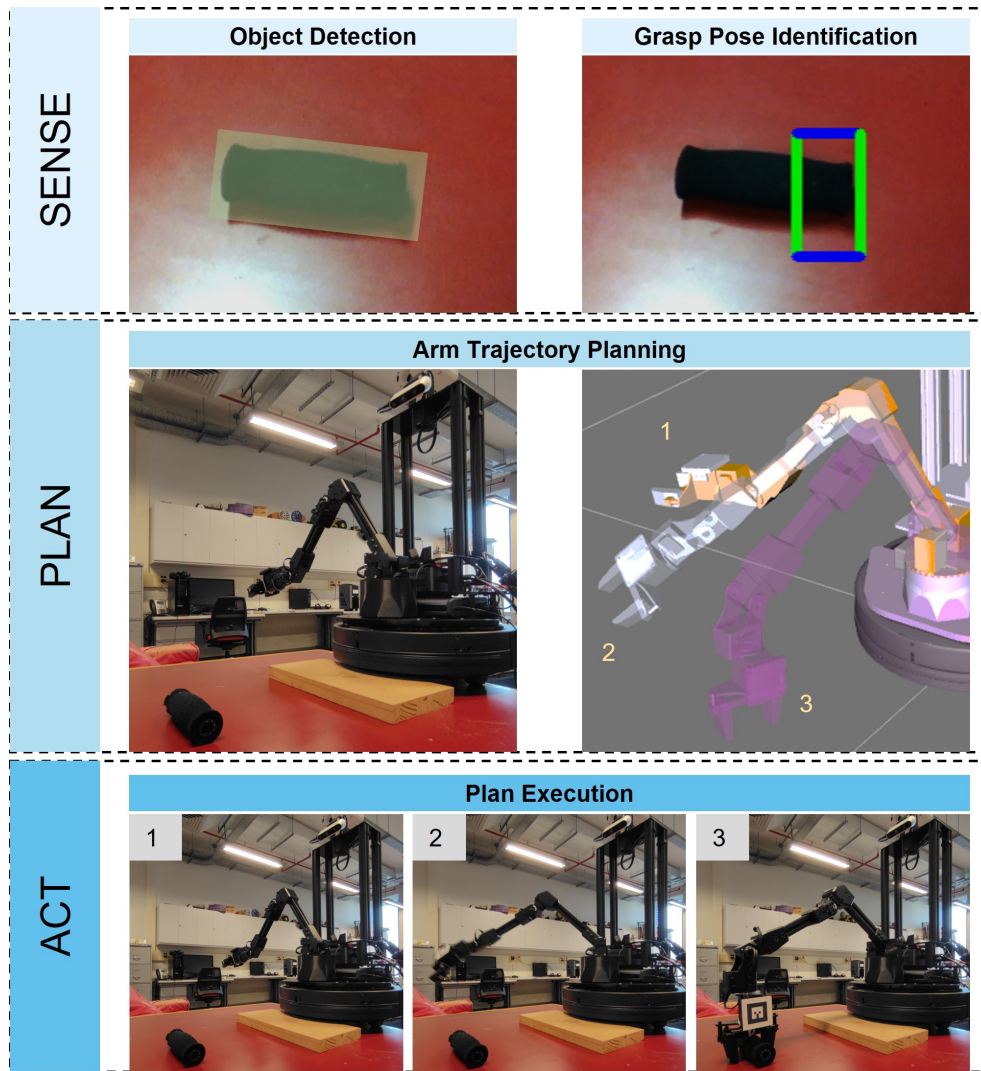


Figure 5.1: Sub-tasks of a robotic grasping task. Object detection and grasp pose identification (top) belong to the *sensing* primitive of the task, arm trajectory planning (middle) belongs to the *planning* primitive, and plan execution (bottom) belongs to the *acting* primitive.

nents wear and tear [125]. Other causes for errors in data-driven approaches are that datasets are mostly collected using simple backgrounds in environments that does not change dynamically [124], which does not reflect real-world conditions.

The proposed application of MISA in robot grasping addresses the identified challenges, limitations, and errors that face robot grasping tasks by including a human in the loop. The goal is to increase the success rate of robot grasping tasks by (1) increasing the accuracy of object detection and grasp pose identification through SC-based assistance, and (2) overcoming the limitations and challenges that face arm trajectory planning and plans execution through SA-based assistance. Thus, this POC contributes in the steps it proposes to enhance the performance of manipulators on one hand, and in expanding the use of low-cost manipulators beyond research labs by making them more reliable on the other hand.

The rest of the chapter is structured as follows: Section 5.2 presents previous related work. Section 5.3 gives an overview on the baseline autonomous grasping approach and how the proposed framework is implemented in robot grasping, while Section 5.4 describes the experimental setup, experiments, and results.

5.2 Related Work

Accurate object detection and grasp pose identification are crucial for a successful grasp, thus several researchers have leveraged the human’s superlative perception and cognition abilities to aid robots in these challenging sub-tasks. In [126] and [127], human operators aid the perception systems of robot manipulators by manually segmenting the target object through a GUI. Similarly, in [128] the user sketches the target object geometry and the actions that could be performed on that object for the robot. Such systems depend on the human input to accurately localize the target object.

Rather than acquiring the accurate object location from human input, other approaches depend on human assistance when the robot fails to detect its target object. In [129], the robot requests more information (*e.g.*, order, direction) if it is not able to identify the object. Instead of explicitly requesting assistance, the robot in [130] communicates its uncertainty through slow movement or waiting before executing the grasp plan to allow the human to correct the grasp goal. Moreover, [131] presented a paradigm where a human corrects falsely detected objects and identifies un-recognized objects through pointing and voice commands. Such systems benefit from human assistance in confirming the object detection; however, a successful grasp is not guaranteed since the remaining grasping sub-tasks are performed autonomously.

Other works include the human in both object detection and grasp pose identification. Leeper *et al.* [132] proposed a grasping strategy that requires the human to select a general area around the target object, then the robot generates a set of grasping poses for the human to select from or adjust. In [133], the human is requested to approve or decline the detected object. After successful detection, the human reviews the grasp pose options and either approves a suitable pose, or activates an online planner to fine-tune a subset of these options. In these methods, the robot does not reason about its confidence in proposing the grasp pose options. Thus, the human operator is always requested to review the robot suggestions before executing any grasp.

With the increasing interest in learning-based approaches for object grasping and manipulation, researchers have been proposing HRI paradigms to correct manipulation policies through human assistance. In [134], the robot requests human demonstration when it fails to learn how to grasp an object. This approach only depends on the autonomous system’s ability to reason about when to request help. Thus, any failure in detecting the need for human assistance would lead to performance deterioration. In [135], a method was proposed to predict if a manipulation task will fail, and thus request human assistance in correcting the policy, while also allowing the human to intervene and correct the policy in case the robot did not predict its failure. In [136], a human can intervene and correct the robot pose in object manipulation tasks to teach the robot a better policy. Similarly in [137], a human decides when to intervene and correct manipulation

policies by completely taking over robot control. These learning-based methods do not guarantee successful grasping when faced with novel objects. In this POC, the aim is to enhance the performance and success rates of end-to-end robot grasping rather than focusing on a certain sub-task. Human assistance is based on robot requests in tasks when it can reason about its SC, and on human SA when the robot cannot.

5.3 System Overview

This section presents a brief description of the adopted autonomous robot grasping baseline approach, followed by an overview of MISA’s implementation on top of this baseline.

5.3.1 Baseline Approach: Autonomous Grasping

To establish a baseline, the method presented in [124] is adopted for identifying the grasp pose from an RGB image. This method proposes learning a planar robotic grasping model through data collected in real homes using low-cost robots. The dataset used for training consists of $28k$ grasps, where each grasp is performed at the same height and perpendicular to the ground. Each entry of the dataset consists of an image, performed grasp pose, and the grasp success label. This approach combines object detection and grasp pose identification; these two sub-tasks are referred to as the ‘perception module’. The baseline approach can be summarized through the modules shown in Figure 5.2 and discussed below.

Perception Module: given an image I , the goal is to find the best grasp pose (x_s, y_s, θ_s) in the image space, where x_s and y_s are the coordinates of the grasp center location in the image, and θ_s is the roll angle of the end-effector. To predict the grasp pose from I , smaller patches I_k are sampled with their centers being at different positions (x_k, y_k) around the detected object. Since an object can be grasped from different angles, the grasp angle is discretized, for each I_k , into N bins as θ_n . The architecture and modeling details are discussed in [124].

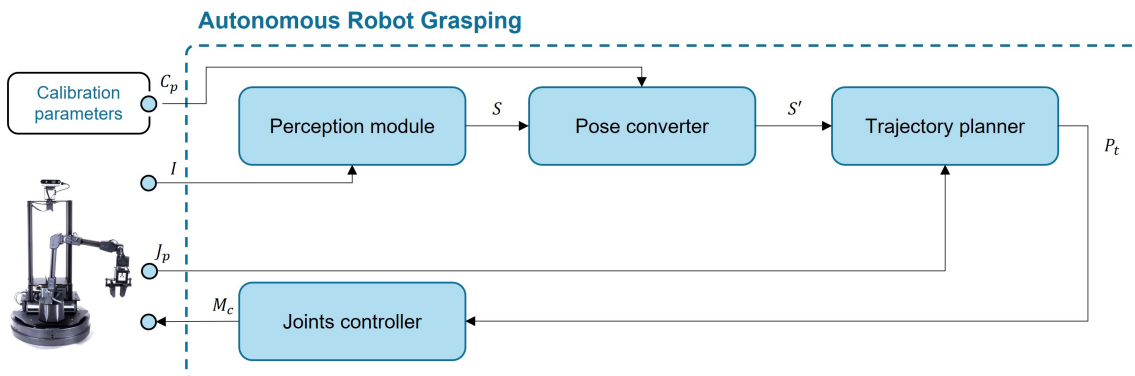


Figure 5.2: Flowchart representing the different modules in Autonomous robot grasping baseline.

