Institute of Mechanism Theory, Machine Dynamics and Robotics

RWTH AACHEN UNIVERSITY

This thesis was submitted to the Institute of Mechanism Theory, Machine Dynamics and Robotics

Enhancing Manipulation Performance with Grounded Haptic UI Teleoperation

Master Thesis

by:

Ajithkumar Narayanan Manaparampil B.Sc.

Student number: 415918

supervised by:

Dr. Neal Y Lii

Elodie Huesing, M.Sc. RWTH

Examiner:

Univ.-Prof. Dr.-Ing. Dr. h. c. Burkhard Corves

Prof. Dr.-Ing. Mathias Hüsing

Aachen, 23 August 2023

Master Thesis

by Ajithkumar Narayanan Manaparampil B.Sc.
Student number: 415918

Enhancing Manipulation Performance with Grounded Haptic UI Teleoperation

---

Teleoperation refers to the process of remotely controlling a robot by a human operator. The objective of teleoperation is to complete tasks in inaccessible or hazardous environments while under human supervision. Within teleoperation, manipulating objects in the environment can be particularly challenging because the human operator has limited feedback compared to the robot. Despite the robot having access to depth information, the human operator may only have access to 2D visuals or a limited representation of 3D data, depending on network bandwidth and time lag. One potential solution is to enable the robot to guide the human operator as the robot has access to a higher level of feedback. With assistance from the robot, teleoperation can potentially improve safety, efficiency, and performance in complex environments.

This thesis investigates a framework for teleoperation that combines depth information and force feedback to enable intuitive and natural manipulation of objects by a human operator. The framework will implement the Elastic Strip framework by Oliver Brock to provide guidance to the operator and convey the desired trajectory for object grasping and manipulation through force feedback. Additionally, uncertainty in obstacle positions will be considered, and probabilistic algorithms will be used to plan a dynamic, collision-free trajectory for grasping. A grounded exoskeleton is integrated as a user interface to provide force feedback to the operator, enhancing the user's situational awareness and helping to improve the performance of teleoperated manipulation tasks.

The performance of the proposed framework will be evaluated through a series of experiments to analyze the effectiveness of the shared control approach in enabling accurate and efficient manipulation of objects in a variety of scenarios. These scenarios range from simple pick and place tasks to more complex assembly tasks, such as peg-in-hole. The context of application in this project is space exploration, with the eventual goal of assisting astronauts in teleoperation tasks. Additional methodologies, including reinforcement learning, will also be explored to further enhance the efficiency of the system. This will involve identifying scenarios where the assistive algorithm fails, and the expert, in this case, the operator, takes over the task. This will eventually lead to a smarter system over time. Overall, this thesis presents a novel framework for teleoperated manipulation that incorporates depth information, force feedback, and a grounded exoskeleton to enable intuitive and effective object manipulation by a human operator.

External Supervisor: Dr. Neal Y Lii
Internal Supervisor: Elodie Huesing, M.Sc. RWTH

Eidesstattliche Versicherung

Ajithkumar Narayanan Manaparampil Matrikel-Nummer: 415918

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Master Thesis mit dem Titel

Enhancing Manipulation Performance with Grounded Haptic UI Teleoperation

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Aachen, 23 August 2023

Ajithkumar Narayanan Manaparampil

Belehrung:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Aachen, 23 August 2023

Ajithkumar Narayanan Manaparampil

The present translation is for your convenience only.
Only the German version is legally binding.

Statutory Declaration in Lieu of an Oath

Ajithkumar Narayanan Manaparampil Student number: 415918

I hereby declare in lieu of an oath that I have completed the present Master Thesis entitled

Enhancing Manipulation Performance with Grounded Haptic UI Teleoperation

independently and without illegitimate assistance from third parties. I have used no other than the specified sources and aids. In case that the thesis is additionally submitted in an electronic format, I declare that the written and electronic versions are fully identical. The thesis has not been submitted to any examination body in this, or similar, form.

Aachen, 23 August 2023 Ajithkumar Narayanan Manaparampil

Official Notification:

Para. 156 StGB (German Criminal Code): False Statutory Declarations

Whosoever before a public authority competent to administer statutory declarations falsely makes such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment not exceeding three years or a fine.

Para. 161 StGB (German Criminal Code): False Statutory Declarations Due to Negligence

(1)If a person commits one of the offences listed in sections 154 to 156 negligently the penalty shall be imprisonment not exceeding one year or a fine.
(2) The offender shall be exempt from liability if he or she corrects their false testimony in time. The provisions of section 158 (2) and (3) shall apply accordingly. I have read and understood the above official notification: :

Aachen, 23 August 2023 Ajithkumar Narayanan Manaparampil

Contents

Formula symbols and indices

q	radian	Robot joint angle
τ	Nm	Robot joint torque
k_{att}	–	Attractive force constant in artificial potential field
k_{rep}	–	Repulsive force constant in artificial potential field
F_{att}	N	Attractive force in artificial potential field
F_{rep}	N	Repulsive force in artificial potential field
U_{att}	J	Attractive potential in artificial potential field
U_{rep}	J	Repulsive potential in artificial potential field
ρ	m	Current robot pose in artificial potential field
ρ_g	m	Goal pose in artificial potential field
ρ_b	m	Distance between robot pose and obstacle
ρ_o	m	Influence region of obstacle in artificial potential field
$F_p^{\mathbf{external}}$	N	External force at point p in Elastic strip framework
$F_p^{\mathbf{internal}}$	N	Internal force at point p in Elastic strip framework
k_c	–	Constant defining contraction gain in Elastic strip framework
K	–	Flag indicating user deviation from trajectory
M	–	Flag indicating movement outside the deadzone
K_P	–	Stiffness matrix in impedance control
K_D	–	Damping matrix in impedance control
A	–	Set consisting trajectory waypoints
F	–	Set consisting haptic forces
F_f	N	Assisitve force at one particular instance

F_u	N	Force applied by user at one particular instance

List of abbreviations

General abbreviations

APF	Artificial Potential Field
ESF	Elastic Strip Framework
LWR	Light Weight Robot
RRT	Rapidly exploring random tree
NASA	National Aeronautics and Space Administration
DLR	Deutsches Zentrum für Luft- und Raumfahrt
TLX	Task Load Index

1 Introduction

The problem statement tackled in the current thesis was derived from a space robotics application involving the robot Justin. Justin is a partial humanoid on a mobile platform [Justin] as shown in Figure 1.1. It is intended to perform certain tasks on a planet's surface while being teleoperated by on-orbit astronauts. These tasks include the manipulation of objects using a 7 DoF robotic arm. One of the tasks includes the insertion of a dip into a processing unit as shown in Figure 1.1. It has been observed during the training sessions that on-orbit astronauts found it particularly difficult to manually complete the task using teleoperation. This task becomes particularly tricky to perform as the only feedback available to the operator is the video feed through the camera mounted on Justin's head. The assumption is that teleoperation will be easier for the astronauts via an extension of the feedback by means of an assistive guiding force.

(a) Justin dip insertion (before) **(b)** Justin dip insertion (after)

Figure 1.1: Justin Dip Insertion: before (a) and after (b)

This thesis investigates a framework for teleoperation that combines depth information from the avatar robotic system and force feedback to enable intuitive and natural manipulation of objects by a human operator. The framework establishes a method to calculate the optimal trajectory to be followed by the user to complete the task. The information about the optimal trajectory is then conveyed to the user in a form that does not add to the cognitive workload. The methods explored to convey this path information are via a visual representation of the trajectory on the user interface and also by converting the path information into appropriate force vectors. These force vectors can then be communicated to the user using a haptic feedback device.

In the current context, the types of feedback devices generally used vary as per the experiment settings and environment. Currently, the haptic device used for space teleoperation

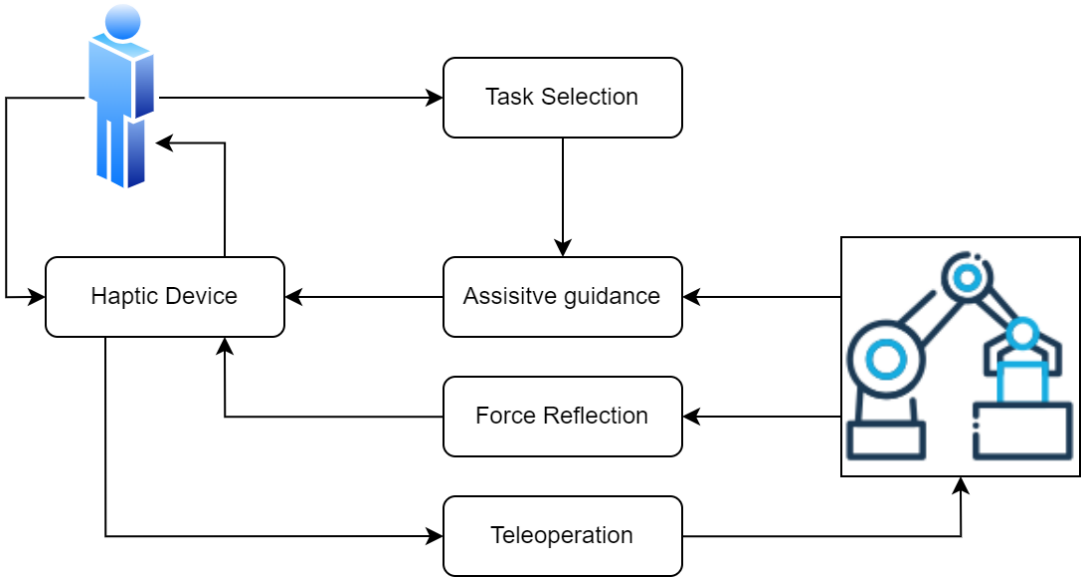

Figure 1.2: Proposed framework for assistive guidance

tasks is a device named "Sigma" by force dimension. Future implementations will potentially use "Exodex" a new experimental haptic device that is a grounded exoskeleton. Considering the wide range of haptic devices that can be integrated, one of the key limitations is to develop the framework to be generic and capable of integrating multiple haptic devices. The assistive teleoperation framework developed during this thesis is tested using a simple haptic device consisting of a joystick mounted on top of a 7 DoF robotic arm and a 7 DoF avatar robotic arm replicating the right arm of the robot Justin. A single task is set up as a commission study that requires the user to navigate the reference frame of the end-effector to a particular pose in the task space. The efficiency of task completion is then compared for systems with and without the assistive teleoperation framework.

The thesis is structured to provide a comprehensive exploration of the work. The subsequent structure include a literature review in Chapter 2, methodology in Chapter 3, implementation in Chapter 4, results and discussion in Chapter 5, conclusion in Chapter in 6, and outlook in Chapter 7. The literature review provides a brief background regarding the current status of research and development in the field of assistive teleoperation. The methodology chapter takes into account all the methods described in the literature review and the constraints as per the current case study to put forth a method to achieve the goal of designing an assistive teleoperation framework. The implementation chapter elaborates on the hardware and software used for the thesis. It explains in detail the software design and the procedure to deploy it on the hardware. The Results and Discussion chapter analyzes the data collected during the validation study and offers a concise overview of the effectiveness of the assistive teleoperation framework. The Conclusion chapter delves into the insights derived from the design and implementation process of

the thesis and addresses any outstanding research inquiries. The Outlook section outlines possible directions for further research to improve the existing framework.

2 Literature review on teleoperation strategies

Teleoperation is used for remote tasks that either cannot be directly carried out by a human or it cannot be automated. In most cases, these tasks are complex, and unstructured and require a certain amount of human skills, knowledge, and decision-making capability [OPV19]. The range of applications includes remote explorations like underwater exploration and space exploration, assistive technology for individuals with a physical disability, for undertaking tasks in hazardous environments, etc. As the applications and development of remote teleoperation systems are in demand, research towards simplifying the use of such systems is gaining attention. This literature review provides a brief overview of the classification, strategies, and research in teleoperation and eventually dives into the subcategory of assistive teleoperation.

2.1 Classification of teleoperation systems

The control of a remote system or robot from a distance can be achieved using various methodologies and various levels of precision. Over the years teleoperation has evolved into various segments, with some clear distinctions but also common features shared by all types. The broad classification of teleoperation can be into four types: Direct Teleoperation, Shared Control Teleoperation, Supervisory Teleoperation, and Telepresence.

1. Direct Teleoperation: In direct teleoperation, the motions of the master system are directly replicated by the avatar robot. It ideally provides real-time control and is normally used for tasks that require precise manual control.

2. Shared Control Teleoperation: This strategy combines aspects of autonomy and manual control to achieve the desired task. The user's inputs are taken as the reference for guidance and then combined with an autonomous control algorithm to complete the specified task [GTT22]. This method is mostly used to increase stability and efficiency.

3. Supervisory Teleoperation: In this strategy, user inputs are in the form of high-level commands which are interpreted by the robots and acted out. The user has the capability to monitor and intervene whenever necessary. The system has two separate levels of control [FS67]:

    a) Direct control of the manipulator by a computer at the remote site, which processes feedback and makes relatively simple routine decisions.

b) Supervision by an operator who occasionally sets sequences of subgoals for the remote device, modifies its parameters, perhaps, and compensates for its limited decision-making capability.

This strategy is normally used when the control space is high dimensionality and too complex to rely solely on the user's input.

4. Telepresence: Telepresence allows the operator to be remotely present at a distant location and provide complete immersion and interaction with the remote environment. To provide perfect telepresence signals pertaining to all human senses should be transmitted to the operator or master. As smell and taste are rarely used during robot teleoperation, the major focus during the implementation of telepresence systems is vision, touch, and hearing. The sense of touch regarding telepresence can be divided into kinesthetic or force feedback and haptic or tactile feedback. But generally, both force and tactile feedback are considered under the broad term of haptic feedback [Lic07]. Another term that has been widely used in a similar context is multimodal teleoperation. Multimodal teleoperation also aims to provide an auditory, visual, and haptic interaction between the user and the remote environment [TMG20].

A teleoperation system consists of at least one master robot locally manipulated by an operator and, at least one slave robot that remotely mimics the maneuvers of the master robot to perform the operation on an environment. A communication channel is then set up between the master and slave robot to transmit the necessary information. But it is not always practical or necessary to just have a single master or slave robot within the system. So based on the number of master and slave robots present in the system, the teleoperation can be classified as [SAP18]:

1. SM-SS: Single master and a single slave
2. MM-SS: Multiple masters and a single slave
3. SM-MS: Single master and multiple slaves
4. MM-MS: Multiple masters and multiple slaves

Apart from the major teleoperation strategies, there are other features within teleoperation that are interchangeable as per the application. A typical teleoperation system consists of a master manipulator, a slave manipulator, a human operator, and the operated environment [ZS00]. Based on the direction of motion and/or forces being transferred, the teleoperation systems can be classified as follows:

1. Unilateral teleoperation: If only the master motion and/or forces are transmitted to the slave, the teleoperation system is called unilateral [ZS00]. This type of information flow is implemented only in a single master and a single slave teleoperation system as it requires one-to-one correspondence.

2. Bilateral Teleoperation [NDV21] [HS06]: In addition to the master motion and/or forces being transmitted to the slave, if the slave motion and/or forces are transmitted to the master, the system is called bilateral. Similar to unilateral teleoperation, bilateral teleoperation is also possible to be implemented on a single master and a single slave teleoperation system.

3. Multilateral Teleoperation: When the teleoperation system includes an MM-SS, SM-MS, or MM-MS configuration, the system contains not only a one-to-one correspondence but also includes collaborative scenarios between multiple operator-master sets and/or multiple slave robots. Such teleoperation systems are called multilateral [SAP18].

The above-mentioned categories are general trends found in teleoperation research. But there are other customized hybrid approaches that do exist as per specific application scenarios. The control signal generation itself can lead to different systems. This is largely dependent on the type of control device used and varies based on factors like available workspace, presence of force feedback, degrees of freedom, etc. But the major distinguishing factor that differentiates the teleoperation control is based on which parameter from the operator side is used to control the relevant parameter on the avatar side. These strategies can be subdivided into control-level approaches based on whether it is a unilateral or bilateral teleoperation system. In the case of a unilateral system the classification is as follows:

1. Position - Position: In this case, the position of the control device directly decides the position of the remote robotic system. This position is generally the cartesian space coordinates but can also refer to joint space or a combination of both. This strategy is used when the workspace of the control device and the remote system are approximately similar and comparable [PSK11].

2. Position - Velocity: Position-velocity-based control controls the velocity of the remote robotic system based on the position of the control device. This is normally used when the workspace of the remote system is comparatively much larger than that of the control device [PSK11].

