HARVARD MEDICAL SCHOOL AND
BRIGHAM AND WOMEN'S HOSPITAL

UCLouvain

FACULTE DES SCIENCES

Faculty of science

"Integrating Robotics and Virtual Simulation for Advanced Medical Procedures: A Focus on Prostate Biopsy Technology"

Thesis submitted in partial fulfillment of the requirements for the academic degree of
Master [120] in medical physics

Autor : Lorena Claeys Azurin
Advisors : Dr. Hilde Bosmans and Dr. Nobuhiko Hata
Lectors : Dr. Raymond Oyen and Dr. Frederik De Keyzer
School of Physics
Academic year: 2023-2024

Acknowledgement

I would like to thank SNRlab, especially Dr. Nobuhiko Hata, Dr. Junichi Tokuda, Dr. Mariana Bernardes, and Quinn Rainer, as well as their entire team for their help during my stay in their laboratory and for teaching me how to be independent in my research.

I also warmly thank Dr. Hilde Bosmans, firstly, for her immense kindness and very valuable support, and secondly for finding the time for me in her overflowing agenda.

Thank you to my readers, Dr. Frederik De Keyzer and Dr. Raymond Oyen. May you find my thesis both enjoyable and instructive.

I am also grateful to the professors who made an impact on my journey in medical physics, particularly Dr. Eduardo Cortina and Dr. Edmond Sterpin, who followed and advised me during these two and a half years, and who always took the time to answer my numerous emails and questions.

But my years in physics are not just about the academic part. During my five years in physics, I forged bonds with many physicist friends, especially my two inseparable companions, with whom I spent sleepless nights working on projects or understanding lectures, Thomas Stinglhamber and Colin Gaban. Thank you also to my parents that came all the way to the USA for one week. And last but not least, my partner Kamil Kon, who supported me in the difficult moments during the completion of my thesis and who corrected my English mistakes in this paper.

Contents

List of Figures

Chapter 1

Introduction

Prostate cancer, a common men's health problem, presents significant diagnostic challenges [Society, 2023]. Indeed, accurately conducting biopsies on small, suspicious lesions to ascertain whether they are malignant or benign is often difficult. To address these challenges, the field of medical robotics has made remarkable advances, revolutionizing the way diagnostic and therapeutic procedures are performed.

1.1 Overview of the current landscape of prostate cancer diagnosis and biopsy procedures.

Prostate cancer ranks as one the most prevalent form of cancer in males [Society, 2023]. It is characterized by the development of cancerous cells within the prostate gland. A screening is often performed to detect cancer, using a blood test known as the prostate-specific antigen (PSA) [Stephen W. et al., 2023]. PSA is a substance produced by the prostate gland. Elevated PSA levels in the blood can indicate the presence of prostate cancer [CDCBreastCancer, 2022]. It's important to note that PSA levels can also be elevated in other prostate-related conditions and diseases [Daniel et al., 2018]. While this approach is non-invasive, it cannot serve as a definitive diagnosis [Mareshal, 2022]. Another way for screening for prostate cancer is the Digital Rectal Examination (DRE) [Daniel et al., 2018]. It is performed by a medical professional inserting a lubricated, gloved finger into a man's rectum to detect any abnormalities in the prostate, such as cancer. However, the use of DRE as a screening test is not recommended due to insufficient evidence regarding its benefits [CDCBreastCancer, 2022]. Both of these methods are useful for screening purposes, but they cannot be used as a diagnostic method, unlike biopsy [Mareshal, 2022].

The most frequently employed method for conducting a prostate biopsy consist of taking tissue samples from multiple regions of the prostate under transrectal ultrasound (TRUS) guidance [Penzkofer et al., 2015]. The primary issue with this approach is the difficulty in distinguishing prostate tumors from regular glandular tissue based on ultrasound images [Mareshal, 2022]. Hence, there is an error rate of 20-30% in failing to detect a positive result [Dimmen et al., 2012]. Transperineal ultrasound is another method, opting to insert the needle for biopsy through the perineum rather than the rectum [Penzkofer et al., 2015]. The transperineal method improves precision in reaching the target but is still not fully effective in detecting

all prostate cancers [Chang et al., 2013].

1.1.1 Overview of the current landscape of targeted treatment

The current treatment options for prostate cancer, particularly prostatectomy and radiotherapy, are often considered invasive and heavy. Prostatectomy, which involves the removal of the prostate, can lead to complications such as urinary incontinence and erectile dysfunction, often temporary but sometimes persistent [Society, 2019]. Radiotherapy, aimed at eliminating cancer cells through high-energy rays, can cause intestinal disorders, diarrhea, rectal bleeding, bladder problems, and also erectile dysfunction [Kim et al., 2002].

These treatments are sometimes applied following a misdiagnosis or overdiagnosis. A study on ablated prostates indicated that a significant proportion of tested prostates actually contained no tumor [Özer and Hasbay, 2020]. This observation raises questions about the accuracy of diagnostics.

In this context, there is increasing interest in focal treatments, which target only the cancerous areas while preserving the surrounding healthy tissues. These methods offer a less invasive alternative, potentially with fewer side effects.