The approach uses a patch I_k at angle θ_n to model the grasp success probability as $P(g|I_k, \theta_n, R)$ given by:

$$P(g|I_k, \theta_n, R) = \sum_{\hat{I}_k \in K} P(g|z = \hat{I}_k, \theta_n, R) \cdot P(z = \hat{I}_k|\theta_n, I_k, R), \quad (5.1)$$

where g represents the grasp success label and R represents the noise in the environment. Since a given grasp might be executed at a patch neighbouring the target patch I_k due to inaccuracies in low-cost robots, z models the latent variable of the executed patch, K is the set of possible hypothesis patches neighbouring I_k , and \hat{I}_k is a subset of K . A standard grasp network is used to model $P(g|z = \hat{I}_k, \theta_D, R)$, and a ‘Noise Modelling Network’ is used to model the probability distribution over noise $P(z = \hat{I}_k|\theta_D, I_k, R)$ [124]. The approach considers 9 hypothesis patches around each I_k .

The above method results in a set E of M grasp poses and their corresponding grasp success probabilities for every image I ; the estimated grasps are referred to as $E_m = (x_m, y_m, \theta_m)$. The grasp pose with the highest success probability is proposed for grasp planning and execution as $S = (x_s, y_s, \theta_s)$.

Pose converter: since S is in the image space, the arm-camera calibration parameters (C_p) are used to convert it into a 3D pose S' in the robot space through a conversion library provided in [138].

Trajectory planning Joints controller: for the grasp trajectory planning, MoveIt library [139] is used to generate arm trajectory plan P_t given S' and the arm joints positions J_p . As for the plan execution, ROS low level joint controllers are implemented to generate motor twist commands M_c .

5.3.2 Applying MISA in Robot Grasping

This section describes how the proposed MISA framework is applied in robot grasping, following the generic methodology presented in Chapter 2. Table 5.1 lists the attributes and selections made to apply the MISA framework.

- **Tasks to be supervised by the human:**

Based on the surveyed literature and the full-autonomy experiments presented in Section 5.4.2, the challenges that face the baseline grasping approach are identified below:

- The success rate in identifying a correct grasp pose decreases significantly when using complex, patterned, and spectral backgrounds.
- Although the robot manipulator has an acceptable repeatability, it suffers from calibration errors that yield lower accuracy in executing the grasp plan (e.g., the planned grasp pose does not align with the actual executed pose).
- In certain conditions, the target object is out of reach, but still detectable by the perception system, which leads to failure in generating arm trajectory plans.

Table 5.1: MISA attributes in robotic grasping applications.

Tasks to be supervised		Attributes		
SC-based tasks	SA-based tasks	SC Attributes	SA Attributes	MISA-based UI
<ul style="list-style-type: none"> Object detection and grasp pose identification Arm trajectory planning 	<ul style="list-style-type: none"> Plan/Execution alignment 	<ul style="list-style-type: none"> Successful grasp pose estimation Successful plan generation SC metric: variance in eq. (5.2) SC event: failing to generate a valid plan 	<ul style="list-style-type: none"> Accurate execution of the grasp SA tools: <ul style="list-style-type: none"> live-streaming of robot's camera visualizing the proposed grasp pose 	<ul style="list-style-type: none"> 2D GUI

Based on the above limitations, a human supervisor should be able to (1) evaluate the grasp pose identification and adjust the pose if needed, (2) detect and correct any misalignment between the planned and executed grasps, and (3) adjust the actual object location in case of failure in generating arm trajectory plans.

• **Robot SC attributes:**

Through experimentation, it is noticed that the possibility of identifying wrong grasp poses increases when the positions of the estimated grasps are dispersed. This behavior is captured by calculating the variance of the distances between the estimated grasps E_m positions and the proposed grasp S position. Hence, this variance is used as the SC metric that is required by MISA to allow the agent to ask the human supervisor to approve, decline, or adjust its proposed grasp pose. For the arm trajectory planning, the robot requests human assistance when it fails to generate a valid plan because the object is out of reach. Here, the human can either push the object and reset the task, or cancel the grasp.

Since the criticality of accurate grasp pose identification varies depending on the application and the objects to be grasped, the supervisor can set the ‘no response’ strategy for grasp pose assistance requests between two options: wait for a defined time and proceed with the proposed grasp, or wait until the human responds. As for the grasp planning, the robot pauses and waits until the human responds.

• **Human SA attributes:**

In the baseline approach, the robot does not have the means to reason about calibration errors that lead to mismatching between the planned and executed grasps. Thus, the human supervisor should leverage his/her SA and judgment abilities to assess the accuracy of the grasps execution. The *SA tools* proposed to enhance human SA are enabled by streaming a live feed from the robot's camera through the MISA-based GUI, and showing the proposed grasp when available. These tools help the supervisor detect any execution error, and thus set the calibration offset through the GUI.

- **MISA-based user interface:** To apply the SA-tools and allow for efficient interaction, a user-friendly GUI is developed to display the camera feed, proposed grasp pose, and

the state of the task. In addition, the GUI allows the supervisor to approve or correct the proposed grasp pose when requested, and to set the calibration offset.

Figure 5.3 presents the flowchart of the MISA framework in robot grasping applications. The modules in the **Automated Robot Grasping** block were discussed above in Section 5.3.1, while the modules in **MISAC** block are discussed below.

5.3.2.1 SC-Interaction handler

When the planner is unable to generate a valid plan, this module triggers the SC-event and notifies the supervisor that the planner is not able to generate a valid grasping trajectory. Here, the human can either adjust the actual object and reset the task, or cancel the grasp. As for the proposed grasp pose, the SC-Interaction handler visualized the proposed grasp on the image I through the GUI, and calculates the variance V to check the robot's SC in the proposed grasp pose as follows:

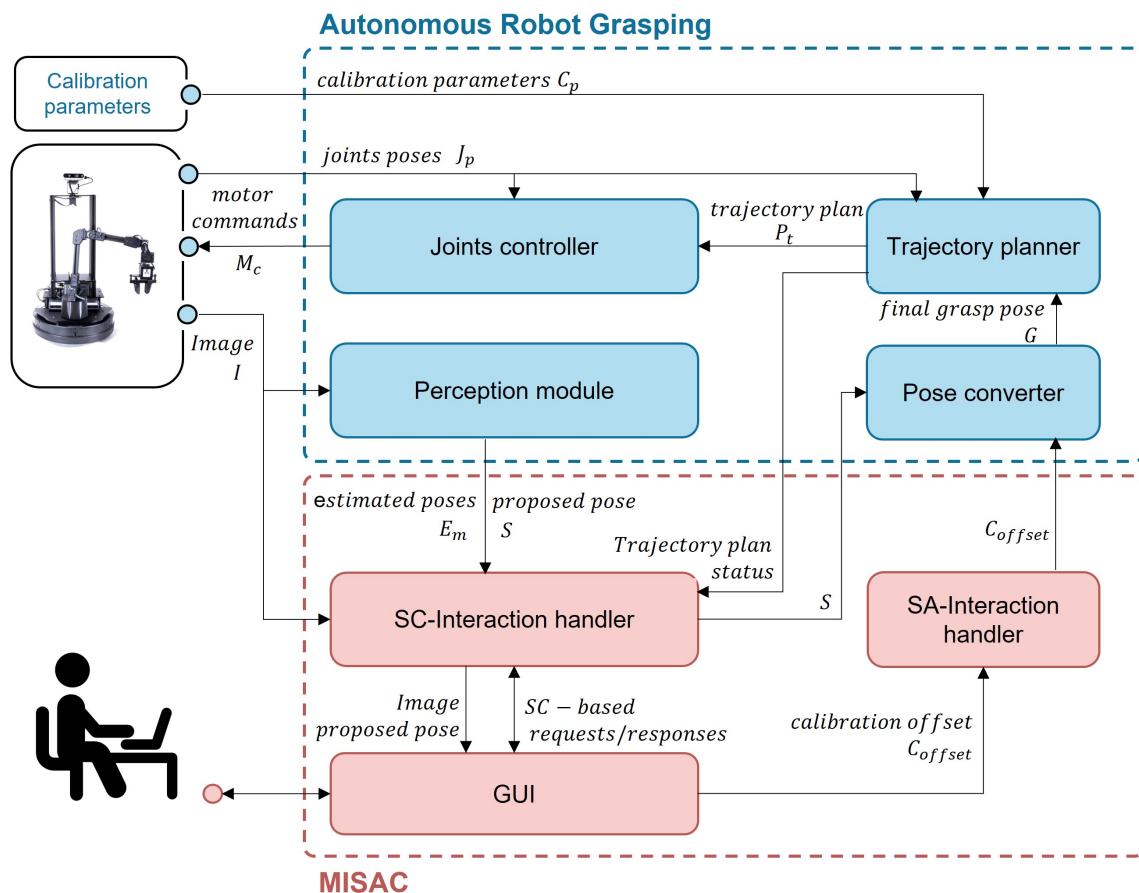


Figure 5.3: Block diagrams of MISA in robot grasping.

$$V = \frac{\sum(d(E_m, S) - \mu)^2}{M}, \quad (5.2)$$

where $d(E_m, S) = \sqrt{(x_m - x_s)^2 + (y_m - y_s)^2}$ is the distance between E_m and S , and μ is the mean of distances.