In the case of a bilateral teleoperation system, the classification based on the control scheme is as follows:

1. Position - Position: In this case, the force reflected to the operator is proportional to the position error and scaled as per the master controller gain. Similarly, the force exerted by the slave is also proportional to the position error but scaled as per the slave controller gain. The position error is the difference between the master and slave positions [FBM].

2. Force - Position: This control scheme is a modification of the position-position based control scheme. The force reflected to the operator is proportional to the interaction force between the slave and environment [FBM].

3. Force - Velocity: In this case, the force reflected by the master and the slave velocity is calculated based on the values of master velocity, interaction force between the slave and the environment, communication time delay, and line impedance. Line impedance is a parameter that defines the remote environment impedance[FBM]. This control is normally used in case of time delay during teleoperation.

Another way of classification in teleoperation is inspired by network theory that discusses variables such as effort and flow. As per the analogy, the effort and flow correspond to mechanical variables force and velocity respectively. This scheme considers the number of quantities communicated i.e. the number of virtual channels used for the interconnection between the master and slave. So as per this scheme, the force-velocity control architecture is a two-channel architecture [HFB]. The broad overview of major classification types are depicted in the Figure 2.1

2.2 Related work

In assistive teleoperation, the robot helps the user accomplish the desired task, making teleoperation easier and more seamless. Rather than simply executing the user's input, which is hindered by the inadequacies of the interface, the robot attempts to predict the user's intent and assists in accomplishing it [AKR13]. The concept of assistive teleoperation is an evolving field and has been under extensive research in recent years. The current thesis is roughly relevant to the sections of Supervisory and shared control teleoperation. The aim of the thesis is to assist the astronauts using a guiding virtual force to complete the specified task. Similar problem statements in different domains have been previously tackled with different end goals and various strategies. Shared control strategies can improve the efficiency of tasks during teleoperation [PSM16][BAH13]. Often most of these approaches make use of virtual fixtures . Virtual fixtures are a form of constraint-based or guidance-based assistance that prevents the user from straying away from the path [AMO07]. In most of these approaches, the decision is made at the interface. There are

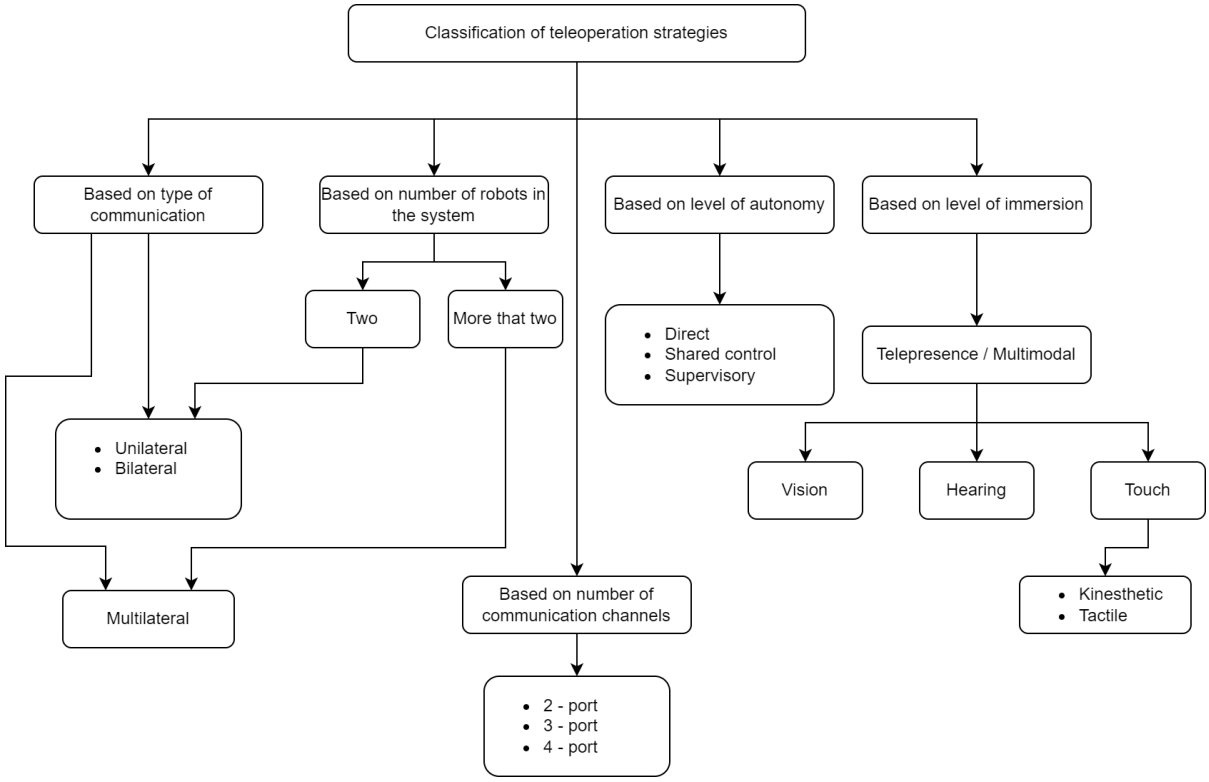

Figure 2.1: Classification of teleoperation strategies

other approaches in which the intentions of the operator and the guidance system are mixed at the state level ie after the interface level [KAW11][SMS16][ETD17][DS13]. In the first method the user is fully aware of the assistive guidance which is generally referred to as Haptic Shared Control (HSC), but the in the latter case the states given by the operator and the assistive system are fused through a weighted combination which is known as State Shared Control (SSC)[ZHC18]. The weights are decided using a wide variety of methods. The virtual fixtures in some cases are manually coded for each task[AMO07]. Another approach is to learn the virtual fixtures from demonstrations. The demonstration is done with the same hardware, stored and refined over multiple demonstrations. One of the major advantages of the shared control method is to reduce the operator's cognitive load. So naturally other applications include assistance for people with a motor disability.

One such application of shared control is represented by [QHI20]. This method defines task-specific skills as Finite State Machines in which each phase specifies task-relevant input mappings and active constraints. This would classify as a State shared control method as the user is unaware of the assistive guidance and it is mixed with the user input after the interface stage. An improvement over this technique is presented in [BQH21]. This method adds supervised autonomy along with the Shared control template and also provides a way to seamlessly switch between those as both methods use the same action

representation.

The previously discussed methods use a manual method of task learning but alternative learning-based approaches provide the capability to learn in arbitrarily noisy environments and handle high-dimensional complex tasks. Using reinforcement learning and using online feedback from the user helps in building such a system. But the amount of human-in-the-loop data required for this function is huge. One method proposed to tackle this hurdle is to use offline pre-training to acquire a latent embedding space of high-level robot behaviors. The system then uses the online user feedback to map the user input to this high-level behaviour[CGR22]. This significantly reduces the dependency on huge online training data and can manage with sparse user feedback.

Another interesting approach in task data collection is to use a virtual environment to collect the task and then use the learnings on actual hardware. One such method is described in [CHM21]. It uses a transformer to provide a window of potential states and actions over a long horizon with no additional computation time during the teleoperation. The human intention can be injected at any location within the transformer sequence if the model-predicted sequence is not as per the user's requirement. If the predictions are correct the user can do nothing and allow the transformer to guide the robot and intervene whenever required.

As it is evident from all the methodologies described above, there exists a range of options to provide assistance to the user during teleoperation. The current thesis focuses on achieving this assistance using a guiding force pulling the user towards the goal while avoiding obstacles. One of the earliest mentions of this is by using an artificial potential field-based algorithm to help the user navigate the obstacles. These pulls and pushes from the APF algorithm are combined with human input to specify the direction that the robot will go. This system has no global awareness of the goal to be accomplished and suggests direction just based on the current robot pose, human input, and obstacle poses. Since the vector summation might allow for some obstacle-avoiding behaviors to cancel out leading the robot into an obstacle, a safeguarding behavior is implemented to veto the direction [CG02].

Similarly, an implementation of a modified APF has been utilized to achieve a guiding behavior in [GTT22]. The modified APF tries to eliminate the local minima problem and also has awareness about the goal point to be reached. In this, no force feedback is given to the user, making this a unilateral teleoperation system. The only form of feedback present was the camera feed showing the experiment scene. This was also a conscious decision in order to survey whether the robot's movements appeared natural to the user.

In this case, certain users felt the robot movement was unnatural even though it was the most efficient maneuver to complete the task.

The architecture mentioned in [OPV19] that was used in a unilateral teleoperation task roughly provides a brief overview of using a motion planner to assist the user. The architecture combines the user input along with the motion planner input to achieve the required robot cartesian space velocity. The user input in this case is based on an interoperable teleoperation protocol that allows switching between position-position control mode and position-velocity control mode. The architecture roughly aligns with the goals of this thesis except for missing the force feedback segment of the bilateral teleoperation framework.

Force feedback is an important segment for providing assistance during teleoperation. The type of master control device or haptic device and the difference in workspace volume play an important role in defining the method to calculate and implement guiding forces. If the difference in workspace volume is too large the most preferred way of teleoperation is postion-veloctiy control scheme. Deadzone implementation on master device to prevent sensitive movement of robot due to small displacement of the master device is another consideration during force feeedback implementation. In some applications, an additional force is always present on the haptic device to align it to its initial reference position whenever the user lets go the device [FR07].

The next chapter describes a methodology that is derived from previously mentioned implementations of APF based methods to guide the robot towards the goal, but also parallely implementing a force feedback system that provides additional information to the user.

3 Methodology

The current thesis focuses on improving the efficiency and performance of teleoperation tasks using a form of bilateral haptic shared control teleoperation method. The framework developed during the course of this thesis is called Assistive teleoperation framework. The assistive teleoperation framework provides additional decision-making hints compared to a bilateral teleoperation strategy to the operator in the form of visual, haptic, or auditory aids. This additional feedback used in the assistive teleoperation framework is different from the direct sensor feedback (eg: the force reflection) as it extracts high-level information from the sensor data considering the task that the user has undertaken. This information is then passed to the user to help make an informed decision. The entire

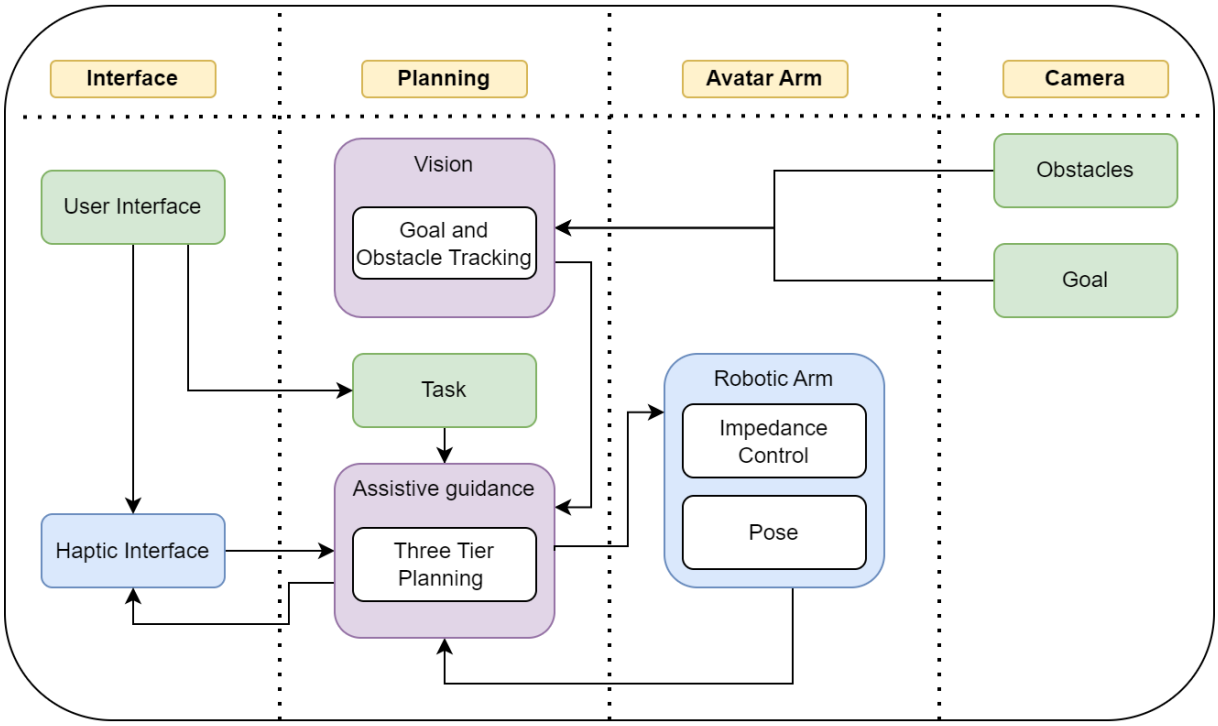

Figure 3.1: Overview of the Assistive Teleoperation Framework (Green blocks indicate input parameters to the system from sensors and the user, Blue blocks indicate the input and output of the robot system, Violet blocks indicate the processing units)

system can be divided into four segments as shown by the segment headings in Figure 3.1. The interface segment includes the user interface and the haptic interface. The user interface displays the live camera feed, the 3D simulation representing the digital twin of the robot, and buttons to control the activation and deactivation of assistive guiding force. The live camera feed also has an overlay that highlights the goal position as per the task chosen by the user and the optimal trajectory to accomplish the task is also projected on

the live camera feed. The user can decide when to start the assistive guidance based on the decided task using the user interface buttons as described in Section 4.4. The planning segment has two major components: vision, and assistive guidance. The vision component continuously keeps track of the obstacles in the dynamic environment and also the goal position. The assistive guidance calculates the trajectory and the corresponding force feedback based on the task selected by the user, the goal position, and the obstacles in the environment. The assistive guidance also recursively keeps track of any dynamic obstacle and modifies the trajectory and the force feedback accordingly. The force feedback is sent back to the user via the haptic device and the trajectory to follow is sent to the avatar robotic arm. The relationship between the user input on the haptic device based on the assistive guidance and the trajectory followed by the robot arm is explained in Section 3.4. The current chapter is divided into five sections which include Planner, Control Strategy, Teleoperation, Haptic Interface, and Computer vision. Next in the methodology is the Planner which explains the algorithm used to calculate the optimal trajectory to reach the goal pose while avoiding static and dynamic obstacles. The control strategy section explains the generation of low-level input signals for the avatar robotic arm using impedance control which inherently also depends on the calculated optimal trajectory and the haptic user input. The final section of the methodology is the computer vision segment that identifies the goal pose and obstacles to send to the planner.

3.1 Planner

The task in consideration is the manipulation of objects using a 7 DOF robotic arm. In this case, the high-level information consists of a proper trajectory to achieve the desired task considering all the static obstacles in the environment. The global trajectory is calculated using a motion planner. Eventually, during teleoperation, the planner should also have the capability to use the information about dynamic obstacles and modify the trajectory in real-time to avoid a collision. This trajectory information is then transmitted to the operator in the form of force feedback. The planning strategy consists of three components to achieve this behavior: the global, mid-level, and local planner. The global planner calculates the global trajectory, the mid-level planner modifies the global trajectory up to a certain distance in front of the current robot position in case the global trajectory is not viable anymore, and the low-level planner responds to the last-minute obstacle avoidance maneuver. These components are further explained in the following sections.

3.1.1 Global planner

The global planner in consideration is a modified version of rapidly exploring random tree-based motion planning. Rapidly exploring random trees (RRT) is a popular motion planning algorithm used to explore high-dimensional configuration spaces. RRT creates a tree-like representation of the search space and then finds a feasible path toward the defined goal. The algorithm starts by randomly sampling points in the configuration space and then checks the feasibility of these points based on whether they are spawned within an obstacle or lie in the free space. Afterward, it connects these points and makes sure the edge connecting the points doesn't pass through an obstacle [LaV06] [LK01]. It roughly repeats this procedure till the goal point is reached. The time complexity of RRT is O(n log n) where n is the number of nodes in the tree. The convergence rate of RRT can be slow in high dimensional space. A new motion planner named RepMap was proposed in [LA18] that effectively exploits the repetitiveness of a set of tasks with small variations to efficiently compute new motions. The current thesis uses an adaptation of the motion planner in which Uniform RRt is used in parallel to the RepMap algorithm [CSM15]. This ensures the probabilistic completeness guarantees of the Uniform RRT and the new solutions gained from the Uniform RRT can enrich the experience of the RepMap.