1.1.2 Towards the development of targeted biopsy

In order to minimize misdiagnosis or overdiagnosis and therefore avoid overtreatment, a new method has emerged: Targeted biopsy. This method allows for needle guidance and provides feedback through an MRI. MRI can detect smaller lesions that are not visible on ultrasound due to it's superior soft tissue contrast. An other advantage of the use of MRI for diagnosis is that calcifications do not interfere with image interpretation [Masoom et al., 2022] (mostly due to MRI's superior soft tissue contrast and multiparametric imaging capability [Sang-Eun Song et al., 2013]). MRI, used alone or with targeted biopsy, improved detection of significant prostate cancer by 57% and reduced biopsy cores per procedure by 77% compared to systematic biopsy [Clare M C and MD Peter R, 2023]. Additionally, 33% of men might have avoided biopsy based on negative multiparametric MRI results [Clare M C and MD Peter R, 2023]. Targeted biopsy can be performed in two different ways: in-bore or MR-US fusion [Masoom et al., 2022]. In the case of in-bore biopsy, similar to MRI-US fusion, a prebiopsy mpMRI (multiparametric MRI) employed to locate lesions. However, instead of using ultrasound, the biopsy procedure is guided by MRI imaging [Birtch, 2021]. MR/US fusion-guided biopsy relies on generating a 3D image of the prostate gland that provides information about the suspected tumor's location, size, and shape. This image is constructed using a prior MRI scan conducted before the biopsy procedure. During the procedure, this stored MRI image is imported into a dedicated device, aligning it with real-time ultrasound images. This allows the urologist to guide the biopsy tool accurately to sample the suspected tumor. It enables sampling of the prostate at a later time after the MRI session, with an accuracy of within 1–2 mm [Cochrane Miller,].

Although MRI-guided biopsy has demonstrated superior clinical outcomes compared to conventional TRUS-guided biopsy, accurate needle placement remains a technical challenge [Moreira et al., 2021]. Improper needle placement can have a significant impact on diagnostic sensitivity and specificity. Multiple needle insertions to compensate for lack of precision result in prolonged procedure times, patient discomfort, and potential risks, including increased tissue damage and prolonged procedure time. Therefore, it is necessary to reduce the number of insertion attempts while maintaining accuracy to improve the clinical outcomes of transpelvic prostate biopsy. To cope with these challenges, SNRLAB (Surgical Navigation and Robotics Lab) introduces the *Smart Model*, a three-degree-of-freedom (DoF) needle guidance controller designed to tilt the needle insertion path toward a given target [Sang-Eun Song et al., 2013]. The *smart model* is designed for in-bore MRI-guided interventions, focusing on improving accuracy without TRUS imaging guidance. Laboratory experiments and in vivo studies using animal models were conducted to evaluate the targeting accuracy of the device. However, the device has not yet been implemented in clinical settings due to the need for further refinement and validation to ensure its reliability and safety in human patients.

1.2 State-of-the-Art

1.2.1 Significance of virtual robotic simulation in the clinical field

In the clinical field, surgeons use virtual simulations to practice complex surgeries, or to chose for the optimal path during the preoperative phase, using virtual representations of real patient's data based on medical imaging.

Virtual practice for complex surgeries

An example illustrating practice for complex surgeries, is the virtual simulation used in training for laparoscopic surgery (Laparoscopy is an operation performed in the abdomen or pelvis using small incisions with the aid of a camera [lap, 2023]). This method provides a virtual environment in which surgeons can practice complex operations and procedures without putting the patient at risk. For example, in 2019, a team developed a Gazebo simulation for soft tissues, for among others the abdomen [Artur et al., 2019]. The plugin demonstrates realistic interaction with soft tissue and therefore allowed surgeons to practice.

Virtual path planning

In the preoperative planning stage of a prostate biopsy, surgeons sometimes face the challenge of determining the optimal surgical approach due to the lack of preoperative visualization methods for choosing the best biopsy path, avoiding important anatomical structures [Lakhal et al., 2023]. To find the optimal path in prostate biopsy, virtual simulation based on virtual reality or complex algorithms can be employed. For example, [Florin et al., 2015]'s work proves that virtual reality technologies allow to simulate robotic biopsy procedure and to generate accurate needle

trajectories that avoid vital organs. A virtual environment has been modelled and simulations for robotic-assisted biopsy of the prostate have been performed. Another example, this time based on algorithms instead of virtual reality is the work of [Lakhal et al., 2023]. The team developed an algorithm for path planning in prostate biopsy, based on combinatorial optimisation. The new approach was evaluated using CoBra, a 1.5T MR robot with 5 degrees of freedom, by conducting a biopsy on a live animal. The biopsy placement in the living tissue had a variation of approximately 3.8mm in a prostate measuring 26mm in diameter.