If $V > V_{th}$, where V_{th} is a confidence threshold obtained through empirical testing, human assistance is requested in approving, declining, or adjusting the proposed pose S via the GUI. If S is adjusted, the updated pose is sent back to the Pose converter module. If the human declines S , the Perception module is commanded to restart the task. If human approves the proposed pose, it is sent back as is to the Pose converter module. In case no human response is received, the SC-Interaction handler applies the no-response strategy set by the supervisor.

5.3.2.2 SA-Interaction handler

This module handles the interaction based on the human's SA. If the supervisor detects a fixed execution offset that is not corrected through the calibration routine, s/he can relay this offset to the Pose converter module by setting the offset parameters $C_{offset} = (x_{offset}, y_{offset}, \theta_{offset})$ in the GUI. Thus, the Pose converter module is adjusted to account for this offset when calculating the final grasp pose $G = S' + C_{offset} = (x, y, \theta)$. As an additional functionality, the SA-Interaction handler allows the supervisor to pause the task and adjust the proposed pose even without any robot request.

5.4 Implementation and Experiments

5.4.1 Experimental Setup

The proposed MISA framework is implemented on top of PyRobot [125], an open-source research platform built on top of ROS. PyRobot provides an implementation of the baseline method, and an arm-camera calibration library using Adam optimizer [140]. In addition, the platform provides interfaces to MoveIt [139], which is used for trajectory planning, and the low-level joints controllers are implemented in ROS. The implementation and experiments were performed on LoCoBot [141], a low-cost robot manipulator that has a Trossen WidowX robotic arm [142] mounted on a Kobuki mobile base [143], which was fixed throughout the entire experimentation procedure. For perception, an Intel Realsense D435 RGBD camera [144] mounted on the robot at a fixed height and orientation is used.

The implementation flowchart is shown in Fig. 5.4. When the system is started, the GUI launches to allow the supervisor to monitor the robot's performance and interact with it. When the perception system detects an object, it generates the estimated grasp poses E_m , and the proposed grasp pose S is visualized on the GUI. If the SC threshold is exceeded ($V > V_{th}$), a notification appears requesting human assistance. Here, the human can approve, decline, or adjust the proposed pose. The system waits for a human response for 20s. If the human does

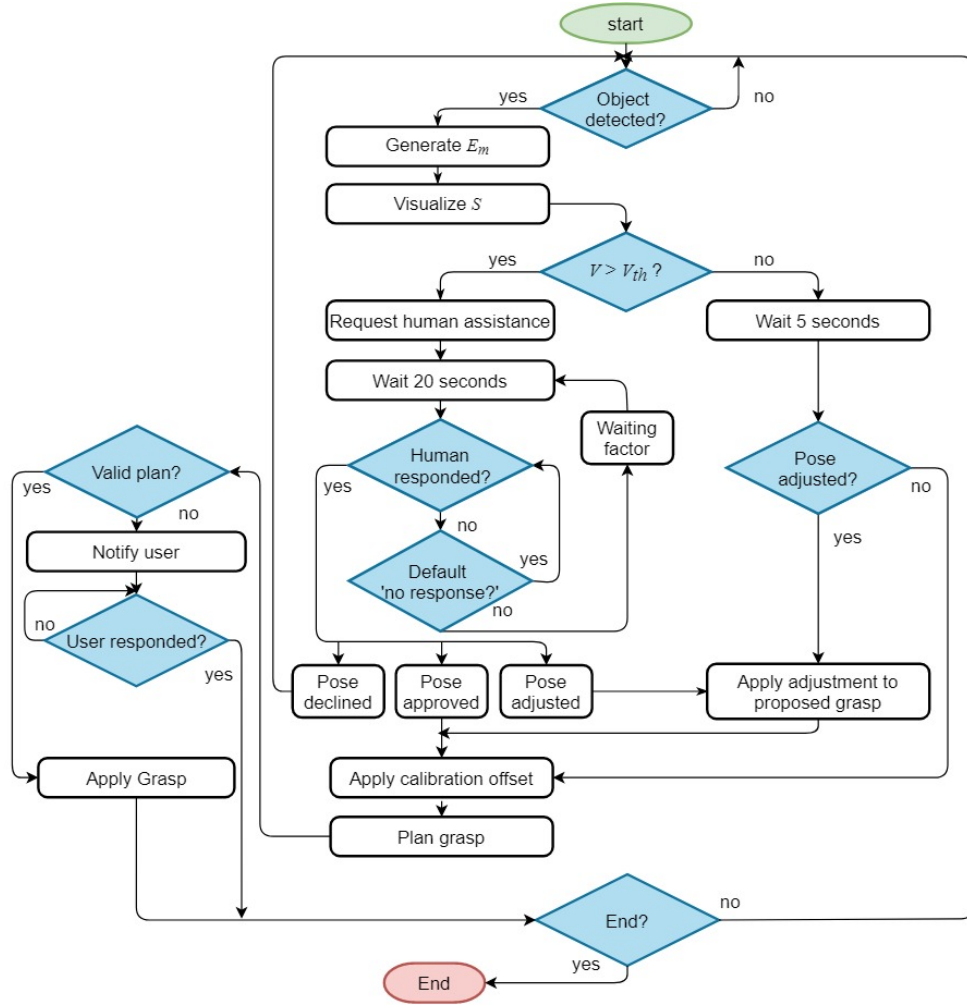


Figure 5.4: Flowchart presenting the implementation logic of MISA in robot grasping.

not respond, the ‘no response’ strategy activates. The default strategy is to wait until response is received; however, the supervisor can change it to a waiting time (factor of 20s) before the system proceeds with the proposed grasp.

If $V < V_{th}$, the system is confident about the proposed grasp pose, and this pose is displayed for 5s before the system proceeds with the grasp. This waiting-time is implemented to allow the supervisor to check the pose, and edit it if s/he finds it necessary based on SA. In case of invalid trajectory plan, a notification appears requesting the human to either adjust the object or cancel the grasp attempt. Moreover, through monitoring the robot, either directly or via the GUI, the human can decide whether the planned grasp pose is being accurately executed or not. If the supervisor detects an execution error, s/he can set the calibration offset, in mm and degrees, through the GUI. The calibration offset is $C = (0mm, 0mm, 06^\circ)$ by default.

Figure 5.5, shows the GUI developed for MISA in robot grasping. The snapshot shows an example of a wrongly proposed pose (rectangle with green and red sides). The GUI displays the proposed grasp pose when available, in addition to the logs, errors, notifications, and warnings. Moreover, the GUI allows the supervisor to adjust the proposed grasp pose upon the robot’s

request, and to set the no-response strategy in addition to the calibration offset. Below is a description of the main components of the GUI:

- **APPROVE**: allows the supervisor to approve the suggested grasp pose,
- **DECLINE**: allows the supervisor to decline the proposed grasp pose,
- **EDIT**: allows the supervisor to edit the proposed grasp pose. When clicked, the editing mode is activated in which the **APPROVE** button is changed to **DONE EDITING**, and the **DECLINE** is changed to **CANCEL EDIT**. When in editing mode (the case shown in Figure 5.5), the supervisor can click on the correct object (dry erase marker) location in the image to set the new grasp pose, shown by the overlaid red line. The orientation of the pose can be adjusted through the **ROTATE LEFT/RIGHT** buttons,
- **ROTATE LEFT**: Shown when in editing mode and allows the supervisor to rotate the newly set grasp pose in a counter-clockwise direction. Each click on the button applies a rotation of 5° .
- **ROTATE RIGHT**: Shown when in editing mode and allows the supervisor to rotate the newly set grasp pose in a clockwise direction. Each click on the button applies a rotation of 5° ,