3.1.2 Dynamic real-time planning

Dynamic real-time planning is achieved by a combination of mid-level and local planners. The local planner in this case is an artificial potential field-based reactive planner. The local planner only considers the imminent obstacles and tries to navigate to the next waypoint. Using the local planner alone with a goal point can lead to local minima problems causing the avatar robotic arm to be stuck at a point away from the goal. The global planner keeps track of the goal point and keeps a rough estimate of the path thus helping the local planner to get out the local minima issue. There is a second category of dynamic obstacles that should be taken care of in case of a practical scenario. These obstacles were not detected during the global planning phase but appeared later during teleoperation. These obstacles are not in the immediate vicinity of the robot so might not immediately collide with the robot instead they are a few waypoints ahead on the trajectory. The mid-level planner takes care of these obstacles and modifies the global trajectory put forth by the global planner to avoid a collision. This partially reduces the dependency on the local planner leading to ample time to modify the global path. The mid-level planning is achieved by using a modified version of the elastic strip framework[BK02]. This trajectory information is encoded in the form of force vectors and then transmitted to the operator side. To better understand dynamic real-time planning

the following subsections explain in brief the artificial potential field and elastic strip framework.

3.1.3 Artificial potential field

Artificial potential field is a method used in robotics motion planning to guide the agent or robot toward its goal while avoiding obstacles. This approach uses no prior planning and is totally dependent on the current sensory input and position of the robot, which classifies it as a reactive approach to planning. This method uses the concept of potential fields to navigate the agent toward a goal. Each point in the environment is defined as a scalar function which is calculated based on its proximity toward an obstacle and the goal. This basically conveys the suitability of being at a particular state in space for the robot. The potential field which decides this suitability is divided into two components namely attractive and repulsive potential fields. The attractive potential field is designed to pull the robot toward the goal. The attractive forces decrease as the robot gets closer to the goal. The negative potential field is designed to push the robot away from the obstacles. The repulsive forces increase as the robot gets closer to an obstacle. The robot aims at reducing the net force to zero. The state with the net force zero is an expected behavior when the robot reaches the goal assuming the goal is not under the influence of an obstacle. The attractive potential function is defined as :

$$U_{att}(q) = \frac{1}{2} k_{att} (\rho - \rho_g)^2 \tag{3.1}$$

where k_{att} is the attractive force constant, ρ and ρ_g stand for the current pose of the robot in task space and the pose of the goal. The repulsive potential function is defined as:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\rho_b} - \frac{1}{\rho_0} \right)^2, & \rho_b \leq \rho_0 \\ 0, & \rho_b > \rho_0 \end{cases} \tag{3.2}$$

where k_{rep} is the repulsive force constant, ρ_b is the distance between the pose of the robot and the obstacle in consideration, and ρ_0 is the influence region of the obstacle [XLS23]. The negative forces affect the robot only when the robot is within the influence region of the obstacle. The Equation 3.2 is defined for a single obstacle but in practical scenarios, the total repulsive force is calculated by the summation of all the repulsive forces affecting the robot. The attractive and repulsive forces derived from the respective potential field

are shown as:

$$F_{att}(q) = -\nabla U_{att}(q) = k_{att}(\rho - \rho_g)$$
$$F_{\text{rep}}(q) = -\nabla U_{\text{rep}}(q) = \begin{cases} k_{rep}\left(\frac{1}{\rho_b} - \frac{1}{\rho_0}\right)\frac{1}{\rho_b^2}, & \rho_b \leq \rho_0 \\ 0, & \rho_b > \rho_0 \end{cases} \tag{3.3}$$

The net force acting on the robot at any given time is:

$$F_{total} = F_{att} + \sum_{i=1}^{n} F_{irep} \tag{3.4}$$

where n is the number of obstacles that have the robot within their influence region and F_{irep} denotes the repulsive force exerted by i^{th}obstacle. The artificial potential field approach is relatively simple and computationally efficient but has certain drawbacks. The algorithm does not have global planning capability and does not maintain a representation of the world around them. This might lead to drawbacks such as running into local minima and showing oscillating behavior. These drawbacks can be addressed using a global planner along with an elastic strip framework, which is an improvement of the artificial potential field.

3.1.4 Elastic strip framework

The elastic strip approach provides a framework to execute a planned motion while allowing reactive obstacle avoidance. This is achieved by incremental modification of the previously planned motion. In this method, a volume of free space around the candidate path given by the global planner is calculated. This volume of free space has other possible robot configurations, which are called homotopic paths. Whenever there is an obstacle exerting influence over the points on the candidate path, the virtual forces push the candidate path prompting it to select a new configuration from the set of homotopic paths. These virtual forces are derived from two potential field components, the external potential field, and the internal potential field. The external potential field is the same as defined in Equation 3.2. The resulting forces acting due to this external potential is given by F_{rep} denoted by Equation 3.3. The external forces are sufficient to select a new candidate path but if an obstacle recedes from the path the deformation would still persist. Considering a scenario where all the obstacles receded back but as the path has already been modified to avoid the obstacles it is currently in a sub-optimal state. To recede the path back to the original length or the optimal state, virtual springs are attached to control points before and after the current point in consideration. This leads to an internal contraction force that pulls the path back to its original form when the obstacles

recede. Consider p_j^i as the position vector of the waypoint attached to the jth link of the robot. The robot is currently in configuration q_i with the previous configuration as q_{i-1} and the next configuration as q_{i+1}. The internal contraction force [BK02] is defined as follows:

$$\mathbf{F}_{i,j}^{\text{internal}} = k_c \left(\frac{d_j^{i-1}}{d_j^{i-1} + d_j^i} \left(p_j^{i+1} - p_j^{i-1} \right) - \left(p_j^i - p_j^{i-1} \right) \right) \tag{3.5}$$

where d_j^i is the distance $\left\| p_j^i - p_j^{i+1} \right\|$ in the initial, unmodified trajectory and k_c is a constant determining the contraction gain of the elastic strip. These forces are then mapped as joint torques using the robot's dynamic model.

$$\tau_{avoidance} = \sum_{p \in \mathcal{R}} J_p^{\mathrm{T}}(\mathbf{q}) \left(\mathbf{F}_p^{\text{internal}} + \mathbf{F}_p^{\text{external}} \right) \tag{3.6}$$

where $J_p(\mathbf{q})$ represents the Jacobian at point p on the robot $\mathcal{R}$ and $\mathbf{F}_p$ designates a force acting at p. Forces resulting from external and internal potentials are simply mapped into torques; these torques are then used to simulate the behavior of the robot in a configuration along the elastic strip. But the computation complexity of the elastic strip framework is comparatively higher than the artificial potential field. In the Elastic Strip Framework (ESF), there are situations where obstacles might cross the path of the robot, causing the robot to deviate from its intended path even after the obstacle has already passed and there exists a straight line path. This issue arises due to the dynamic nature of the elastic strip and the continuous adjustments it makes based on the presence of obstacles. If an obstacle comes to rest on the optimal path as shown in Figure 3.2 (c), the

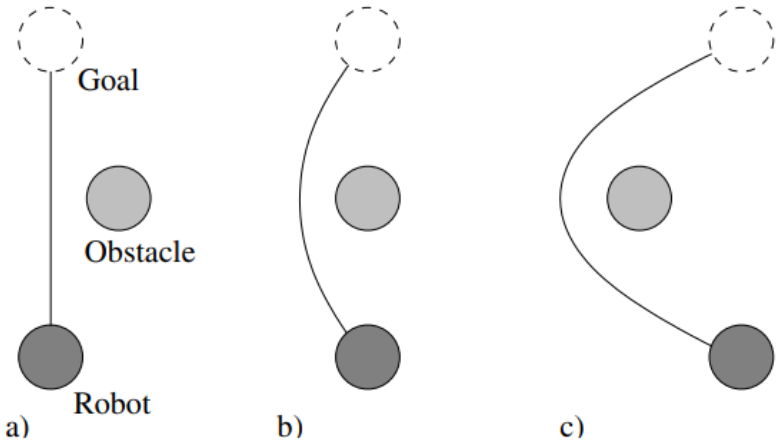

Figure 3.2: A candidate path becomes sub-optimal after the environment changes[BK02]

internal forces will not be able to reconnect the elastic strip as an apparent straight path

during this procedure. A straight path seems to be feasible and optimal in such a scenario. This difficulty can be avoided by maintaining two versions of the elastic strip, one which is continuously modified to avoid the obstacle and one which is broken in an attempt to let it pass through [BK02]. But that potentially adds complexity and increases memory consumption. To tackle this issue, modifications can be made to the ESF to address the following two concerns:

1. Computational Complexity: The ESF involves continuous monitoring and updating of the elastic strip to ensure obstacle avoidance and smooth navigation. This can introduce computational complexity, especially in real-time scenarios.

2. Path Modification by Distant Obstacles: In some cases, the elastic strip may react to obstacles that are not in the immediate vicinity of the robot. This can result in unnecessary path modifications or deviations.

To address the above-mentioned concerns, a range limitation can be set to consider only the obstacles and waypoints within a certain distance from the current position of the robot as relevant for path adjustment. This distance is called the Illumination region. This will inherently reduce the number of obstacles and waypoints to be monitored and at the same time, the influence of the distant obstacles can be limited. So if the scenario from Figure 3.2 (c) occurs at a further distance from the current robot position, the path won't be modified in the first place unless the obstacle is within the decided range. This will eliminate the need to maintain two versions of the elastic strip for all waypoints outside the Illumination region.

3.1.5 Three-tier planning system

The three-tier planning system consists of the global planner, mid-level planner, and local planner, each serving a specific role in the robot's motion planning process. The local planner is responsible for monitoring and responding to dynamic objects in the immediate vicinity of the robot. Unlike the global planner, it doesn't rely on a global representation but utilizes sensor readings and the next global waypoint in line to make decisions. If a collision is imminent as shown by obstacle 1 in Figure 3.3, the local planner deviates from the global planner's waypoint, prioritizing collision avoidance. Once the immediate danger is averted, it tries to realign with the global trajectory as quickly as possible. The mid-level planner has an illumination area as shown in Figure 3.3 that covers a range of waypoints ahead as well as behind the current robot pose. Any obstacles as depicted by obstacle 2 in Figure 3.3 detected within this illumination area ahead of the robot are used to modify the global trajectory. This modification follows the principles of the elastic strip

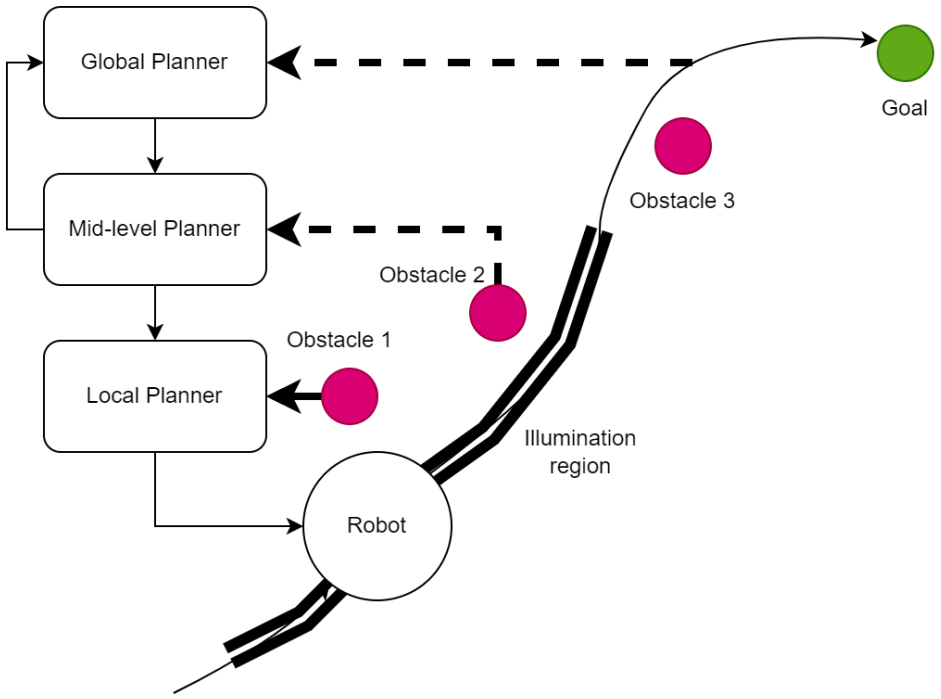

Figure 3.3: Planning system components of the assistive teleoperation framework handling their respective obstacle type

framework, allowing resources to be focused on specific sections of the trajectory instead of the entire path. By addressing obstacles that are in close proximity to the robot's current position, the mid-level planner prevents deviations caused by obstacles far ahead on the path that might intersect the trajectory before the robot reaches that point.

When the assistive teleoperation framework is activated by the user using the user interface, the high-level global planner is the first component within the planning system that becomes active. It determines an optimal global trajectory, taking into account the goal and obstacles present in the environment. This trajectory serves as a reference for both the mid-level and local planners, guiding their decision-making processes. By utilizing the optimized global trajectory, mid-level, and local planners can coordinate their actions in alignment with the overall mission objectives.

This three-tier planning system ensures a comprehensive approach to motion planning, integrating immediate collision avoidance, dynamic obstacle modification, and global optimization. It maximizes the utilization of available resources and enhances the robot's ability to navigate in complex environments while efficiently reaching its intended goal.

3.2 Control strategy

The control strategies define how the system should respond and adjust its behavior to achieve the desired outcome. The control strategies can be broadly classified as position control and torque control modes. In position control mode the goal is to accurately control the joint angles or end-effector position of the robotic arm. Position control mode uses inverse kinematics to track the end-effector pose to achieve the desired pose. But position control is known for its rigid behavior. This lack of compliance is not suitable for tasks involving interactions with other objects and uncertain environments. Torque control regulates the joint torques and in turn the end-effector forces. It is used in cases where the robot interacts with the environment and generates control input to achieve the desired force that is to be exerted on the objects. But there is always a risk of applying excessive force in unpredictable interactions with a dynamic environment. The previously mentioned drawbacks can be tackled by implementing an impedance control strategy [Hog85]. This control strategy adjusts the stiffness and damping of the system, thus regulating the force that is exerted on the objects. In the current scenario, the desired outcome is to follow the trajectory to achieve the desired task while maintaining a compliant behavior to respond to dynamic environment changes. Impedance control can be broadly classified into Joint and Cartesian impedance behavior. The current thesis uses a combination of both Joint and Cartesian-based control to achieve multiple desired goals.

3.2.1 Cartesian impedance control

Impedance control is a control strategy to dynamically tune the relationship between the force and position of a manipulator. The impedance controller takes the position as input and controls the torque on the joints considering the interaction with its surrounding. The input position can be in the cartesian space or the joint space of the robot and in turn decide the type of impedance control. The former is called Cartesian Impedance control where the Cartesian position of the end-effector is taken as an input. The latter is called joint impedance control where the joint angles are taken as input. Both these control strategies are implemented on a Kuka Light Weight Robot also referred to as LWR throughout this document. Figure 3.4 shows a PD-based impedance control in Cartesian space. It controls how the robot behaves during an interaction with the environment by adjusting the stiffness and damping. The output depends on the Cartesian space position error and its derivative, thus inherently controlling the positional error and damping

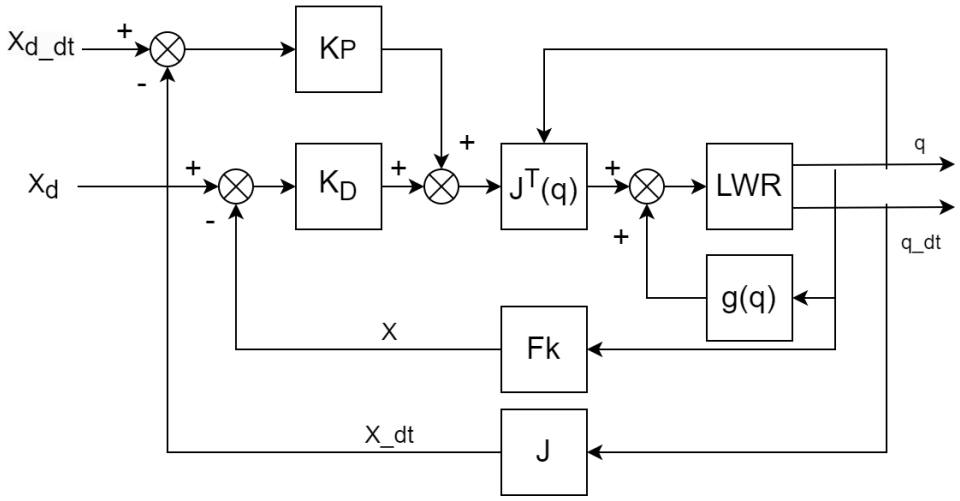

Figure 3.4: PD-based Cartesian Impedance Control [PCT20]

characteristics. The torques at the joints can be described as:

$$\tau = M(q)\ddot{q} + C(q, \dot{q}) + g(q) + \tau_{cartesian-ext} \tag{3.7}$$

where M is the mass matrix, C are Coriolis torques and g are gravity torques. The control law can be defined as :

$$\tau = J^T[K_P\,(x_d - x) + K_D\,(\dot{x}_d - \dot{x})] + M(q)\ddot{q} + C(q, \dot{q}) + g(q) \tag{3.8}$$

where x_d and x stand for the desired pose and current pose respectively. This results in $\tau_{cartesian-ext}$ to be as follows:

$$\tau_{cartesian-ext} = J^T[K_P\,(x_d - x) + K_D\,(\dot{x}_d - \dot{x})] \tag{3.9}$$

where K_P and K_D represent the stiffness and damping respectively.