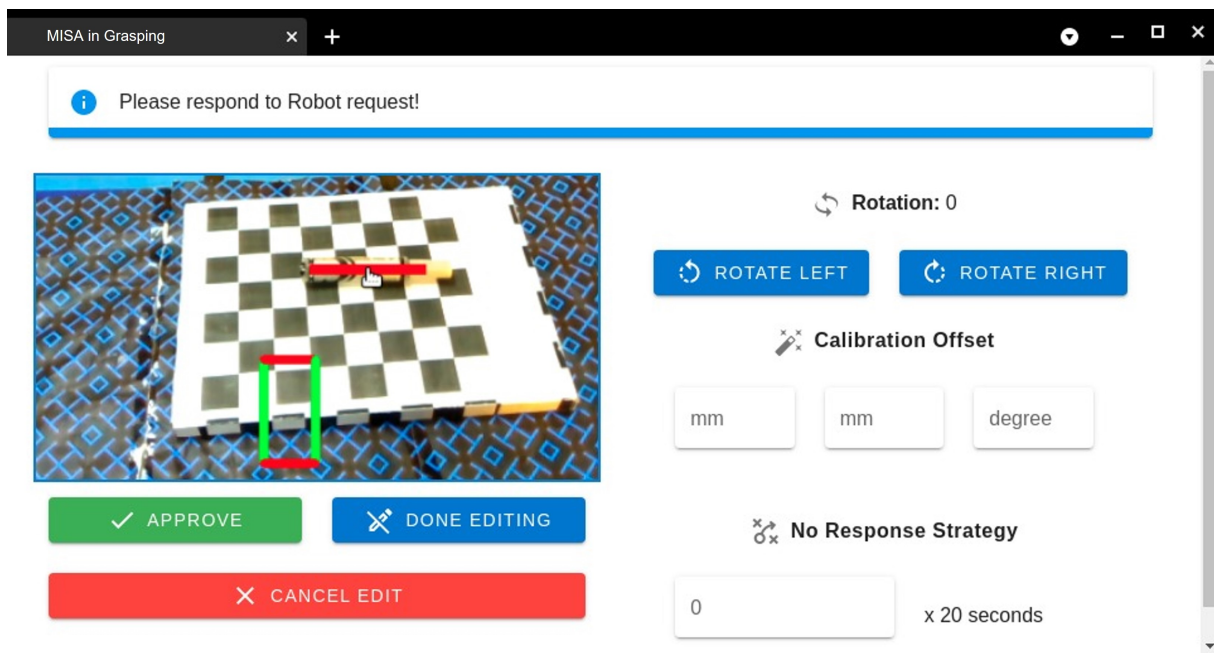


Figure 5.5: The GUI developed for MISA in robot grasping. The snapshot shows an example of a wrongly proposed pose (rectangle with green and red sides), after the supervisor had pressed **EDIT** To adjust the proposed grasp pose, the supervisor clicks on the correct object (dry erase marker) location in the image and edits the pose, shown by the overlaid red line. The orientation of the pose can be adjusted through the Rotate Left/Right buttons, which change the orientation of the pose by 5° increments/decrements.

- **Calibration Offset:** allows the supervisor to set the calibration offset in *mm* and *degree*,
- **No Response Strategy:** allows the supervisor to set the no-response strategy. When the value is kept Zero, the default no-response strategy is applied. However, the supervisor can specify the factor of *20s* to allow the system to wait longer for the response.

5.4.2 Experimental Testing and Results

The first set of experiments (Experiment-1a and Experiment-1b) was conducted to test the performance of the baseline system in full autonomy mode in order to define the MISA framework parameters.

Experiment-1a: Repeatability tests showed an acceptable error ($<1mm$), which aligns with the repeatability error obtained in [125]. To evaluate the perception module, we conducted experiments using 10 different backgrounds (shown in Figure 5.6) and three novel objects. It is important to mention that in [124], the presented results were also based on novel objects in unseen environments. For each object-background pair, we ran 20 trials with the object placed in random orientations and locations. A human evaluated the results to be a success or fail based on SA; for instance, image (1) in Figure 5.6 shows an example of a ‘fail’ proposed pose and image (9) is a ‘success.’ Table 5.2 summarizes all of the obtained results. Examining the results in the Experiment-1 column, we notice that the baseline system achieved acceptable success rates (above 70%) for simple and Lambertian backgrounds (2, 6, and 8); however, the rates decreased drastically with complex, patterned, and spectral backgrounds. The average success rate for all tests was 50.8%.

Through these experiments, it was noticed that the possibility of proposing a wrong grasp pose increases as the dispersion of the estimated grasp poses increases. Figure 5.7 (upper row) shows examples of wrong ‘proposed grasp pose’, where the estimated grasp poses (in blue) are more dispersed than the estimated poses in cases of successful proposed grasp poses (Figure 5.7, lower row). Given that the perception module generates a set of estimated grasp poses with their success probabilities, three parameters were evaluated to select a suitable SC-metric for grasp pose identification: (1) success probabilities, (2) variance of success probabilities, and (3) variance of the distances between the estimated grasp poses and the proposed grasp pose. Using the results of Experiment-1, and for each of the three parameters, the threshold was varied between the parameter’s minimum and maximum values, calculated the FN and FP rates, and constructed the Receiver Operating Characteristic (ROC) curve. The first two parameters had high FP rates, which indicates high probability of predicting success while the grasp pose is wrong; and low area under the ROC curve (AUC) score, which measures how well the parameter can distinguish between successful and failed grasps. The third parameter had a high AUC score (91.9%), and thus it is adopted as the SC-metric. The value of V_{th} was experimentally defined to minimize both FP and FN rates, resulting in $V_{th} = 334$, FP = 16.4%, and FN = 16.6%.

Experiment-1b: the same experimental procedure was performed to evaluate the baseline system’s full autonomy performance in end-to-end grasping, where success is defined as being able to identify a correct grasp pose and successfully grasp the object. Results in the

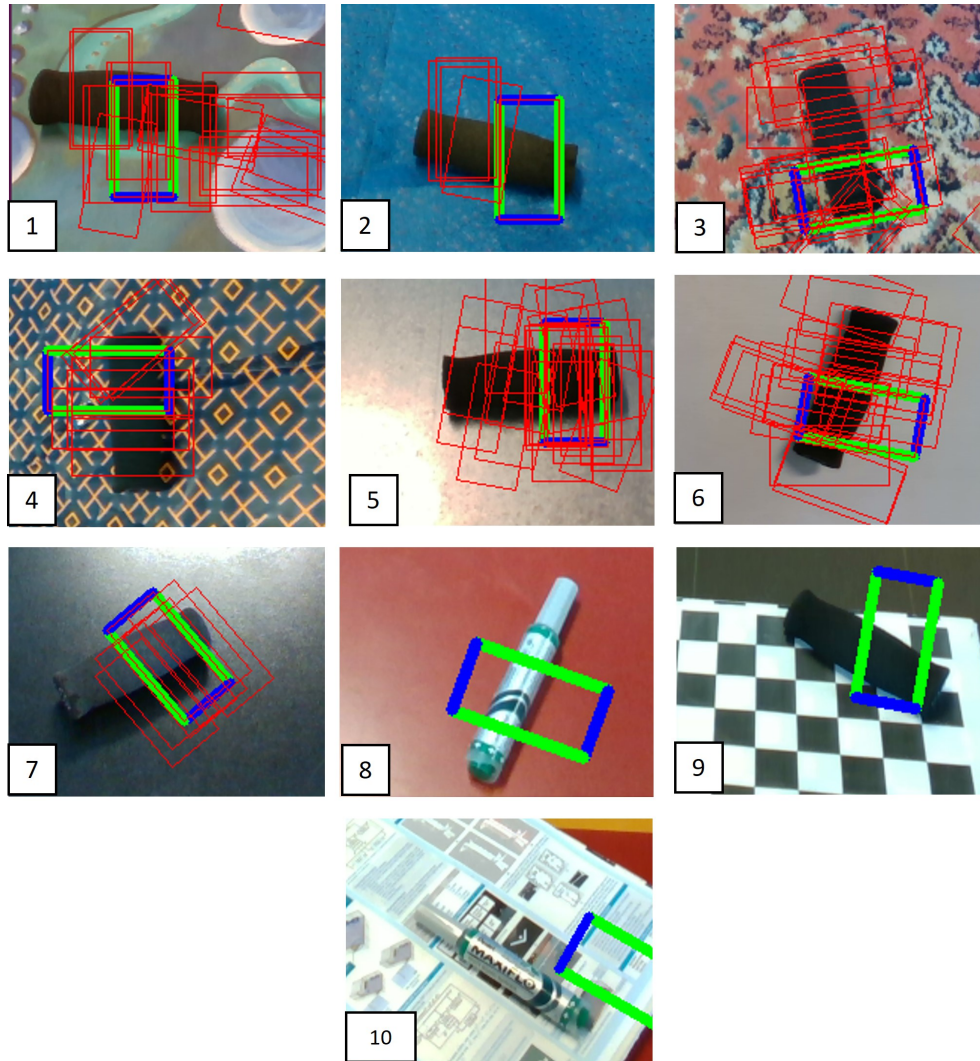


Figure 5.6: The 10 different backgrounds used in the MISA in robot grasping experiments.

Experiment-1b column in Table 5.2 show that even when the system achieves good success rates in grasp pose identification, actual grasping suffers from low accuracy (average of 35%) due to execution errors.

To evaluate the execution error on the LoCobot, we conducted additional tests using five different files of calibration parameters that were obtained using the calibration routine provided in [125]. Each calibration file was generated by using different parameters for the Adam optimizer. For each calibration file, red marks were placed on the robot's work space, and these marks were selected, one by one, as grasp locations through the GUI; the grasp orientation was given a random value in each test. To visualize the execution error, a blue marker was fixed at the center of the LoCobot gripper in a way that it marks the executed grasp location. As shown in Figure 5.8, it was noticed that a fixed execution offset is obtained for each calibration file, irrespective of the location and orientation of the target pose. The minimum attained offset was $C = (20mm, 2mm, 0^\circ)$.