3.2.2 Joint impedance control

Joint space impedance control is a subset of impedance control strategies used to dynamically control the relationship between force and position. In joint space impedance control, the position refers to the joint space position as shown in Figure 3.5. The control equation is similar to cartesian space impedance control except the external torque calculations depend on the joint angles rather than the Cartesian space coordinates. The torques at the joint are described as in Equation 3.7 where M is the mass matrix, C are

Coriolis torques and g are gravity torques. The control law can be defined as :

$$\tau = J^T[K_P\,(q_d - q) + K_D\,(\dot{q}_d - \dot{q})] + M(q)\ddot{q} + C(q, \dot{q}) + g(q) \tag{3.10}$$

This results in τ_{ext} to be as follows:

$$\tau_{joint-ext} = J^T[K_P\,(q_d - q) + K_D\,(\dot{q}_d - \dot{q})] \tag{3.11}$$

where K_P and K_D represent the stiffness and damping respectively.

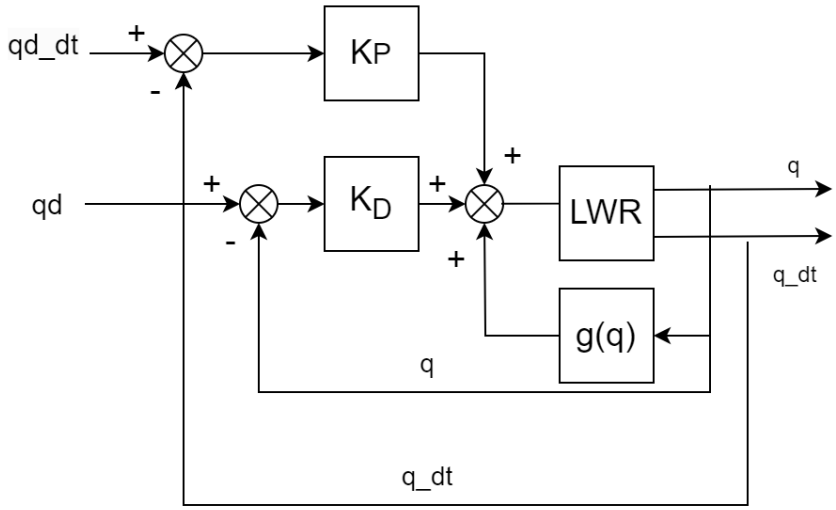

Figure 3.5: PD-based Joint Impedance Control

3.3 Combination of control strategy and three-tier planning system

The three-tier planning system and the control strategy together help to have a dynamic control over the motion of the avatar robotic arm. The three-tier planning system uses static as well as dynamic obstacle information to calculate and modify a global trajectory. The avatar robotic arm follows the trajectory using the Cartesian impedance control considering the user has opted to follow the trajectory which is explained in Section 3.4. The trajectory is in the form of Cartesian coordinates which are used as input to the Cartesian impedance controller. The impedance controller calculates the torque output of each joint required to reach the next waypoint on the trajectory all while being compliant. Each joint of the avatar robotic arm is continuously tracked to check for collision. If the joint is within the influence region of an obstacle, the low-level planner gives a Cartesian pose vector to avoid a collision. This vector is converted into appropriate joint torques by the Cartesian impedance controller. The joint impedance control is parallelly used to calculate the torque required to maintain a particular joint configuration throughout

the task execution, for example, maintaining the last joint to continuously point upwards while executing the trajectory. The control strategies along with the three-tier planning system help in executing a trajectory and maintaining a task-specific configuration while trying to avoid potential collisions.

3.4 Assistive teleoperation

The control strategies and planning system within an autonomous scenario can operate independently of the teleoperation system for controlling the avatar robotic arm. However, in the present use case involving mission-critical tasks, it is strongly recommended to maintain user supervision and the capability to assume control if the autonomous behavior deviates from the desired outcome. This section outlines an approach to integrate an assistive teleoperation setup with the control strategies and planning system detailed earlier. The command values published to the avatar robotic arm are a combination of

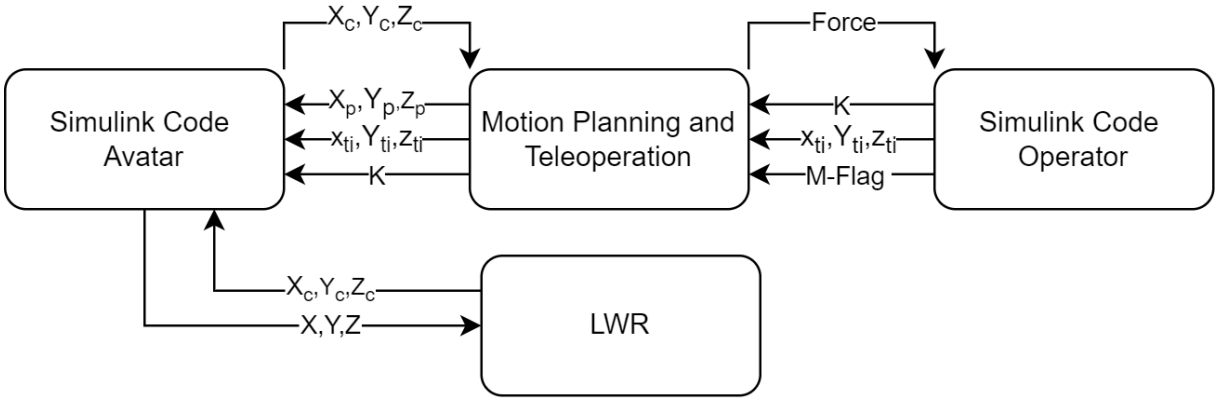

Figure 3.6: Control command and assistive force generation during assistive teleoperation

the output from the calculated assistive path and the teleoperation signals provided by the operator. The assistive path refers to the trajectory calculated by the planning system along with real-time modifications during the run time. The switching from the assistive to the manual control is based on the variable K which is calculated on the basis of the user's deviation from the calculated assistive path. If the variation is beyond the defined threshold, the value of K is equal to 0 leading to all the manual commands being published and vice-versa.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = M \left[K \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix} + (1-K) \left[\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} + \begin{bmatrix} x_{ti} \\ y_{ti} \\ z_{ti} \end{bmatrix} \right] \right] \; with\, K \in [0,1]\, and\, M \in [0,1] \quad (3.12)$$

where p stands for output of the motion planner, ti represents the increment values from direct teleoperation and c stands for the current robot pose values. In the above equation, the values $[x_p, y_p, z_p]$ have to be selected based on the current position, previous commands, and the value of K. The value of K is calculated as follows:

$$K = \begin{cases} 0, & \text{if } \cos^{-1}(\hat{F}_f \cdot \hat{F}_u) > th \\ 1, & \text{otherwise} \end{cases} \tag{3.13}$$

where th stands for the threshold value of the angular error that can be tolerated, F_f is the assistive force given to the user and F_u is the force exerted by the user on the haptic device. The value of M is decided based on the current position of the haptic device with reference to the dual dead zone which is explained in detail in Section 3.5.2. The force feedback given to the operator as shown in Figure 3.6 is a combination of the assistive guidance force and the force reflection. The assistive guidance force is calculated from the three-tier planning system from Figure 3.3 based on the obstacle and goal location. It is a virtual force guiding the user to the goal.

3.4.1 Assistive guidance

The assistive guidance to the user is in the form of force feedback. Ideally, the force should guide the user to the next computed Cartesian point considering all the obstacles and the goal position. But as the waypoints are in close proximity to the previous and next waypoints, the time taken by the avatar robot to reach the next waypoint is on average 200 milliseconds. Considering the short duration of travel time, the force vector corresponding to each waypoint is transmitted to the user for an average of 200 milliseconds. That would lead to a frequency of 5 Hz at which the force is changing. Initial trials indicated that this frequency of direction and magnitude change in force feedback is not differentiable for the user. The human operator is only capable of sensing a difference in force and direction if the temporal difference is above a threshold [Bur96]. Below the threshold, the force feedback would just manifest as random vibrations on the operator side, with no particular assistive characteristics. The respective perceptual threshold values for all kinds of stimuli put on the human body have been studied. Apart from very detailed information for every modality a human being can perceive, one major conclusion emerged from these studies: Human haptic perception often follows Weber's Law [Bur96]. Ernst Weber was an experimental psychologist who in 1834 first discovered the following implication

$$\frac{\Delta I}{I} = k \quad \text{or} \quad \Delta l = kI \tag{3.14}$$

where ΔI is called the difference threshold or the Just Noticeable Difference (JND). It describes the smallest amount of change of an (arbitrary) stimulus which can be detected just as often as it cannot be detected [Bur96]. I is the initial stimulus that is altered by the JND and the constant k describes the linear relationship between the JND and the initial stimulus [Bur96]. Weber's law is not fully applicable in all scenarios but some studies have shown that the human haptic perception system follows Weber's law quite well [JH92]. The factor k which signifies the magnitude of the change in a stimulus that can be perceived lies mostly between 5% and 15% [Bur96].

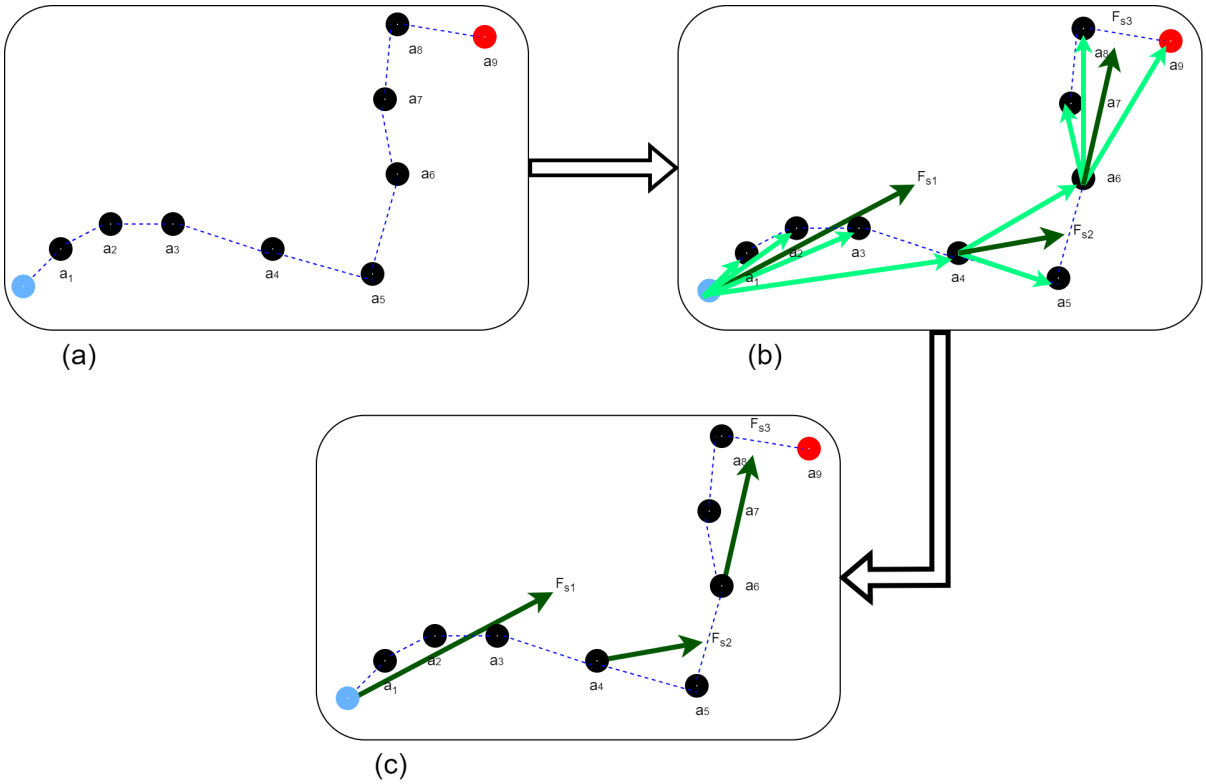

Figure 3.7: Assistive Force

The strategy to compute the assistive force feedback is defined as follows:

$$F = \{\} \quad and \quad A = \{a_1, a_2, a_3, ..., a_n\} \tag{3.15}$$

where F is an empty set that will eventually hold the assistive force values and A is a set of all the points in the trajectory as shown in Figure 3.7. To select the forces to be transmitted to the operator, all the trajectory points that fall within the threshold angle are collected. The average of these points is then transmitted as a force vector to the operator.

To select the forces to be transmitted to the operator, the vector from the first point to the second is calculated first. Then the angle between the vector from the first point to

the third. Starting with $n = 0$ and $m = n + 1$ the value of θ is calculated as follows:

$$\theta = \cos^{-1}((\hat{F_{a_n - a_{n+m}}}), (\hat{F_{a_n - a_{n+m+1}}})) \quad where \quad \theta <= threshold \tag{3.16}$$

The value of m is incremented till the value of θ gets above the threshold. Once it is above the threshold, F and A are updated as follows:

$$F = F \cup (1/(m-1) \sum_{l=n}^{m-1} (F_{a_l - a_{l+1}})) \quad and \quad A = A - (\{a_1, ..., a_m\}) \tag{3.17}$$

and the value of n is set to m and the value of m is set to n + 1. The steps from Equations 3.16 and 3.17 are repeated till A becomes an empty set. The above procedure maps a set of points in the trajectory to a constant average force vector that is transmitted to the haptic interface [HHC08]. Figure 3.8 depicts an example in which the trajectory waypoints from a_1 to a_4 transmit the force F_{s1} to the haptic interface, trajectory waypoints from a_4 to a_6 transmit the force F_{s2} to the haptic interface and trajectory waypoints from a_6 to a_9 transmit the force F_{s3} to the haptic interface

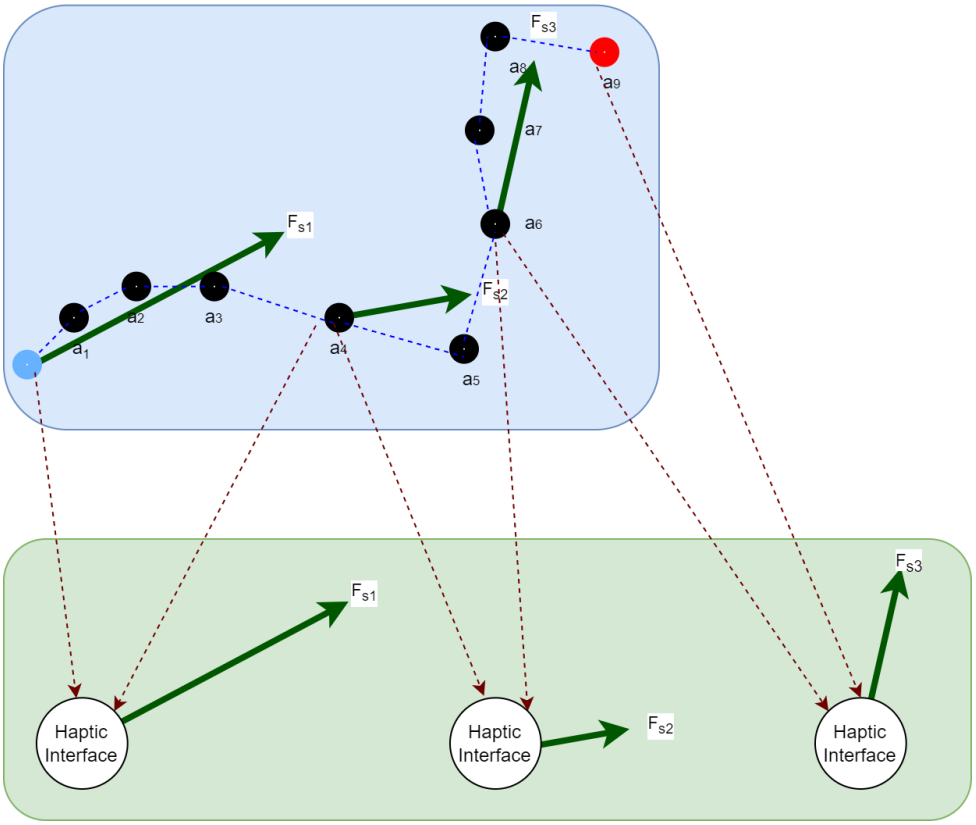

Figure 3.8: Assistive Force on Haptic Device

3.5 Additional features on haptic interface

The haptic interface enables the users to perceive the forces and sensations encountered by the robot. In this thesis, the user also receives additional virtual guiding forces that the teleoperated robot does not explicitly receive from the environment. These virtual guiding forces are also referred to as 'virtual fixtures' or 'virtual guidance'. This virtual guidance is calculated based on the intended task to be executed. The calculations for this guidance force are briefly explained in Sections 3.4 and 3.4.1

3.5.1 Center-Sprung mechanism

The haptic interface includes a joystick mounted on a 7 DoF KUKA Light Weight Robot (LWR) arm. The position-based center-sprung joystick control is used to control the avatar robotic arm. The center-sprung mechanism is implemented on the LWR with the joystick mounted on it, using the Cartesian impedance control scheme as shown in Section 3.2.1. In this case, a Cartesian point is set as the desired pose on the Cartesian impedance controller. This makes sure that the joystick returns to the set reference Cartesian point when released by the user and in turn causes the robot to stop. This leads to precise control of the avatar robotic arm and also prevents accidental collision. It has been observed that during a situation in which the avatar robotic arm is about to collide with an obstacle, one of the user tendencies is to release the joystick. Considering a scenario where the center-sprung had not been implemented, the joystick would stay at the last left position, and the avatar robotic arm would continue moving toward the obstacle as it is position-velocity-based teleoperation. But on the contrary, as the center sprung mechanism has been implemented, the joystick would return to the set reference position upon release and thus would also stop the avatar robotic arm from moving.