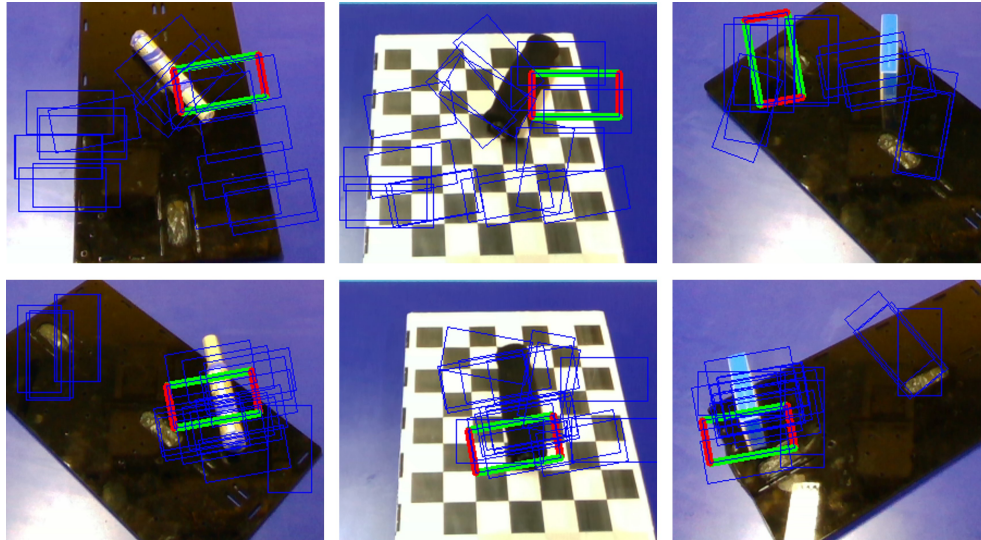


Figure 5.7: Sample results visualizing the dispersion of the estimated grasp poses in cases of: successful grasp pose (lower row), and wrong grasp pose (upper row).

Experiment-2: The second set of experiments was conducted to test the success rate of end-to-end robot grasping through MISA. The tests were conducted with the same conditions as Experiment-1; however, the three objects to grasp were changed. In these tests, the human supervisor was instructed to adjust the proposed grasp pose upon SC requests only. Upon the experiments' start, the supervisor adjusted the calibration offset three times within the first six runs until reaching a specified execution accuracy ($<2mm$, $<2mm$, $<5^\circ$). The calibration routine was performed every 100 runs, and the supervisor was free to adjust the calibration offset based on SA. However, the offset was not changed once set for each calibration routine. As shown in the Experiment-2 column in Table 5.2, the average success rate increased to 87.6% in these tests. This experiment demonstrates that augmenting the baseline system with MISA significantly improves its end-to-end success rate, which increased from 35.0% to 87.6%.

Experiment-3: Additional 10 runs on each object-background pair were conducted. In these tests, the supervisor was allowed to intervene and adjust the proposed grasp pose even without the robot's request, resulting in 100% success rate, as one would expect. However, this comes at the cost of requiring full attention from the human throughout the entire process, which is not practical for all grasping applications.

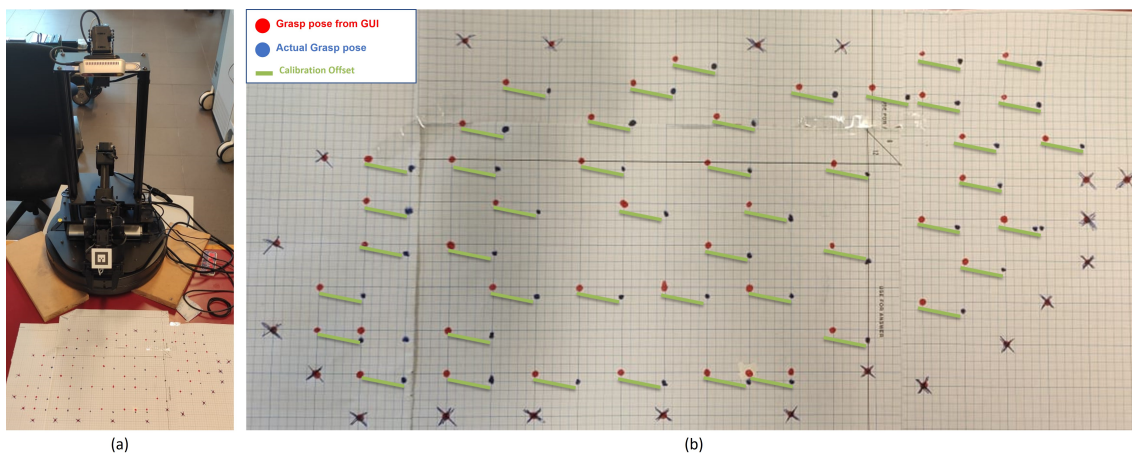


Figure 5.8: A sample result of the execution error experiments. Red dots represents grasp locations that were selected by the supervisor through the GUI, these dots were drawn on the robot’s work space at the beginning of each experiment. Blue dots represent the actual executed grasp location, these dots were obtained through fixing a blue marker at the center of the gripper. Green lines visualize the offset between the target poses and actual poses.

Table 5.2: Experimental Results for MISA in robot grasping.

Background (See Fig. 5.6)	Success Rate		
	Experiment-1a: Baseline (Perception Module Only)	Experiment-1b: Baseline (End-to-End)	Experiment-2: MISA-based System (End-to-End)
1	25.0%	16.7%	83.3%
2	77.7%	44.4%	92.2%
3	41.7%	33.3%	86.7%
4	58.3%	38.3%	86.7%
5	46.7%	32.2%	88.8%
6	70.0%	46.7%	91.7%
7	51.7%	38.3%	83.3%
8	86.7%	66.7%	93.3%
9	35.6%	24.4%	86.7%
10	14.4%	8.9%	83.3%
Average	50.8%	35.0%	87.6%

Chapter 6

Conclusion and future work

The aim of this research is to propose and validate a mixed-initiative supervised autonomy framework for human-robot interaction. Unlike other frameworks that focus on the allocation of functions between humans and robots, MISA is proposed to extend the spectrum of HRI to autonomous systems where it is not feasible for a human to ‘completely takeover’ the automated mission tasks. In the proposed MISA framework, the autonomous agent (robot) performs tasks in an autonomous fashion, but can ask a human supervisor for assistance based on its self-confidence. In addition, the human can intervene and influence the autonomous sensing, planning, and acting based on his/her SA. This framework aims to overcome autonomy challenges and enhance its performance by combining the cognition, flexibility, and problem solving skills of humans with the the strength, endurance, productivity, and precision of robots. As a proof-of-concept, the proposed MISA framework is applied in three different systems: collaborative grid-based SLAM, automated jigsaw puzzle reconstruction, and autonomous robot grasping. In the three POC systems, the following are defined: (1) the challenges and limitations affecting full autonomy performance, (2) the confidence measure that triggers the robot’s request for human assistance, (3) the type and level of intervention that the human can perform. In addition, an augmented reality interface (for SLAM) and 2D GUIs (for puzzle reconstruction and robot grasping) are custom-designed to enhance the human situation awareness and efficiently communicate information and requests.

Through experimental validation, it was shown that applying the MISA framework in fully autonomous systems can help overcome several autonomy limitations and enhance the overall system performance. In fact, MISA outperformed full autonomy in the three implemented systems. In collaborative SLAM, post processing of grid maps was eliminated and the MISA system produced more accurate maps in less number of trials and less run-time for each trial. In automated puzzle reconstruction, results showed that the MISA system outperforms both fully autonomous systems and systems where the human merely intervenes (accepts/rejects) upon the agent’s requests only. Finally, in robot grasping, MISA showed an ability to increase end-to-end success rate of learning-based grasping by more than 50%.

Since this is a first step towards mixed-initiative supervised autonomy, there are some limitations that need to be addressed in any future evolution of MISA. First, the cost of each interaction over its benefit is not evaluated in this work. This is a crucial component that could

be included in the SC/SA attributes to reason about when it is best to request/get assistance through measuring: (1) the cost of interrupting the human supervisor, which is most important when the human is supervising multiple robots; and (2) the cost of waiting for human response, which is important in missions where the robot needs to make rapid decisions and actions. Another limitation is that MISA assumes the human to be available and ready to provide assistance whenever needed, which is not always the case. This assumption could be relaxed by including a module to estimate the human's availability and evaluate his/her capability to provide the needed help. Third, the number of help requests within a mission could be reduced if the robot has a learning-from-interaction capability. Agent learning would lead to extending the types of human assistance to include providing demonstrations, information, and preemptive advice. Finally, to relax the assumption that the human always has superiority over the robot decisions, a human performance evaluation module could be included to reason about whether to accept the human assistance, negotiate it, or refuse it.

Appendix A

Abbreviations

AR	Augmented Reality
AR-HMD	Augmented Reality Head Mounted Device
ARI	Augmented Reality Interface
AUC	Area Under Curve
CAI	Computer Assisted Instruction
DCM	Direct Comparison Metric
DSF	Disjoint Set Forest
FN	False Negative
FP	False Positive
FP GUI	Graphical User Interface
GLCM	Gray Level Co-Occurrence Matrix
HCI	Human-Computer Interaction
LORA	Levels of Robot Autonomy
MI	Mixed Initiative
MI-HRI	Mixed-Initiative Human-Robot Interaction
MISA	Mixed-Initiative Supervised Autonomy
MGC	Mahalanobis Gradient Compatibility
MST	Minimum Spanning Tree
NCM	Neighbour Comparison Metric
OGM	Occupancy Grid Mapping
POC	Proof of Concept
RBPF	Rao-Blackwellized particle filter
ROS	Robot Operating System
ROC	Receiver Operating Characteristic
SC	Self-Confidence
SA	Situation Awareness
SLAM	Simultaneous Localization and Mapping
TBR	Tree Based Reconstruction
URB	University Research Board
VOI	Value of Information

Bibliography

- [1] M. Desai, M. Medvedev, M. Vázquez, S. McSheehy, S. Gadea-Omelchenko, C. Bruggeman, A. Steinfeld, and H. Yanco, “Effects of changing reliability on trust of robot systems,” in *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 73–80, IEEE, 2012.
- [2] S. Rosenthal, J. Biswas, and M. M. Veloso, “An effective personal mobile robot agent through symbiotic human-robot interaction.,” in *AAMAS*, vol. 10, pp. 915–922, 2010.
- [3] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, “A model for types and levels of human interaction with automation,” *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.
- [4] M. Lippi and A. Marino, “Human multi-robot physical interaction: a distributed framework,” *Journal of Intelligent & Robotic Systems*, vol. 101, no. 2, pp. 1–20, 2021.
- [5] P. Glogowski, A. Böhmer, H. Alfred, and K. Bernd, “Robot speed adaption in multiple trajectory planning and integration in a simulation tool for human-robot interaction,” *Journal of Intelligent & Robotic Systems*, vol. 102, no. 1, 2021.
- [6] S. Li, H. Wang, and S. Zhang, “Human-robot collaborative manipulation with the suppression of human-caused disturbance,” *Journal of Intelligent & Robotic Systems*, vol. 102, no. 4, pp. 1–11, 2021.
- [7] J. M. Lockhart, M. H. Strub, J. K. Hawley, and L. A. Tapia, “Automation and supervisory control: A perspective on human performance, training, and performance aiding,” in *Proceedings of the Human Factors and Ergonomics Society annual meeting*, vol. 37, pp. 1211–1215, SAGE Publications Sage CA: Los Angeles, CA, 1993.
- [8] A. B. Moniz and B.-J. Krings, “Robots working with humans or humans working with robots? searching for social dimensions in new human-robot interaction in industry,” *Societies*, vol. 6, no. 3, p. 23, 2016.
- [9] X. Liu, S. S. Ge, F. Zhao, and X. Mei, “A dynamic behavior control framework for physical human-robot interaction,” *Journal of Intelligent & Robotic Systems*, vol. 101, no. 1, pp. 1–18, 2021.

- [10] T. Kaupp, A. Makarenko, and H. Durrant-Whyte, “Human–robot communication for collaborative decision making—a probabilistic approach,” *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 444–456, 2010.
- [11] M. R. Endsley, “From here to autonomy: lessons learned from human–automation research,” *Human factors*, vol. 59, no. 1, pp. 5–27, 2017.
- [12] P. K. Pook and D. H. Ballard, “Deictic human/robot interaction,” *Robotics and Autonomous Systems*, vol. 18, no. 1-2, pp. 259–269, 1996.
- [13] J. Fritsch, M. Kleinhagenbrock, S. Lang, T. Plötz, G. A. Fink, and G. Sagerer, “Multi-modal anchoring for human–robot interaction,” *Robotics and Autonomous Systems*, vol. 43, no. 2-3, pp. 133–147, 2003.
- [14] K. Severinson-Eklundh, A. Green, and H. Hüttenrauch, “Social and collaborative aspects of interaction with a service robot,” *Robotics and Autonomous systems*, vol. 42, no. 3-4, pp. 223–234, 2003.
- [15] J. van der Rijt, P. Van den Bossche, M. W. van de Wiel, S. De Maeyer, W. H. Gijsselaers, and M. S. Segers, “Asking for help: A relational perspective on help seeking in the workplace,” *Vocations and learning*, vol. 6, no. 2, pp. 259–279, 2013.
- [16] D. E. Cantor and Y. Jin, “Theoretical and empirical evidence of behavioral and production line factors that influence helping behavior,” *Journal of Operations Management*, vol. 65, no. 4, pp. 312–332, 2019.
- [17] M. L. Frazier and C. Tupper, “Supervisor prosocial motivation, employee thriving, and helping behavior: A trickle-down model of psychological safety,” *Group & Organization Management*, vol. 43, no. 4, pp. 561–593, 2018.
- [18] G. S. Van der Vegt and E. Van de Vliert, “Effects of perceived skill dissimilarity and task interdependence on helping in work teams,” *Journal of management*, vol. 31, no. 1, pp. 73–89, 2005.
- [19] V. Srinivasan and L. Takayama, “Help me please: Robot politeness strategies for soliciting help from humans,” in *Proceedings of the 2016 CHI conference on human factors in computing systems*, pp. 4945–4955, 2016.
- [20] J. R. Carbonell, “Ai in cai: An artificial-intelligence approach to computer-assisted instruction,” *IEEE transactions on man-machine systems*, vol. 11, no. 4, pp. 190–202, 1970.
- [21] J. Allen, C. I. Guinn, and E. Horvitz, “Mixed-initiative interaction,” *IEEE Intelligent Systems and their Applications*, vol. 14, no. 5, pp. 14–23, 1999.
- [22] E. Horvitz, “Principles of mixed-initiative user interfaces,” in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 159–166, 1999.

- [23] R. Cohen, C. Allaby, C. Cumbaa, M. Fitzgerald, K. Ho, B. Hui, C. Latulipe, F. Lu, N. Moussa, D. Pooley, *et al.*, “What is initiative?,” *User Modeling and User-Adapted Interaction*, vol. 8, no. 3-4, pp. 171–214, 1998.
- [24] S. Jiang and R. C. Arkin, “Mixed-initiative human-robot interaction: definition, taxonomy, and survey,” in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 954–961, IEEE, 2015.
- [25] S. Zieba, P. Polet, F. Vanderhaegen, and S. Debernard, “Principles of adjustable autonomy: a framework for resilient human–machine cooperation,” *Cognition, Technology & Work*, vol. 12, no. 3, pp. 193–203, 2010.
- [26] S. A. Mostafa, M. S. Ahmad, and A. Mustapha, “Adjustable autonomy: a systematic literature review,” *Artificial Intelligence Review*, vol. 51, no. 2, pp. 149–186, 2019.
- [27] T. B. Sheridan and W. L. Verplank, “Human and computer control of undersea teleoperators,” *Human and computer control of undersea teleoperators*, 1978.
- [28] S. Thrun, “Toward a framework for human-robot interaction,” *Human–Computer Interaction*, vol. 19, no. 1-2, pp. 9–24, 2004.
- [29] J. M. Beer, A. D. Fisk, and W. A. Rogers, “Toward a framework for levels of robot autonomy in human-robot interaction,” *Journal of human-robot interaction*, vol. 3, no. 2, p. 74, 2014.
- [30] F. Tang and E. Ito, “Human-assisted navigation through sliding autonomy,” in *2017 2nd International Conference on Robotics and Automation Engineering (ICRAE)*, pp. 26–30, IEEE, 2017.
- [31] M. A. Goodrich, D. R. Olsen, J. W. Crandall, and T. J. Palmer, “Experiments in adjustable autonomy,” in *Proceedings of IJCAI Workshop on autonomy, delegation and control: interacting with intelligent agents*, pp. 1624–1629, Seattle, WA, 2001.
- [32] G. Gemignani, R. Capobianco, E. Bastianelli, D. D. Bloisi, L. Iocchi, and D. Nardi, “Living with robots: Interactive environmental knowledge acquisition,” *Robotics and Autonomous Systems*, vol. 78, pp. 1–16, 2016.
- [33] M. Y. Cheng and R. Cohen, “A hybrid transfer of control model for adjustable autonomy multiagent systems,” in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pp. 1149–1150, 2005.
- [34] S. Rosenthal, M. Veloso, and A. K. Dey, “Is someone in this office available to help me?,” *Journal of Intelligent & Robotic Systems*, vol. 66, no. 1, pp. 205–221, 2012.
- [35] T. Fong, C. Thorpe, and C. Baur, “Robot, asker of questions,” *Robotics and Autonomous systems*, vol. 42, no. 3-4, pp. 235–243, 2003.