3.5.2 Deadzone implementation

In a regular teleoperation session, when the user wants to move the avatar robotic arm by using a joystick, they have to push against the center-sprung mechanism. This mechanism is designed to automatically bring the joystick back to its starting point when the user lets go. However, while the user holds onto the joystick, the center-sprung mechanism only gives a subtle resistance. The user still has full control over how they move the joystick. If the user wants to change the direction in which the avatar robotic arm moves or if they need to make a decision about the movement, they can briefly bring the joystick back to its starting point. This will momentarily halt the movement of the avatar robotic arm.

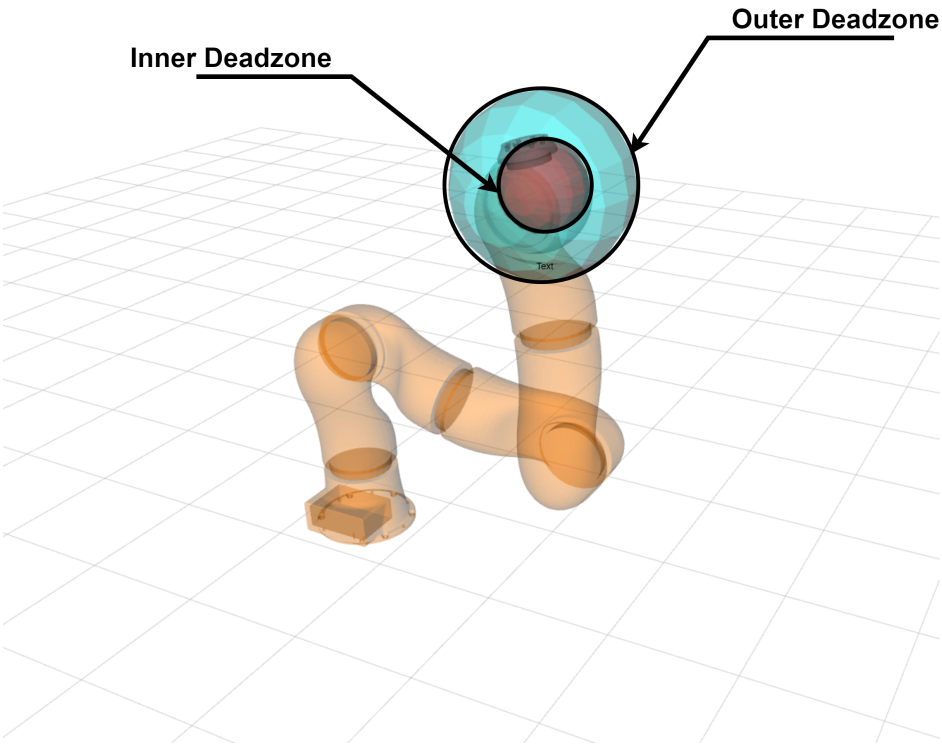

Figure 3.9: Dual Deadzone (Red sphere marks the inner dead zone and blue sphere marks the outer dead zone)

It's not always necessary for the user to perfectly align the joystick with the center point; a small area where the joystick can move without affecting the avatar robotic arm's motion is intentionally included to account for minor errors in positioning. This is called the inner deadzone and it is in the form of a virtual sphere at the reference position or zero position set on the haptic interface. The joystick will not send any teleoperation commands when it is within this deadzone sphere [HHC08]. There is also a second deadzone implemented that marks the outer limit of the haptic user interface. When the user crosses this limit, the teleoperation signals are stopped from being sent to the avatar robotic arm. The outer deadzone is implemented to address two constraints. The first reason is to limit the overall workspace so that the system can be mounted and used in a smaller environment. The second purpose is to avoid a sudden surge in the velocity of the robotic arm avatar. This velocity is linked to how far the haptic interface (in this context, the joystick) is moved from its starting point, known as the reference position or zero position. Another behavior seen from users during collisions is the inclination to swiftly pull the joystick in the opposite direction of the collision. However, this can lead to unregulated or sudden motion in the opposite direction. The outer deadzone acts as a max limit for such motions and helps in reducing the control commands triggered by joystick movements. The value of M is calculated based on the position of the haptic device in comparison to the red and blue zone. The center reference position is denoted by P_{hr} and the current position of the

haptic device is denoted by P_{hc}. Therefore the value of M is calculated as follows:

$$M = \begin{cases} 1, & \text{if } d >= R_r \text{ and } d <= R_b \\ 0, & \text{otherwise} \end{cases} \tag{3.18}$$

where d is the distance between points P_{hr} and P_{hc}

3.6 Vision system for goal and obstacle tracking

The vision subsection plays a crucial role in tracking both the goal and obstacles encountered during the robot's motion planning process. The goal is typically known in advance, as the user specifies it using the interface. This simplifies the task of tracking the goal, as the objects required to complete the task can be easily identified using markers, such as AprilTags. AprilTags are visual fiducial system that uses a 2D bar code style "tag", allowing full 6 DOF localization of features from a single image [Ols11].

On the other hand, obstacles can be categorized into two types: known obstacles and unknown obstacles. Known obstacles refer to objects that are already present in the robot's world representation but have unexpectedly moved and now obstruct the robot's planned trajectory. These known obstacles already have an identification marker attached to them. So to identify these known obstacles, marker-based identification techniques can be used, such as using AprilTags. Alternatively, a deep learning-based object classification method specifically trained on the representation of objects in the robot's world can be employed.

Unknown obstacles, on the other hand, are objects that were not present in the robot's initial world representation and are encountered for the first time. To address these obstacles, a generic deep learning-based object detection algorithm can be applied initially to classify and identify them. However, if the classification fails, a substitution method can be utilized, where the unknown obstacle is replaced with a primitive shape that approximates its size and shape. This allows the robot to consider the presence of the obstacle in its motion planning process, even if specific object details are unavailable.

The hardware employed in this methodology is the Intel RealSense camera, which provides depth information and facilitates the implementation of object classification and tracking tasks. The RealSense camera's capabilities, such as depth sensing and RGB imaging, allow for accurate perception of the environment, which is crucial for tracking the goal and identifying obstacles during motion planning.

By combining vision-based techniques, such as marker-based identification, deep-learning based object classification, and utilizing the Intel RealSense camera, the methodology enhances the robot's perception abilities compared to just having a marker-based identification system. This, in turn, enables effective tracking of the goal, identification of known and unknown obstacles, and supports the decision-making process in motion planning tasks. The main focus of the current thesis is the implementation of an assistive teleoperation framework. The vision system is a peripheral but important component. So the scope of the current thesis is limited to detecting the goal point using AprilTags and then simulating the obstacles virtually.

4 Implementation

The current system consists of four major sections, the operator robotic arm, the haptic interface, the avatar robotic arm, and the robotic gripper as depicted in Figure 4.1. The operator robotic arm is a standard KUKA Light Weight robot (LWR) and the avatar robotic arm is a modified KUKA Lightweight robot (LWR) with the last joint modified to structurally act as robot Justin's Arm. Both robotic arms have 7 DoF and can be either position-controlled or torque controlled. The haptic interface is called Exodex Adam and is a reconfigurable, dexterous, whole-hand haptic input device that is mounted atop the operator robotic arm. The Exodex haptic interface is capable of giving force feedback to three fingers and the palm section. This in turn is useful for in-hand manipulation utilizing the signals received from the robotic gripper. However the scope of the current thesis focuses on assisted teleoperation of the avatar robotic arm and not specifically any improvements in in-hand manipulation. During the testing and validation process, the entire system is simplified to an operator robotic arm, a joystick, and the avatar robotic arm.

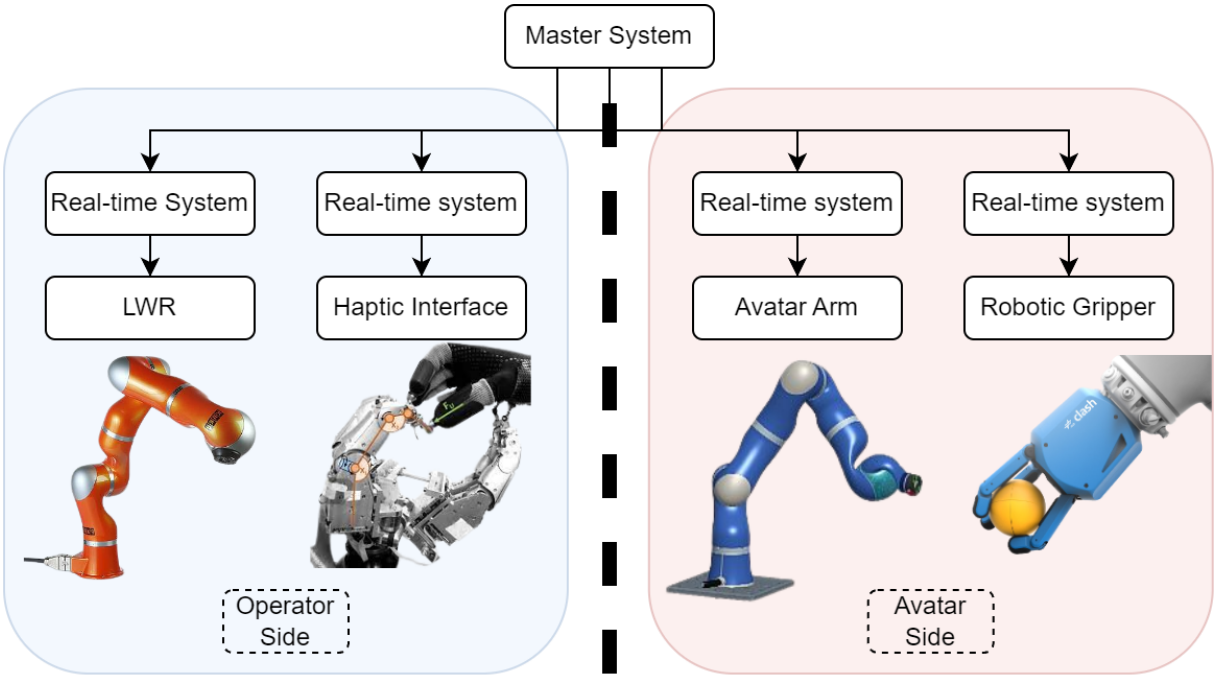

Figure 4.1: High level system depicting the teleoperation setup

4.1 Hardware and software overview

The two robotic systems are all connected to individual real-time computers. These computers are then connected to a single master computer. The master computer is used for building the code, running the processes, and monitoring the data during operation. The code pertaining to the control system of each specific robotic system is built using Simulink for the specific hardware and then deployed onto the real-time system. Two

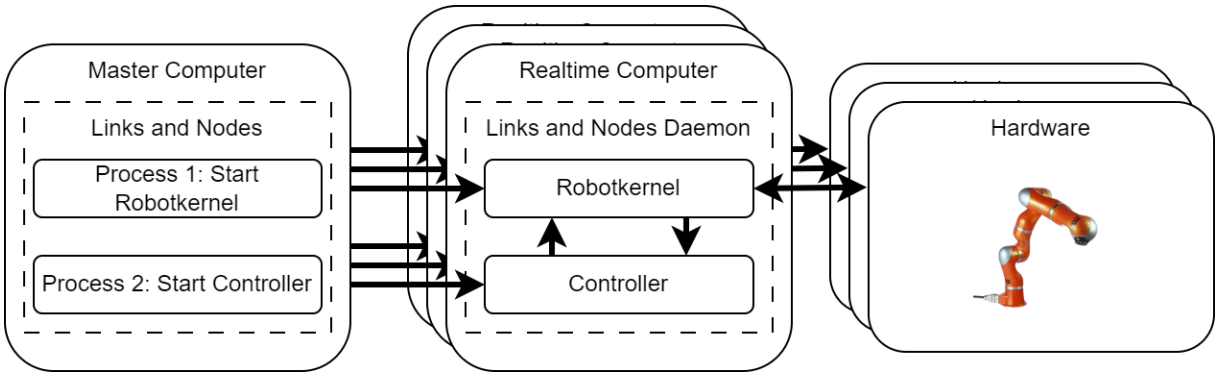

Figure 4.2: Process Handling for the robotic system

cross-platform solutions handle the remote execution of processes on all the different robotics hardware systems:

1. Links and Nodes (LN): A process management and real-time capable communication middleware for distributed systems.

2. Robotkernel (Rk): A framework to configure, manage and interconnect driver modules via human-readable config files.

The generic procedure utilized to configure and deploy the code is as follows:

1. Configuring and deploying the Rk on the real-time system

2. Building the code using Simulink on Master Computer and deploying it on the real-time system

3. Adding the processes on links and nodes manager on the Master computer to start the Rk and the executable code on the real-time system

4. Adding additional processes on the links and nodes manager for the simulation setup on the master system

Figure 4.2 shows an overview of how processes are distributed over the master computer and the real-time computers of each hardware. The links and nodes manager on the master system has the necessary processes to start the Rk and the controller code on

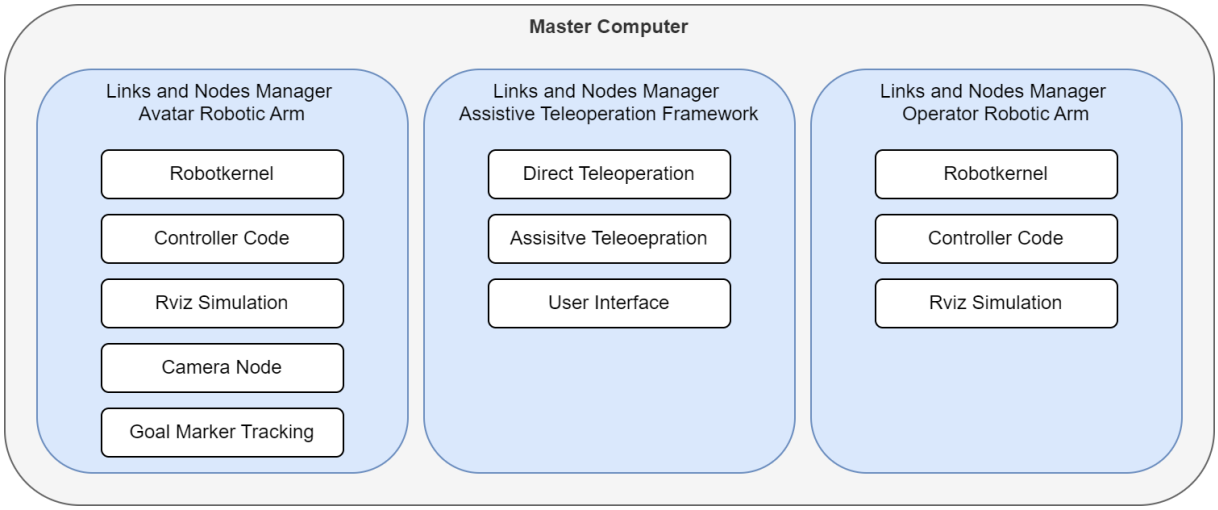

Figure 4.3: Links and Nodes manager and processes for the current thesis implementation on Master Computer

the real-time system. Figure 4.2 shows a generic setup of two major processes being initiated to control the hardware but in practice, the master system links and nodes manager can have multiple other processes listed. Since the links and nodes manager on the master system obtains data from the hardware through the real-time system, this data is subsequently utilized by various processes established on the master computer. These processes either make determinations for interconnected hardware or employ the data for configuring simulations. The master system also has the capacity to potentially operate multiple links and nodes managers, enabling them to communicate with other managers in order to transmit necessary data. The current thesis setup consists of three major links and nodes manager for the operator arm, avatar arm, and the assistive force calculation framework as shown in Figure 4.3. There are other two additional processes for the Exodex haptic interface and the robotic gripper respectively, which can be added at later stages.