- [36] T. M. Roehr and Y. Shi, “Using a self-confidence measure for a system-initiated switch between autonomy modes,” in *Proceedings of the 10th international symposium on artificial intelligence, robotics and automation in space, Sapporo, Japan*, pp. 507–514, 2010.
- [37] L. Burks, N. Ahmed, I. Lofgren, L. Barbier, J. Muesing, J. McGinley, and S. Vunnam, “Collaborative human-autonomy semantic sensing through structured pomdp planning,” *Robotics and Autonomous Systems*, p. 103753, 2021.
- [38] M. Chiou, N. Hawes, and R. Stolkin, “Mixed-initiative variable autonomy for remotely operated mobile robots,” *arXiv preprint arXiv:1911.04848*, 2019.
- [39] M. Guo, S. Andersson, and D. V. Dimarogonas, “Human-in-the-loop mixed-initiative control under temporal tasks,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6395–6400, IEEE, 2018.
- [40] E. A. M. Ghalamzan, F. Abi-Farraj, P. R. Giordano, and R. Stolkin, “Human-in-the-loop optimisation: mixed initiative grasping for optimally facilitating post-grasp manipulative actions,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3386–3393, IEEE, 2017.
- [41] R. Chipalkatty, G. Droge, and M. B. Egerstedt, “Less is more: Mixed-initiative model-predictive control with human inputs,” *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 695–703, 2013.
- [42] R. R. Murphy, J. Casper, M. Micire, J. Hyams, *et al.*, “Mixed-initiative control of multiple heterogeneous robots for urban search and rescue,” *proceedings of the IEEE Transactions on Robotics and Automation*, 2000.
- [43] J. Engel, J. Sturm, and D. Cremers, “Accurate figure flying with a quadcopter using onboard visual and inertial sensing,” *Imu*, vol. 320, no. 240, 2012.
- [44] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, “Autonomous uav surveillance in complex urban environments,” in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, pp. 82–85, IEEE, 2009.
- [45] N. Fairfield, G. Kantor, and D. Wettergreen, “Real-time slam with octree evidence grids for exploration in underwater tunnels,” *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 03–21, 2007.
- [46] S. B. Williams, O. R. Pizarro, M. V. Jakuba, C. R. Johnson, N. S. Barrett, R. C. Babcock, G. A. Kendrick, P. D. Steinberg, A. J. Heyward, P. J. Doherty, *et al.*, “Monitoring of benthic reference sites: using an autonomous underwater vehicle,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 73–84, 2012.
- [47] C. Beall, B. J. Lawrence, V. Ila, and F. Dellaert, “3d reconstruction of underwater structures,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4418–4423, IEEE, 2010.

- [48] A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg, and S. Thrun, “6d slam with an application in autonomous mine mapping,” in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA’04. 2004*, vol. 2, pp. 1998–2003, IEEE, 2004.
- [49] S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Hahnel, D. Montemerlo, A. Morris, Z. Omohundro, *et al.*, “Autonomous exploration and mapping of abandoned mines,” *IEEE Robotics & Automation Magazine*, vol. 11, no. 4, pp. 79–91, 2004.
- [50] P. Schmuck and M. Chli, “Multi-uav collaborative monocular slam,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3863–3870, IEEE, 2017.
- [51] R. Dubé, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, “An online multi-robot slam system for 3d lidars,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1004–1011, IEEE, 2017.
- [52] P. Fankhauser, M. Bloesch, P. Krüsi, R. Diethelm, M. Wermelinger, T. Schneider, M. Dymczyk, M. Hutter, and R. Siegwart, “Collaborative navigation for flying and walking robots,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2859–2866, IEEE, 2016.
- [53] C. P. Quintero, S. Li, M. K. Pan, W. P. Chan, H. M. Van der Loos, and E. Croft, “Robot programming through augmented trajectories in augmented reality,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1838–1844, IEEE, 2018.
- [54] D. Krupke, F. Steinicke, P. Lubos, Y. Jonetzko, M. Gerner, and J. Zhang, “Comparison of multimodal heading and pointing gestures for co-located mixed reality human-robot interaction,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, IEEE, 2018.
- [55] M. Pollefeys, “Microsoft hololens,” Jun 2018.
- [56] E. A. Topp and H. I. Christensen, “Tracking for following and passing persons,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2321–2327, IEEE, 2005.
- [57] A. Diosi, G. Taylor, and L. Kleeman, “Interactive slam using laser and advanced sonar,” in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 1103–1108, IEEE, 2005.
- [58] P. Vieira and R. Ventura, “Interactive mapping in 3d using rgb-d data,” in *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 1–6, IEEE, 2012.
- [59] D. Sprute, K. Tönnies, and M. König, “Virtual borders: Accurate definition of a mobile robot’s workspace using augmented reality,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8574–8581, IEEE, 2018.

- [60] H. Surmann, N. Berninger, and R. Worst, “3d mapping for multi hybrid robot cooperation,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 626–633, IEEE, 2017.
- [61] H. Liu, Y. Zhang, W. Si, X. Xie, Y. Zhu, and S.-C. Zhu, “Interactive robot knowledge patching using augmented reality,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1947–1954, IEEE, 2018.
- [62] H. Fang, S.-K. Ong, and A. Y. Nee, “Novel ar-based interface for human-robot interaction and visualization,” *Advances in Manufacturing*, vol. 2, no. 4, pp. 275–288, 2014.
- [63] M. Zolotas, J. Elsdon, and Y. Demiris, “Head-mounted augmented reality for explainable robotic wheelchair assistance,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1823–1829, IEEE, 2018.
- [64] C. Reardon, K. Lee, and J. Fink, “Come see this! augmented reality to enable human-robot cooperative search,” in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 1–7, IEEE, 2018.
- [65] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with rao-blackwellized particle filters,” *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.
- [66] K. P. Murphy, “Bayesian map learning in dynamic environments,” in *Advances in Neural Information Processing Systems*, pp. 1015–1021, 2000.
- [67] F. Lu and E. Milius, “Globally consistent range scan alignment for environment mapping,” *Autonomous robots*, vol. 4, no. 4, pp. 333–349, 1997.
- [68] H. Moravec and A. Elfes, “High resolution maps from wide angle sonar,” in *Proceedings. 1985 IEEE international conference on robotics and automation*, vol. 2, pp. 116–121, IEEE, 1985.
- [69] M. Montemerlo, N. Roy, and S. Thrun, “Perspectives on standardization in mobile robot programming: The carnegie mellon navigation (carmen) toolkit,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3, pp. 2436–2441, IEEE, 2003.
- [70] “Gmapping ros package.”
- [71] “Move_base ros package.”
- [72] “Rosbridge suite ros package.”
- [73] “Holotoolkit sharing library.”
- [74] “Rviz interface.”
- [75] “vuforia.”

- [76] C. Zanoci and J. Andress, “Making puzzles less puzzling: An automatic jigsaw puzzle solver,” 2016.
- [77] T. Altman, “Solving the jigsaw puzzle problem in linear time,” *Applied Artificial Intelligence an International Journal*, vol. 3, no. 4, pp. 453–462, 1989.
- [78] E. D. Demaine and M. L. Demaine, “Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity,” *Graphs and Combinatorics*, vol. 23, no. 1, pp. 195–208, 2007.
- [79] W. Marande and G. Burger, “Mitochondrial dna as a genomic jigsaw puzzle,” *Science*, vol. 318, no. 5849, pp. 415–415, 2007.
- [80] Y.-X. Zhao, M.-C. Su, Z.-L. Chou, and J. Lee, “A puzzle solver and its application in speech descrambling,” in *WSEAS Int. Conf. Computer Engineering and Applications*, pp. 171–176, Citeseer, 2007.
- [81] B. J. Brown, C. Toler-Franklin, D. Nehab, M. Burns, D. Dobkin, A. Vlachopoulos, C. Doumas, S. Rusinkiewicz, and T. Weyrich, “A system for high-volume acquisition and matching of fresco fragments: Reassembling theran wall paintings,” *ACM transactions on graphics (TOG)*, vol. 27, no. 3, pp. 1–9, 2008.
- [82] K. Hori, M. Imai, and T. Ogasawara, “Joint detection for potsherds of broken earthenware,” in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, vol. 2, pp. 440–445, IEEE, 1999.
- [83] H. Liu, S. Cao, and S. Yan, “Automated assembly of shredded pieces from multiple photos,” *IEEE transactions on multimedia*, vol. 13, no. 5, pp. 1154–1162, 2011.
- [84] E. Justino, L. S. Oliveira, and C. Freitas, “Reconstructing shredded documents through feature matching,” *Forensic science international*, vol. 160, no. 2-3, pp. 140–147, 2006.
- [85] M. A. Marques and C. O. Freitas, “Reconstructing strip-shredded documents using color as feature matching,” in *Proceedings of the 2009 ACM symposium on Applied Computing*, pp. 893–894, 2009.
- [86] L. Zhu, Z. Zhou, and D. Hu, “Globally consistent reconstruction of ripped-up documents,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 1, pp. 1–13, 2007.
- [87] H.-C. Shih and C.-L. Lu, “Divide-and-conquer jigsaw puzzle solving,” in *2018 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–2, IEEE, 2018.
- [88] T. S. Cho, S. Avidan, and W. T. Freeman, “A probabilistic image jigsaw puzzle solver,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 183–190, IEEE, 2010.
- [89] R. Yu, C. Russell, and L. Agapito, “Solving jigsaw puzzles with linear programming,” *arXiv preprint arXiv:1511.04472*, 2015.

- [90] A. C. Gallagher, “Jigsaw puzzles with pieces of unknown orientation,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 382–389, IEEE, 2012.
- [91] X. Zheng, X. Lu, and Y. Yuan, “Image jigsaw puzzles with a self-correcting solver,” in *2013 International Conference on Virtual Reality and Visualization*, pp. 112–118, IEEE, 2013.
- [92] D. Palmer, M. Kirschenbaum, E. Mustee, and J. Dengler, “Human-swarm hybrids outperform both humans and swarms solving digital jigsaw puzzles,” in *2014 IEEE Symposium on Swarm Intelligence*, pp. 1–8, IEEE, 2014.
- [93] H. Freeman and L. Garder, “Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition,” *IEEE Transactions on Electronic Computers*, no. 2, pp. 118–127, 1964.
- [94] A. Zisserman, D. A. Forsyth, J. L. Mundy, and C. A. Rothwell, “Recognizing general curved objects efficiently,” in *Geometric Invariance in Computer Vision*, pp. 228–251, 1992.
- [95] D. Goldberg, C. Malon, and M. Bern, “A global approach to automatic solution of jigsaw puzzles,” in *Proceedings of the eighteenth annual symposium on Computational geometry*, pp. 82–87, 2002.
- [96] H. Wolfson, E. Schonberg, A. Kalvin, and Y. Lamdan, “Solving jigsaw puzzles by computer,” *Annals of Operations Research*, vol. 12, no. 1, pp. 51–64, 1988.
- [97] D. A. Kosiba, P. M. Devaux, S. Balasubramanian, T. L. Gandhi, and K. Kasturi, “An automatic jigsaw puzzle solver,” in *Proceedings of 12th International Conference on Pattern Recognition*, vol. 1, pp. 616–618, IEEE, 1994.
- [98] Z. Ling-ling, G. Qing-ping, and J. Zhao-zhong, “Image matching algorithm based on edge color used in computer jigsaw puzzle [j],” *Computer Engineering and Design*, vol. 16, 2010.
- [99] R. W. Webster, P. S. LaFollette, and R. L. Stafford, “Isthmus critical points for solving jigsaw puzzles in computer vision,” *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 5, pp. 1271–1278, 1991.
- [100] T. R. Nielsen, P. Drewsen, and K. Hansen, “Solving jigsaw puzzles using image features,” *Pattern Recognition Letters*, vol. 29, no. 14, pp. 1924–1933, 2008.
- [101] F.-H. Yao and G.-F. Shao, “A shape and image merging technique to solve jigsaw puzzles,” *Pattern Recognition Letters*, vol. 24, no. 12, pp. 1819–1835, 2003.
- [102] E. D. Demaine and M. L. Demaine, “Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity,” *Graphs and Combinatorics*, vol. 23, no. 1, pp. 195–208, 2007.

- [103] H.-C. Shih, J.-L. Lu, and C.-H. Ma, “Solving jigsaw puzzles via hausdorff-based border compatibility,” in *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pp. 504–505, IEEE, 2019.
- [104] H.-C. Shih, J.-L. Lu, and C.-H. Ma, “Square puzzle solving using border compatibility matching,” in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–5, IEEE, 2019.
- [105] S.-Y. Jin, S. Lee, N. A. Azis, and H.-J. Choi, “Jigsaw puzzle image retrieval via pairwise compatibility measurement,” in *2014 International Conference on Big Data and Smart Computing (BIGCOMP)*, pp. 123–127, IEEE, 2014.
- [106] H. Li, Y. Zheng, S. Zhang, and J. Cheng, “Solving a special type of jigsaw puzzles: Banknote reconstruction from a large number of fragments,” *IEEE transactions on multimedia*, vol. 16, no. 2, pp. 571–578, 2013.
- [107] H. Zhang, J. K. Lai, and M. Bächer, “Hallucination: A mixed-initiative approach for efficient document reconstruction,” in *Workshops at the twenty-sixth AAAI conference on artificial intelligence*, 2012.
- [108] A. Deever and A. Gallagher, “Semi-automatic assembly of real cross-cut shredded documents,” in *2012 19th IEEE International Conference on Image Processing*, pp. 233–236, IEEE, 2012.
- [109] S. Shang, H. T. Sencar, N. Memon, and X. Kong, “A semi-automatic deshredding method based on curve matching,” in *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 5537–5541, IEEE, 2014.
- [110] K. Son, J. Hays, and D. B. Cooper, “Solving square jigsaw puzzles with loop constraints,” in *European Conference on Computer Vision*, pp. 32–46, Springer, 2014.
- [111] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, *et al.*, “Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching,” in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3750–3757, IEEE, 2018.
- [112] J. Shi and G. S. Koonjul, “Real-time grasping planning for robotic bin-picking and kitting applications,” *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 809–819, 2017.
- [113] J. Zhang, W. Zhang, R. Song, L. Ma, and Y. Li, “Grasp for stacking via deep reinforcement learning,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2543–2549, IEEE, 2020.
- [114] J. Zhao, D. Troniak, and O. Kroemer, “Towards robotic assembly by predicting robust, precise and task-oriented grasps,” *arXiv preprint arXiv:2011.02462*, 2020.
- [115] M. Graña, M. Alonso, and A. Izaguirre, “A panoramic survey on grasping research trends and topics,” *Cybernetics and Systems*, vol. 50, no. 1, pp. 40–57, 2019.

- [116] M. Vohra, R. Prakash, and L. Behera, “Real-time grasp pose estimation for novel objects in densely cluttered environment,” in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1–6, IEEE, 2019.
- [117] D. Prattichizzo, J. C. Trinkle, B. Siciliano, and O. Khatib, “Springer handbook of robotics,” *Grasping; Springer: Berlin/Heidelberg, Germany*, pp. 671–700, 2008.
- [118] J. Mahler, S. Patil, B. Kehoe, J. Van Den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg, “Gp-gpis-opt: Grasp planning with shape uncertainty using gaussian process implicit surfaces and sequential convex programming,” in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 4919–4926, IEEE, 2015.
- [119] Q. M. Marwan, S. C. Chua, and L. C. Kwek, “Comprehensive review on reaching and grasping of objects in robotics,” *Robotica*, pp. 1–34, 2021.
- [120] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, “6-dof grasping for target-driven object manipulation in clutter,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6232–6238, IEEE, 2020.
- [121] Y. Lin, H. Min, and H. Wei, “Inertial measurement unit-based iterative pose compensation algorithm for low-cost modular manipulator,” *Advances in Mechanical Engineering*, vol. 8, no. 1, p. 1687814015626850, 2016.
- [122] M. Quigley, A. Asbeck, and A. Ng, “A low-cost compliant 7-dof robotic manipulator,” in *2011 IEEE International Conference on Robotics and Automation*, pp. 6051–6058, IEEE, 2011.
- [123] D. V. Gealy, S. McKinley, B. Yi, P. Wu, P. R. Downey, G. Balke, A. Zhao, M. Guo, R. Thomasson, A. Sinclair, *et al.*, “Quasi-direct drive for low-cost compliant robotic manipulation,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 437–443, IEEE, 2019.
- [124] A. Gupta, A. Murali, D. Gandhi, and L. Pinto, “Robot learning in homes: Improving generalization and reducing dataset bias,” *arXiv preprint arXiv:1807.07049*, 2018.
- [125] A. Murali, T. Chen, K. V. Alwala, D. Gandhi, L. Pinto, S. Gupta, and A. Gupta, “Pyrobot: An open-source robotics framework for research and benchmarking,” *arXiv preprint arXiv:1906.08236*, 2019.
- [126] D. J. Butler, S. Elliot, and M. Cakmak, “Interactive scene segmentation for efficient human-in-the-loop robot manipulation,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2572–2579, IEEE, 2017.
- [127] B. Pitzer, M. Styer, C. Bersch, C. DuHadway, and J. Becker, “Towards perceptual shared autonomy for robotic mobile manipulation,” in *2011 IEEE International Conference on Robotics and Automation*, pp. 6245–6251, IEEE, 2011.
- [128] S. Masnadi, J. J. LaViola Jr, J. Pavlasek, X. Zhu, K. Desingh, and O. C. Jenkins, “Sketching affordances for human-in-the-loop robotic manipulation tasks,” *2nd Robot Team-mates Operating in Dynamic, Unstructured Environments (RT-DUNE)*, 2019.

- [129] S. Rosa, A. Russo, A. Saglinbeni, and G. Toscana, “Vocal interaction with a 7-dof robotic arm for object detection, learning and grasping,” in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 505–506, IEEE, 2016.
- [130] J. Hough and D. Schlangen, “It’s not what you do, it’s how you do it: Grounding uncertainty for a simple robot,” in *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 274–282, IEEE, 2017.
- [131] S. Valipour, C. Perez, and M. Jagersand, “Incremental learning for robot perception through hri,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2772–2777, IEEE, 2017.
- [132] A. E. Leeper, K. Hsiao, M. Ciocarlie, L. Takayama, and D. Gossow, “Strategies for human-in-the-loop robotic grasping,” in *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pp. 1–8, 2012.
- [133] J. Weisz, P. K. Allen, A. G. Barszap, and S. S. Joshi, “Assistive grasping with an augmented reality user interface,” *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 543–562, 2017.
- [134] J. DelPreto, J. I. Lipton, L. Sanneman, A. J. Fay, C. Fourie, C. Choi, and D. Rus, “Helping robots learn: a human-robot master-apprentice model using demonstrations via virtual reality teleoperation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10226–10233, IEEE, 2020.
- [135] T. Ablett, F. Marić, and J. Kelly, “Fighting failures with fire: Failure identification to reduce expert burden in intervention-based learning,” *arXiv preprint arXiv:2007.00245*, 2020.
- [136] A. Bajcsy, D. P. Losey, M. K. O’Malley, and A. D. Dragan, “Learning from physical human corrections, one feature at a time,” in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 141–149, 2018.
- [137] A. Mandlekar, D. Xu, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese, “Human-in-the-loop imitation learning using remote teleoperation,” *arXiv preprint arXiv:2012.06733*, 2020.
- [138] “grasp library.”
- [139] S. Chitta, I. Sukan, and S. Cousins, “Moveit![ros topics],” *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.
- [140] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [141] “Locobot, an open source low cost robot.”
- [142] “Widowx 200 robot arm.”
- [143] “Kobuki mobile base.”

[144] “Intel realsense d435 rgb camera.”