4.2 Assisitve teleoperation implementation

The assistive teleoperation framework subscribes to the current goal position and the obstacles from the vision system as described in Section 3.6. It calculates the trajectory which acts as the reference for the mid-level and low-level planner. The user interface displays the calculated trajectory on the live video feed for the user to make a decision. Once the user activates the assistive force based on the trajectory overlay on the user interface, the trajectory is converted into appropriate force vectors and passed to the operator robotic arm. This force guides the user towards the goal position and helps in completing the task. While the global trajectory is being executed, the mid-level planner

and the low-level planner are active and searching for any deviation from the known obstacles or the arrival of additional obstacles. If any such addition or deviation occurs in the immediate vicinity of the avatar arm, the low-level planner applies appropriate torque on the joints of the robot while trying to maintain the end-effector position [Ott08]. In such a situation, as the end-effector is still maintaining the desired trajectory, the user will continue to receive the forces calculated by the method defined in Section 4.2. The mid-level planner keeps monitoring up to a particular horizon in the future to check for potential collisions. If any potential obstacles are found on the path in the monitored horizon, the global path itself is modified rather than making last-minute decisions. This modified section of the global trajectory is then processed using the steps defined in Section 3.4.1 and the set F defined in Equation 3.17 is modified.

The assistive guidance helps the user to follow the calculated trajectory, but if the user notices any miscalculation or potential collisions that went unnoticed by the three-tier planner, the user has full control to intervene and correct the trajectory. As soon as the user deviates from the calculated path, the teleoperation switches from assistive teleoperation to direct teleoperation due to the teleoperation strategy mentioned in Section 3.4. After this deviation from the suggested trajectory occurs, if the user brings the avatar robotic arm within a particular distance of any point on the trajectory, called influence distance, the user feels an assistive force that tries to merge the user on the pre-calculated trajectory. The user can opt to merge by following the force or continue as per the operator's judgment by pulling out of the influence distance. At any point, the user can completely scrap the assistive force and re-engage the assistance when required by using the start and stop buttons on the user interface. The re-engaged assistive force will be an entirely new set of calculated forces as per the current robot position and the goal position. In the current implementation, the goal point is solely calculated based on the detection of April tags, which hold the id that is linked to a particular task. As described in Section 3.6, the task information, and additional visual cues are overlayed on the live video feed to help the user make a clear decision about the usage of the assistive force feedback. In some cases, if the global motion planner fails to calculate a correct path to the goal, the user can always re-calculate the trajectory by entering the direct teleoperation mode on the user interface and moving the arm to a more appropriate position.

The assistive teleoperation framework has two major components, the teleoperation strategy as mentioned in Section 3.4 and the planning section mentioned in Section 3.1.5 to calculate the end effector coordinates along with the assistive forces. The teleoperation strategy is implemented as processes listed under the Assistive teleoperation framework links and nodes manager depicted in Figure 4.3. Links and nodes operate using the publish-subscribe protocol. The different processes publish relevant data to a particular

topic for other processes to use and also subscribe to topics published by other processes gaining access to their data. The processes under the assistive teleoperation framework subscribe to the values K, M, the pose increment data from the operator robotic arm (defined in section 3.4), and the telemetry data from the avatar arm along with the obstacle and goal position from the vision system. It publishes the force vectors to the operator arm and the trajectory coordinates to the avatar arm. The data utilized and generated by the assistive teleoperation framework is described in 4.1. The column titled "Mode" defines whether the data is subscribed to or published by the Assistive Teleoperation Framework. The columns titled "Operator robotic arm", "Avatar robotic arm", and "Vision" list the data published by each system respectively and then utilized by assistive teleoperation framework if the mode is set to subscribe and vice versa. The global trajec-

Table 4.1: Published and Subscribed Data by the Assistive Teleoperation Framework

Mode	Operator robotic arm	Avatar robotic arm	Vision
Subscribe	K	Joint pose	Obstacle Pose
Subscribe	M	-	Goal Pose
Subscribe	Pose Increment	-	-
Publish	Assistive Force	Trajectory	Trajectory

tory planner requires the current pose of the avatar arm, the goal pose and the obstacle pose to calculate the initial reference trajectory. Once the trajectory is generated, it is published to the vision system, which overlays it on the live video feed. The trajectory is then used to calculate the assistive guiding forces which are published to the operator side. The framework then monitors the value of K, M, and the pose increment which roughly estimates the user's intended motion based on the assistive guiding forces that were transmitted to the operator robotic arm. Once the framework receives the gains it decides whether to pass the pose increments or the next trajectory waypoint to the avatar arm. If the waypoint is published to the avatar arm, the framework waits for the commands to be executed, then looks up the next force vector to be passed to the operator side and waits for K, M, and the pose increment to enter the next cycle. If the user intents to deviate from the proposed path, the pose increment is passed to the avatar arm instead of the trajectory waypoint. During the next cycle, the force vector that is passed to the operator side tries to get the user back on the trajectory. It continues to apply this force till the avatar arm is back on the trajectory. Once it is back on the trajectory, the entire system falls back to assisting the user to follow the path.

While the trajectory is being executed the framework continuously monitors the goal pose and obstacles. If there are any new obstacles in the vicinity of the avatar arm, the ESF-based algorithm is used to calculate an additional force vector which is converted to joint

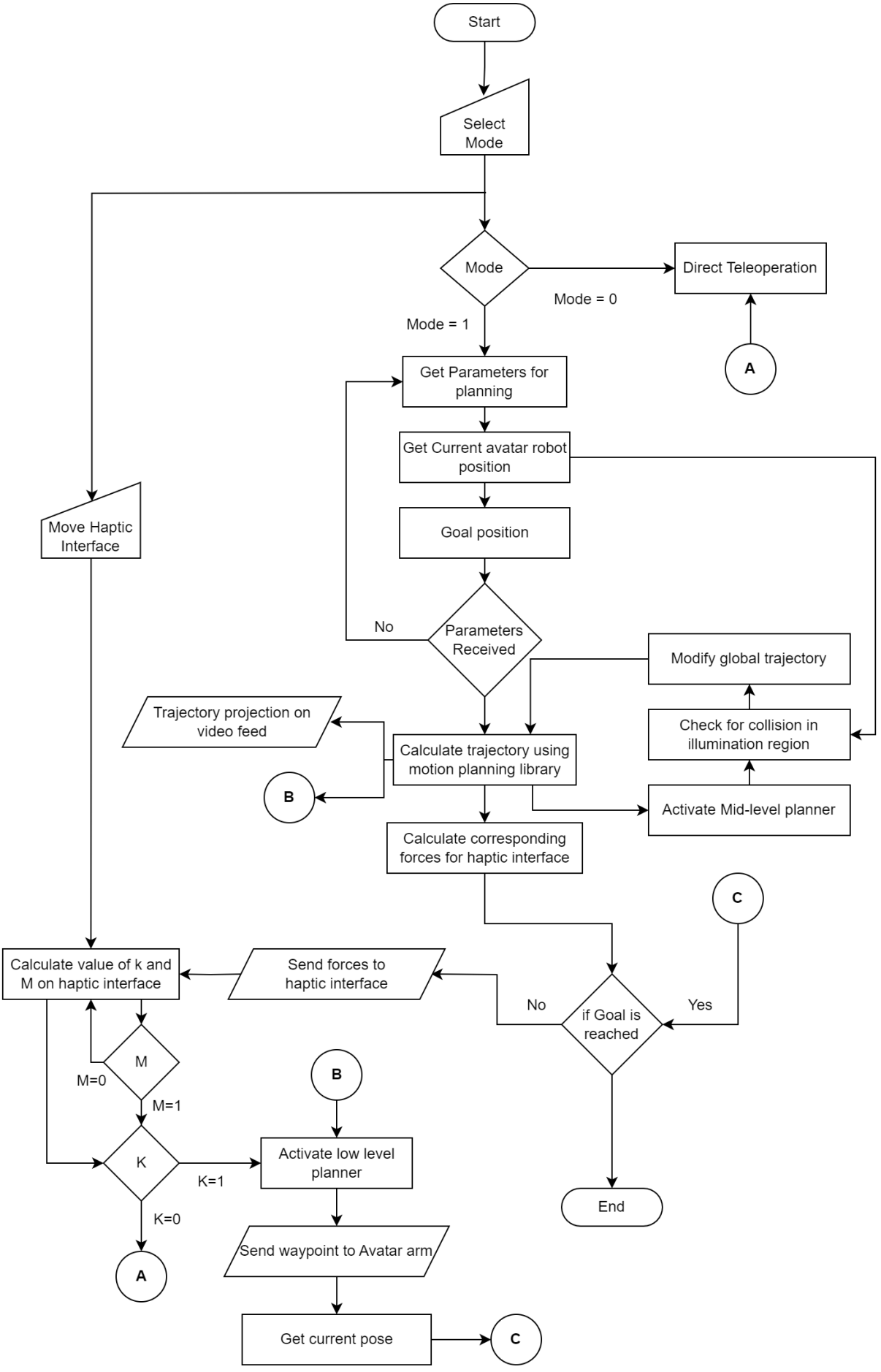

Figure 4.4: Assisitve teleoperation framework

torques and given to the avatar arm. The overview of the entire assistive teleoperation program is depicted in Figure 4.4

4.3 Goal tracking implementation

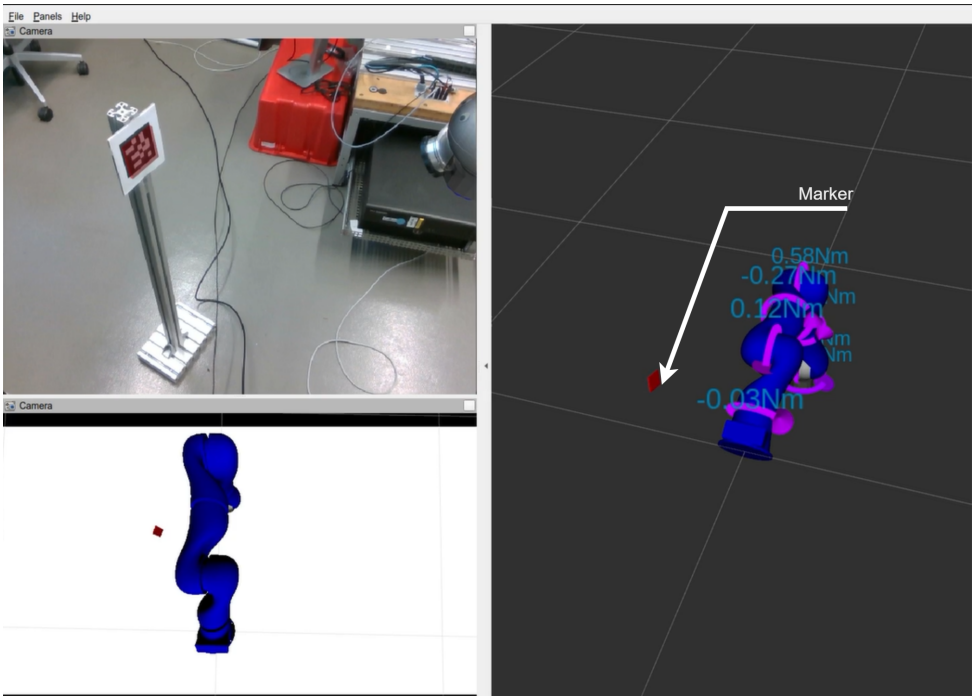

Figure 4.5: Goal tracking and visualization (goal is highlighted by red colored marker)

The goal pose and task to be executed are encoded using AprilTag. The implementation of marker detection and pose estimation is implemented in the Python language using opencv library. This marker detection section of the code is executed on the master computer itself (contrary to the real-time system) using the links and nodes manager for the avatar robotic arm as shown in Figure 4.3. The links and nodes process detects the goal marker from the camera feed obtained by subscribing to the ROS-based image topic published by the camera node specified in Figure 4.3. Once the marker is detected, the pose and orientation data is retrieved, and then published as a Rviz marker and also published on the topic "goal" in the links and nodes network. The published marker is then visualized in the Rviz simulation as shown in Figure 4.5 while the 'goal' topic is subscribed by the assistive teleoperation framework to plan the trajectory. Once the trajectory is planned, the trajectory is overlayed on top of the video feed as shown in Figure 4.6. This overlay acts as an additional visual aid for the operator. The user interface is mainly utilized to switch between direct teleoperation and assistive teleoperation. These appear as four buttons next to the video feed and simulation window for starting and stopping the direct and assistive teleoperation respectively as shown in Figure 4.7. The direct teleoperation

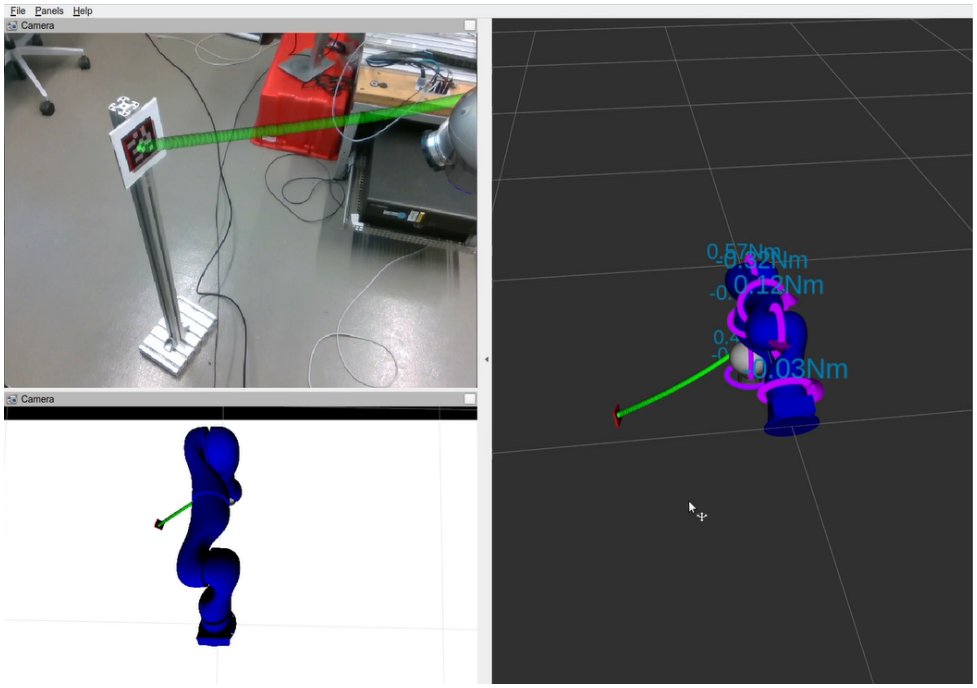

Figure 4.6: Overlayed trajectory on the video feed (set of green waypoints depict the calculated trajectory)

button starts the position-velocity-based direct teleoperation, mentioned in Section 2.1. The assistive teleoperation button starts the following procedure once clicked:

1. Get the current robot position and calculate the end effector pose.
2. Get the goal pose and orientation from the 'Goal Marker Tracking' process from the links and nodes manager for the avatar robotic arm.
3. Start the motion planning server and calculate the trajectory from the current end effect position to the goal marker pose as per the task identified by the marker id.
4. Visualize the trajectory on the video feed and Rviz simulation as depicted in Figure 4.6.
5. Wait for the user to make movement on the haptic device on the operator robotic arm.

4.4 User interface

Once the assistive teleoperation start button is pressed, the user can feel the assistive guiding force on the haptic device. The user can follow the assistive guiding force and complete the task. Once the goal is reached the mode switches to direct teleoperation by default. In case the user wants to stop assistive guidance mid-trajectory, the user can click

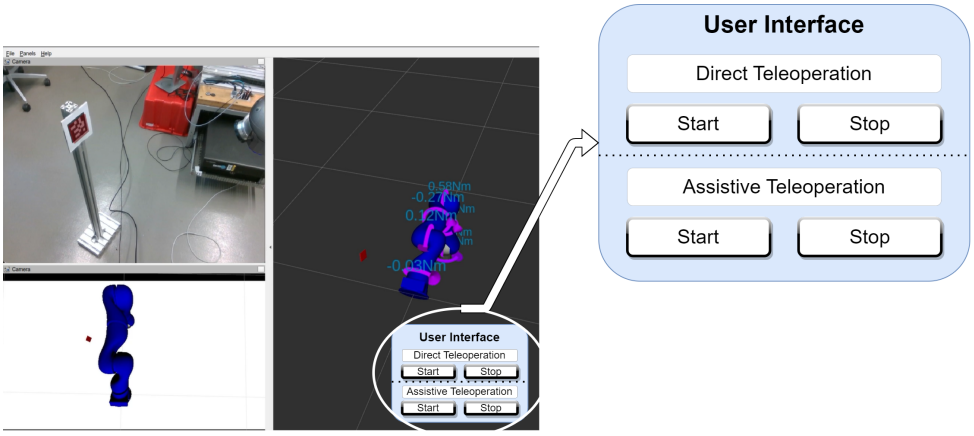

Figure 4.7: User Interface showing respective buttons for direct and assistive teleoperation

on the stop button under the assistive teleoperation section. If the trajectory generation fails for some reason, the user can click on the start button under the direct teleoperation section and move the robot to a more suitable position and then click on start assistive teleoperation.

5 Results and Discussion

The methodology and implementation in the current document are geared towards aiding the operator to efficiently undertake the teleoperation task. The assistive teleoperation framework that is developed and implemented in this thesis provides a visual presentation of the trajectory overlayed on the video feed, an assistive guiding force on the haptic device, and a simplified user interface to start and stop the assistive teleoperation framework. The assistive teleoperation framework reduces the cognitive load on the operator and gives the operator a better understanding of the trajectory to be followed.

The advantages of the assistive teleoperation framework are put to test in an evaluation test setup consisting of five participants. The participants are given three variations of the same teleoperation task to be completed. The variations include

- Direct Teleoperation : A direct teleoperation task
- Direct Teleoperation with trajectory overlayed: A direct teleoperation task with the trajectory overlayed on the video feed
- Assistive Teleoperation: A teleoperation task with the trajectory overlayed on the video feed along with assistive force feedback

The tasks are then compared using a subjective load assessment method - NASA Task Load Index (TLX) along with the use of some additional subjective feedback that is given by the user. This is then combined with the analysis of some recorded parameters like the time taken to complete the task to give an objective dimension to the validation study.

5.1 Evaluation

The NASA TLX assessment is a subjective tool to measure the mental workload of the user. In the current context, this method is used to evaluate if the assistive framework is leading to additional mental workload while in an attempt to increase manipulation performance. The NASA TLX gives a score between 0 to 100 for each specified task. A lower score on the NASA TLX assessment indicates a lower mental workload. The NASA TLX assessment also provides a pairwise combination of the parameters in consideration and asks the participant to select the parameter that has more effect on the workload. This helps in mitigating the differences in the parameter that contribute to the workload as perceived by the users. Following are the six parameters assessed in evaluating each task using NASA TLX as described by NASA[Har]:

- Mental Demand - How much mental and perceptual activity was required? Was the task easy or demanding, simple or complex?
- Physical Demand - How much physical activity was required? Was the task easy or demanding, slack or strenuous?
- Temporal Demand - How much time pressure did you feel due to the pace at which the tasks or task elements occurred? Was the pace slow or rapid?
- Own Performance - How successful were you in performing the task? How satisfied were you with your performance?
- Effort - How hard did you have to work (mentally and physically) to accomplish your level of performance?
- Frustration Level - How irritated, stressed, and annoyed versus content, relaxed, and complacent did you feel during the task?

The task is to reach a goal pose using the avatar robotic arm via three modes of teleoperation. The goal point is marked using an AprilTag and tracked and highlighted on the video feed as shown in Figure 4.5. The participants tried to complete the task using only the video feed from the robot and the following were the results.

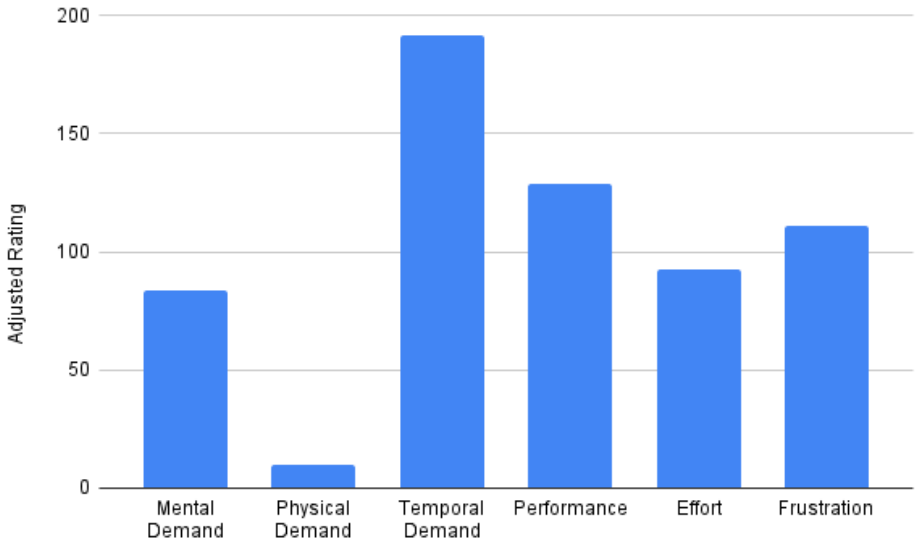

Figure 5.1: Direct Teleoperation

During direct teleoperation, the participants on average felt difficulty in estimating the distance from the end effector reference frame of the avatar robotic arm to the goal point using just the video feed. This led to additional mental load and also lead to high completion time. This eventually contributed to the frustration level and effort as depicted

in Figure 5.1. The second task included the estimated trajectory towards the goal point

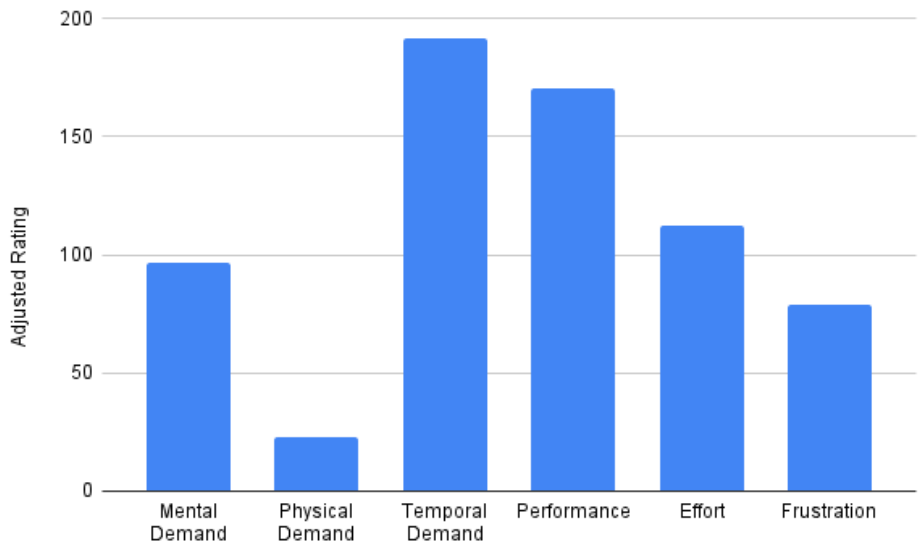

Figure 5.2: Direct Teleoperation with trajectory overlayed

overlayed on the camera feed. This helped in getting a better understanding of the depth of the video feed and led to reducing the completion time. But one user, in particular, felt that the projected path was hindering their ability to complete the task. On average, the overall performance was better than the first task as projected from Figure 5.2. The frustration on average was lower compared to the direct teleoperation task, but slightly more effort was required to carefully follow the trajectory, which inherently also added to a comparatively higher mental demand. This led to an overall weighted rating of 44.75 as per NASA TLX which is higher than the rating observed for the direct teleoperation task shown in Table 5.1. The final experiment was set up to accomplish the same task but with the trajectory overlayed on the video feed along with assistive force on the haptic device. This led to the shortest possible time to reach the goal as shown in Figure 5.4 along with lower frustration, mental demand, and effort compared to the previous two cases. The NASA TLX weighted rating for the assistive teleoperation task is 24.08 which is the lowest compared to the other two task setups.The individually assessed parameter values for the assistive teleoperation task are shown in Figure 5.3

Table 5.1: NASA Task Load Index weighted rating

Task	Weighted Rating as per NASA TLX
Direct teleoperation	41.17
Direct teleoperation with trajectory overlayed	44.75
Assistive Teleoperation	24.08

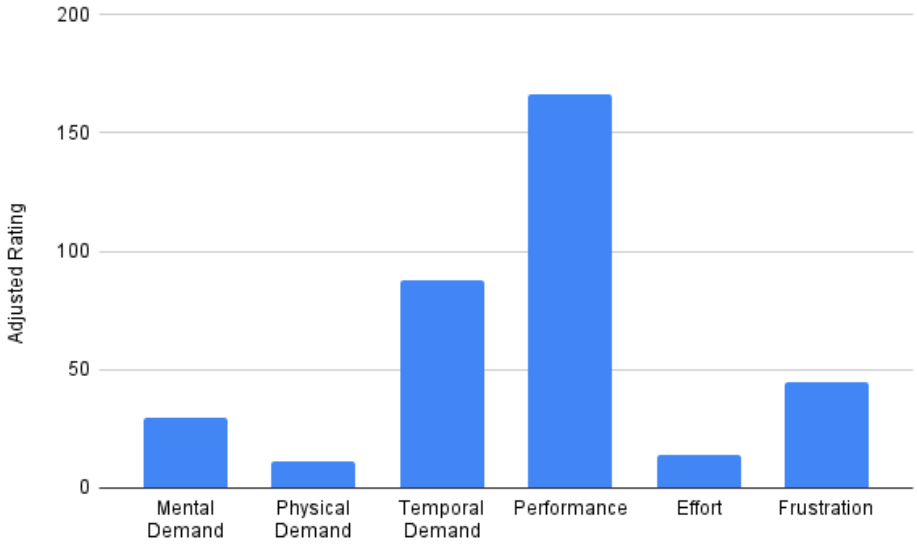

Figure 5.3: Assisitve Teleoperation

The completion time of the three tasks has decreased with the addition of a visual trajectory overlay by 14.78 percent. It has further increased by 47.44 percent with the addition of the assistive force on the haptic device. The time of completion for each of the three evaluation tasks is compared in Figure 5.4

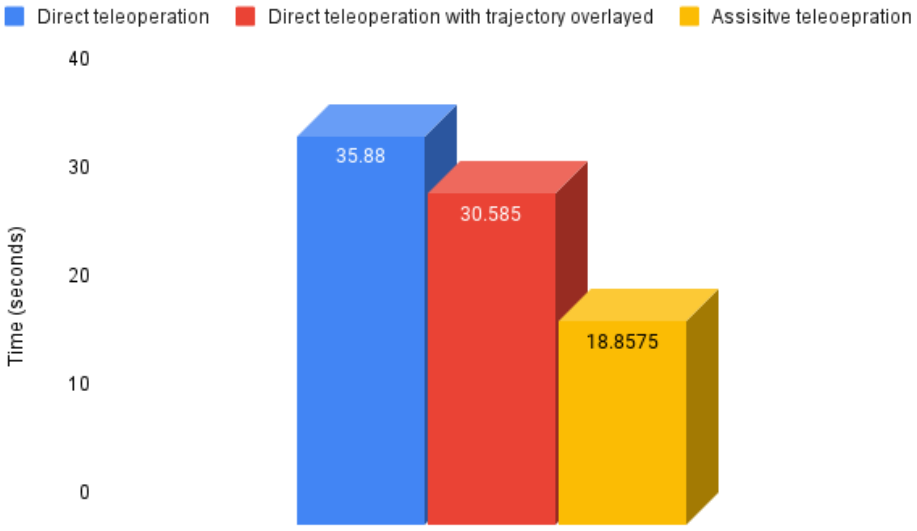

Figure 5.4: Time of completion

5.2 Discussion

The following section covers general discussion and feedback from participants, along with the learnings and errors encountered during the implementation phase of the thesis. The three types of tasks defined within the evaluation study gave a perfect opportunity to access the effect of assistive force. The direct teleoperation task in general took more time to complete. The lack of depth perception in the teleoperation task was constant feedback from all the participants. This reconfirms the problem statement that formed the basis for this thesis. Along with these data, there were other subjective questions that were to evaluate other factors in the assistive teleoperation framework. The collected data and the assessment procedure clearly indicate that the assistive guiding force helps in increasing the efficiency of the teleoperation task, but there were certain improvements that were suggested. The assistive guiding force was also accompanied by a projection of the desired trajectory on the video feed. When the projected trajectory was in combination with the assistive force feedback, the participants found it easier to navigate and reach the endpoint. But during the second version of the task, only the projected trajectory was displayed on the video feed but with no assistive guiding force. During the second task version, the reviews were mixed. Some participants found it helpful to have a projected trajectory on the video feed but some participants found it hindering their capability to navigate the robot.

Another common feedback was the lack of a proper task definition during the trials. This was an inherent drawback of the way in which the task was defined. The end goal was to reach within a particular radius of a cartesian pose and this induced lack of a definite evaluation criteria for the user as to when to stop the consider the task as done. The user studies in the future could include a particular task like a peg-in-hole test setup to evaluate the effectiveness.

The next potential flaw was observed during the trial runs of certain participants. Once the assistive guiding force was engaged, the user found it easy to follow the force feedback and navigate the robot. But once the robot reached the goal point, the force feedback abruptly stops and the users continued to hold the haptic interface at the last left position for a few more seconds before realizing that the goal point has reached. As per the framework, the robot switches to direct teleoperation mode after the goal point is reached and as the user held the haptic interface in a deflected position, the robot continued moving beyond the estimated goal point. The center sprung mechanism gives a subtle force to center the haptic interface and is only designed to center the haptic interface once the user lets go. This was a framework design consideration that was previously unknown but will definitely help in tuning the system further. The potential solution would be to

strengthen the center sprung mechanism when the robot reaches the endpoint and centers the haptic interface before switching to direct teleoperation mode.

One major section of the assistive teleoperation framework is regarding the resolution of the force feedback on the haptic device. If the force feedback changes direction and magnitude too often, the participants find it indistinguishable. The smoothing of the force feedback helps in reducing the problem but will affect the minute precision control of the avatar robotic arm. A balance should be maintained for the smoothing of the assistive force to provide human-distinguishable guidance and comparable precision control. The reaction times of humans differ from person to person, similarly, the frequency at which the change in force magnitude and direction can be detected might vary from person to person. Currently, the threshold is set using a trial-and-error method and is not fine-tuned for each user. Additional research and tests could be conducted to develop and evaluate a methodology to tune the threshold value for each user. Further alternative methods to manage the force feedback as a whole may also be considered. Together, with the field of augmented and virtual reality, this research should find synergy to improve the immersive user experience and assistive functionality.

The three-tier planning system implemented during the thesis helps navigate static and dynamic obstacles during the assistive teleoperation. The avatar robotic arm tries to navigate around the obstacles and parallelly tries to reach the goal by following the trajectory. But there are two particular instances that need specialized decision-making capability which is currently missing. There might be particular scenarios where the robotic arm might not be able to navigate toward the goal. This situation might arise when multiple obstacles are influencing the robot's motion during the teleoperation. The three-tier motion planning algorithm tries to avoid this simultaneous effect by taking care of obstacles away from the robot but on the trajectory before the robot even reaches near them. The global trajectory is modified to a certain extent beforehand and then further changed by the local planner. But the local planner now takes care of only the imminent collision. So the three-tier planner roughly isolates the obstacles on a temporal scale whenever possible and deals with them at different points in time. But there are still cases where sequentially dealing with the obstacles might not be possible. One more layer of the fallback plan would include considering the additional characteristics of the obstacles than those considered currently. The pose and the size of the obstacles are the only parameters that are currently considered, but the trajectory of the obstacles, the actual shapes of the obstacles, and the identification of the obstacles would help make additional navigation decisions. For example, it is just necessary to pause the robot for a moment if the obstacle is on a trajectory going outside the current workspace. This will save some computation and time to calculate and execute the avoidance maneuvers.

The current implementation of assistive teleoperation can be considered a bilateral teleoperation system as the user receives assistive guidance in the form of force feedback. But force reflection forms an important part of the bilateral teleoperation system. Force reflection incorporates the interaction forces between the slave robot and the environment which is then transmitted to the master haptic interface. Force reflection helps in completing tasks that rely heavily on contact with the environment. But force reflection integration is much more simple and straightforward when it comes to a position-position based teleoperation system as slave motion and/or force can be mimicked at the master haptic interface. But the current thesis implements a position velocity-based teleoperation system. And based on the current implementation, using the avatar robotic arm's position-based error to generate force reflection on the master haptic interface seems infeasible. But a change in the position of the end effector can be used to decide if there is restricting force acting on it. This information can then be utilized for force reflection on the haptic interface. But there might be a false trigger situation in which the avatar robotic arm has reached the end of its workspace and no further motion is possible. This might reflect as interaction forces on the haptic interface. Another potential solution would be to incorporate a force torque sensor at the end effector of the avatar robotic arm and use the readings from the sensor to generate appropriate force reflection on the haptic interface.

The assistive teleoperation framework currently tries to avoid collision and guide the user toward the goal. Collision avoidance maneuvers are achieved by keeping track of all the static and moving obstacles in the vicinity. This is normally done using a depth camera mounted on the robot. But there might be obstacles that close in towards the robot from regions that are not in the current field of view of the camera. These obstacles might not be tracked and accounted for during the generation of the assistive guiding force. So eventually the moving obstacles might collide with the robot and as the robot is in compliant mode, the obstacles might exert force on the robot making the robot move in the corresponding direction. There are two major problems arising out of this situation, one being the collision itself and there should be a methodology designed to avoid this. The second problem is that in case the collision is not avoided, it might exert additional forces on the robot and cause robot movement making it difficult to differentiate between the force reflected due to the assistive guidance force and the force reflected due to the movement initiated by the collision.

6 Conclusion

The primary objective of this thesis was to devise an assistive teleoperation framework capable of supporting operators engaged in teleoperation tasks. These operators often possess limited sensory input compared to the sensory capabilities of the robotic avatar system. The framework's purpose is to offer appropriate guidance based on sensor data obtained from the robotic avatar system.

The central feature of the implemented assistive teleoperation framework is the integration of assistive guiding forces transmitted to a haptic device. These forces facilitate the user's movement along an estimated optimal trajectory required to fulfill tasks on the avatar's side of the system. The optimal trajectory is computed through an enhanced version of the rapidly exploring random tree algorithm. This algorithm factors in both static environmental obstacles and the robot's physical structure. Enhancements were also made by introducing a decision-making layer to dynamically adjust the global trajectory in real-time, thus accounting for new static and dynamic obstacles. The assistive guiding forces, guiding the user, are derived from this dynamically adjusted trajectory, with variations in direction and magnitude smoothed out. This smoothing process compensates for the user's reduced perception of rapid changes in direction and magnitude.

In addition to the successful framework implementation, a user-friendly interface was developed to manage the activation and deactivation of assistive guiding forces. To offer users better insight into the task, the generated trajectory was superimposed onto the live video feed. This visual reference augmented teleoperation efficiency by providing a clear trajectory guide.

The assistive teleoperation framework notably enhances user efficiency in maneuvering the robotic arm to precise cartesian poses as dictated by task requirements. It is important to note that the ultimate manipulation decision remains with the user, who can choose to heed or disregard the guiding forces based on their judgment. The evaluation study shown in Chapter 5 shows that the framework successfully achieves its goal of supporting users in remotely teleoperating a robot despite limited sensory feedback.

While the framework was designed with a focus on aiding space telerobotics, its applicability extends to any teleoperation task involving a robotic arm. The framework's design emphasizes hardware independence, making it versatile and adaptable to various avatar robotic systems and haptic devices. Potential areas for future research could encompass integrating orientation planning alongside pose adjustments. By combining these aspects, assistive guiding forces could seamlessly steer users toward both the desired orientation

and pose. Presently, task selection hinges on marker detection and identification; however, refining this process to operate marker-free could enhance overall performance. Additionally, devising an optimal method for determining assistive force resolution would further fine-tune the system, enhancing its efficiency.

Prioritizing the seamless integration of assistive technology within teleoperation-based systems remains paramount. It is essential to ensure that assistance does not lead to an undue cognitive burden, as the ultimate aim of assistive guidance could be compromised. An effective technology should operate inconspicuously while significantly enhancing efficiency.

7 Outlook

In conclusion, the current assistive teleoperation framework has on average improved the efficiency and performance of the user in completing the defined task. The current implementation mainly focuses on assisting the user to achieve the required pose in cartesian space via assistive guiding forces and trajectory visualization all while avoiding obstacles in the environment. It aids the user in decision making but the final decision still completely rests on the user. The current approach is useful in scenarios where a robotic arm is teleoperated on the basis of a live video feed from the avatar robotic system.

In order to further reduce the cognitive load on the user, the assistance can be further improved to be seamless. The activation and deactivation of the assistive forces should be handled by the framework rather than through explicit commands by the user. This switch from activation to deactivation can be handled based on the user's intent detection. Additional evaluation parameters can be introduced to measure the accuracy of the assistive guidance, which can be then used as a weight to decide whether to use the guidance or re-calculate it.

The current implementation fully relies on traditional control and rule-based methods. The latest Machine learning-based methods are capable of solving more complex manipulation problems. Rule-based methods display high performance and reliability in well-defined scenarios but lack flexibility when it comes to unseen scenarios. Further methodologies can be explored to effortlessly switch between rule-based methods and Machine learning-based methods to achieve a particular task with the desired accuracy. These methods will be successfully able to leverage the reliability of the rule-based methods and the adaptability of the machine learning based methods.

Another major segment is the capability to distinguish between the assistive guiding force, and forces due to collision. The current implementation is capable of handling both types of forces and with additional visual feedback via a live video feed, it becomes easier for the user the distinguish the two mentioned types. But the actual problem arises when the part of the robot colliding is not in the camera frame and remains unknown to the user. Further research to distinguish collision from guiding forces would help enhance the overall system. A similar problem arises when the task itself involves introducing intended and unintended physical contact with an object/environment. In this case, even if the process is visible on screen, it becomes difficult for the user to determine if the applied force is as per requirement or if there is a sudden increase in force due to other unforeseen reasons.

Bibliography

[AKR13] Agarwal, P. et al.
Formalizing Assistive Teleoperation
In: Robotics: Science and Systems VIII, 2013, pp. 73–80.

[AMO07] Abbott, J. J.; Marayong, P.; Okamura, A. M.
Haptic virtual fixtures for robot-assisted manipulation
In: pp. 49–64.

[BAH13] Boessenkool, H.; Abbink, D. A.; Heemskerk, C. J. M.; Helm, F. C. T. van der; Wildenbeest, J. G. W.
A task-specific analysis of the benefit of haptic shared control during telemanipulation
In: IEEE Transactions on Haptics, 6 (June 2013) 1, pp. 2–12.

[BK02] Brock, O.; Khatib, O.
Elastic strips: A framework for motion generation in human environments
In: The International Journal of Robotics Research, 21 (2002) 12, DOI 10.1177/027836490202101200 2, pp. 1031–1052.

[BQH21] Bustamante, S.; Quere, G.; Hagmann, K.; Wu, X.; Schmaus, P.; Vogel, J.; Stulp, F.; Leidner, D.
Toward seamless transitions between shared control and supervised autonomy in Robotic Assistance
In: IEEE Robotics and Automation Letters, 6 (2021) 2, DOI 10.1109/lra.2021.3064449, pp. 3833–3840.

[Bur96] Burdea, G. C.
Force and touch feedback for virtual reality
Wiley, 1996.

[CG02] Crandall, J.; Goodrich, M.
Characterizing efficiency of human robot interaction: a case study of shared-control teleoperation
In: 1290–1295 vol.2.

[CGR22] Chen, S.; Gao, J.; Reddy, S.; Berseth, G.; Dragan, A. D.; Levine, S.
Asha: Assistive teleoperation via human-in-the-loop reinforcement learning
In: 2022 International Conference on Robotics and Automation (ICRA) (, 2022), DOI 10.1109/icra46639.2022.9812442.

[CHM21] Clever, H. M.; Handa, A.; Mazhar, H.; Parker, K.; Shapira, O.; Wan, Q.; Narang, Y.; Akinola, I.; Cakmak, M.; Fox, D.
Assistive Tele-op: Leveraging Transformers to Collect Robotic Task Demonstrations
arXiv: 2112.05129 [cs.RO].

[CSM15] Coleman, D.; Sucan, I. A.; Moll, M.; Okada, K.; Correll, N.
Experience-based planning with sparse ROADMAP SPANNERS
In: 2015 IEEE International Conference on Robotics and Automation (ICRA) (, 2015), DOI 10.1109/icra.2015.7139284.

[DS13] Dragan, A.; Srinivasa, S.
A policy blending formalism for shared control
In: International Journal of Robotics Research (, May 2013).

[ETD17] Ezeh, C.; Trautman, P.; Devigne, L.; Bureau, V.; Babel, M.; Carlson, T.
Probabilistic vs linear blending approaches to shared control for wheelchair driving
In: pp. 835–840.

[FBM] Ferre, M.; Barrio, J.; Melchiorri, C.; Bogado, J. M.; Castedo, P. L.; Ibarra, J. M.
Experimental results on bilateral control using an industrial telemanipulator
In: Springer Tracts in Advanced Robotics (), DOI 10.1007/978-3-540-71364-7_12, pp. 177–190.

[FR07] Farkhatdinov, I.; Ryu, J.-H.
Hybrid position-position and position-speed command strategy for the bilateral teleoperation of a mobile robot
In: pp. 2442–2447.

[FS67] Ferrell, W. R.; Sheridan, T. B.
Supervisory control of remote manipulation
In: IEEE Spectrum, 4 (1967) 10, DOI 10.1109/mspec.1967.5217126, pp. 81–88.

[GTT22] Gottardi, A.; Tortora, S.; Tosello, E.; Menegatti, E.
Shared Control in Robot Teleoperation With Improved Potential Fields
In: IEEE Transactions on Human-Machine Systems, 52 (2022) 3, DOI 10.1109/THMS.2022.3155716, pp. 410–422.

[Har] Hart, S. G.
https://ntrs.nasa.gov/citations/20000021488.

[HFB] Hirche, S.; Ferre, M.; Barrio, J.; Melchiorri, C.; Buss, M.
Bilateral control architectures for Telerobotics
In: Springer Tracts in Advanced Robotics (), DOI 10.1007/978-3-540-71364-7_11, pp. 163–176.

[HHC08] Hinterseer, P.; Hirche, S.; Chaudhuri, S.; Steinbach, E.; Buss, M.
Perception-Based Data Reduction and Transmission of Haptic Data in Telepresence and Teleaction Systems
In: IEEE Transactions on Signal Processing, 56 (2008) 2, DOI 10.1109/TSP.2007.906746, pp. 588–597.

[Hog85] Hogan, N.
Impedance Control: An approach to manipulation: Part I—theory
In: Journal of Dynamic Systems, Measurement, and Control, 107 (1985) 1, DOI 10.1115/1.3140702, pp. 1–7.

[HS06] Hokayem, P. F.; Spong, M. W.
Bilateral teleoperation: An historical survey
In: Automatica, 42 (2006) 12, DOI 10.1016/j.automatica.2006.06.027, pp. 2035–2057.

[JH92] Jones, L. A.; Hunter, I. W.
Human operator perception of mechanical variables and their effects on tracking performance
In: Advances in Robotics, 42 (1992), pp. 49–53.

[Justin] https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-11427/#gallery/35411.

[KAW11] Katzourakis, D.; Alirezaei, M.; Winter, J. C. de; Corno, M.; Happee, R.; Ghaffari, A.; Kazemi, R.
Shared control for road departure prevention
In: pp. 1037–1043.

[LA18] Lehner, P.; Albu-Schäffer, A.
The Repetition Roadmap for Repetitive Constrained Motion Planning
In: IEEE Robotics and Automation Letters, 3 (2018) 4, DOI 10.1109/LRA.2018.2856925, pp. 3884–3891.

[LaV06] LaValle, S. M.

In: Planning algorithms (, 2006), DOI 10.1017/cbo9780511546877.

[Lic07] Lichiardopol, S.
A Survey on Teleoperation
In:

[LK01] LaValle, S. M.; Kuffner, J. J.
Rapidly-exploring random trees: Progress and prospects
In: Algorithmic and Computational Robotics (, 2001), DOI 10.1201/9781439864135-43, pp. 303–307.

[NDV21] Nahri, S. N.; Du, S.; Van Wyk, B. J.
A review on haptic bilateral teleoperation systems
In: Journal of Intelligent amp;amp; Robotic Systems, 104 (2021) 1, DOI 10.1007/s10846-021-01523-x.

[Ols11] Olson, E.
AprilTag: A robust and flexible visual fiducial system
In: pp. 3400–3407.

[OPV19] Omarali, B.; Palermo, F.; Valle, M.; Poslad, S.; Althoefer, K.; Farkhatdinov, I.
Position and velocity control for Telemanipulation with Interoperability Protocol
In: Towards Autonomous Robotic Systems (, 2019), DOI 10.1007/978-3-030-23807-0_26, pp. 316–324.

[Ott08] Ott, C.
Cartesian impedance control of redundant and flexible-joint robots
In: Springer Tracts in Advanced Robotics (, 2008), DOI 10.1007/978-3-540-69255-3.

[PCT20] Parent, D.; Colomé, A.; Torras, C.
Variable Impedance Control in Cartesian Latent Space while Avoiding Obstacles in Null Space
In: pp. 9888–9894.

[PSK11] Park, S.; Seo, C.; Kim, J.-P.; Ryu, J.
Robustly stable rate-mode bilateral teleoperation using an energy-bounding approach
In: Mechatronics, 21 (2011) 1, DOI 10.1016/j.mechatronics.2010.10.002, pp. 176–184.

[PSM16] Perez-del Pulgar, C. J.; Smisek, J.; Muñoz, V. F.; Schiele, A.
Using learning from demonstration to generate real-time guidance for haptic shared control
In: pp. 3205–3210.

[QHI20] Quere, G.; Hagengruber, A.; Iskandar, M.; Bustamante, S.; Leidner, D.; Stulp, F.; Vogel, J.
Shared control templates for assistive robotics
In: 2020 IEEE International Conference on Robotics and Automation (ICRA) (, 2020), DOI 10.1109/icra40945.2020.9197041.

[SAP18] Shahbazi, M.; Atashzar, S. F.; Patel, R. V.
A Systematic Review of Multilateral Teleoperation Systems
In: IEEE Transactions on Haptics, 11 (2018) 3, DOI 10.1109/TOH.2018.2818134, pp. 338–356.

[SMS16] Saeidi, H.; McLane, F.; Sadrfaidpour, B.; Sand, E.; Fu, S.; Rodriguez, J.; Wagner, J.; Wang, Y.
Trust-based mixed-initiative teleoperation of mobile robots
In: pp. 6177–6182.

[TMG20] Triantafyllidis, E.; Mcgreavy, C.; Gu, J.; Li, Z.
Study of Multimodal Interfaces and the Improvements on Teleoperation
In: IEEE Access, 8 (2020), DOI 10.1109/ACCESS.2020.2990080, pp. 78213–78227.

[XLS23] Xia, X.; Li, T.; Sang, S.; Cheng, Y.; Ma, H.; Zhang, Q.; Yang, K.
Path planning for obstacle avoidance of robot arm based on improved potential field method
In: Sensors, 23 (2023) 7, DOI 10.3390/s23073754, p. 3754.

[ZHC18] Zeestraten, M. J.; Havoutis, I.; Calinon, S.
Programming by demonstration for shared control with an application in teleoperation
In: IEEE Robotics and Automation Letters, 3 (2018) 3, DOI 10.1109/lra.2018.2805105, pp. 1848–1855.

[ZS00] Zhu, W.-H.; Salcudean, S.
Stability guaranteed teleoperation: an adaptive motion/force control approach
In: IEEE Transactions on Automatic Control, 45 (2000) 11, DOI 10.1109/9.887620, pp. 1951–1969.

List of Tables

List of Figures