1.3 Goal of the thesis

This thesis aims to create a functional simulation platform for path planning of the 3 Degrees of freedom robot used for prostate biopsies in animal studies, developed by SNRLAB. This work employs technologies such as Robot Operating System (ROS), RVIZ, MoveIt, and Gazebo. Most path planning methods described in literature use complex software techniques or virtual reality to determine the optimal path during the biopsy procedure. The novelty of our research is the use of a simple tool that is built around 'MoveIt' for path planning in prostate biopsy. The ultimate aim of this virtual simulation is to assist in the planning phase before the surgical practice, creating a simulation based on the patient's anatomy. The advancement of having a 3D simulation of each patient's prostate and of the robot for prostate biopsy, along with detailed representations of surrounding anatomical features such as the bladder, would be a major step forward in medical precision. By accurately defining a target area within this virtual model, medical professionals could meticulously plan the trajectory of a needle for procedures like biopsies or treatments. A predictive path finding could ensure the needle navigates through the intricate landscape of internal structures, avoiding vital organs and tissues. This level of planning would not only enhance the accuracy of the procedure but also significantly reduces the risk of potential complications by minimizing any unintended contact with surrounding anatomical elements. Such technology would represent a leap forward in patient-specific treatment planning, offering a tailored approach that accounts for the unique internal landscape of each individual, thereby optimizing both safety and effectiveness of medical interventions. The creation of a virtual simulation of a robot for prostate biopsy fits into the wider aim of the SNRLAB (Harvard university project team) to improve diagnostics and treatments for prostate cancer. Pre-surgical or intersurgical simulations would decrease the risk involved in the procedure. The simulation would enable a finer adaptation to the anatomical specifics of each patient, thus optimizing the effectiveness of the diagnosis and treatment. The risk of overdiagnosis could be reduced, avoiding unnecessary treatments and their side effects.

In this master thesis project, a proof of concept platform was built that used data of a single patient and of which the output was visually checked in the simulation platform.

1.4 Structure

Chapter 1: Introduction

This chapter provides an overview of the thesis, the background and motivation for the thesis, highlights the challenges in prostate cancer diagnosis and the role of robotic assistance in medical procedures.

Chapter 2: Materials

This chapter describes the core tools used in the work: ROS (Robot Operating System), RVIZ, MoveIt, and Gazebo. It explains how these tools enable the creation, visualization, and testing of complex robotic systems, and discusses their role and integration in robot development, particularly in medical applications.

Chapter 3: Project Context

In this chapter, we expose the context of the project, explaining the prior work on witch this thesis relies on.

Chapter 4: Methodology

This chapter covers the methodological approach of the project, focusing on the modelling of the robot and the prostate using URDF and 3D Slicer. It also outlines the simulation process with Gazebo, the integration with ROS2 control, and the combination of MoveIt and RVIZ for planning and execution of paths to reach the target.

Chapter 5: Results

This chapter presents the results of the modelling and simulation process. This includes a comparative analysis of URDF and prostate models with real data, evaluates simulation results using Gazebo. It also discusses the integration of MoveIt and RVIZ, for path planning, chosing the optimal path to reach the prostate.

Chapter 6: Discussion

This chapter discusses the results and how they can be improved.

Chapter 7: New research opportunities

This chapter explores potential new research directions and improvements in the field of medical robots, specifically in relation to prostate biopsy robots. We discuss the application of the TF framework, needle insertion simulation, liver biopsy path planning, and training opportunities for medical staff using Gazebo simulation.

Chapter 7: Conclusion

The final chapter summarizes the most important results and contributions of this work. It reflects on the challenges and lessons learned and suggests directions for future research in the field of medical robotics, especially to avoid false negatives results for prostate biopsy and to improve autonomy of robots in various surgical procedures.

Chapter 2

Materials

In this section, we discover the core tools of thesis: ROS, RVIZ, MoveIt, and Gazebo. As explained in Section 3.1, this research contributes to a larger project, where ROS was already used as the primary interface for robotic interaction. The choice to incorporate RVIZ, MoveIt, and Gazebo was driven by complementary research, which revealed these tools as optimal for our objectives. Their compatibility with ROS, cost-free availability, and strong support from an extensive community of developers and users made them ideal for the needs of the project.

ROS serves as the foundational framework for robotic development, offering an extensive collection of libraries and tools tailored for robotic development. It provides the necessary infrastructure for building and operating robotic systems. Within ROS, RVIZ and MoveIt play pivotal roles. RVIZ is the interface through which we visualize and interact with our robotic creations in a 3D environment. It allows us to see the world from the robot's point-of-view and to direct its actions. MoveIt, on the other hand, takes on the intricate task of motion planning, ensuring that robot's movements are precise, efficient, and safe. Gazebo complements these tools by offering a simulation environment where robots can be tested virtually. It models real-world physics, allowing for accurate and realistic testing scenarios. This is especially valuable in robotics, where real-world testing can be costly and complex.

2.1 Introduction to ROS

Each Robot Operating System (ROS) is designed to ensure compatibility with a specific Linux distribution. In this work, Ubuntu Jammy (22.04) was used as the Linux variant to host the compatible ROS distribution. ROS Humble Hawksbill was used, released on May 23, 2022 and valid until May 2027. Contrary to what its name suggests, ROS is not actually an operating system. Rather, it is a comprehensive collection of software libraries and tools that serve as a basic framework for developing advanced robotic systems. It is a modular open source middleware that greatly streamlines the process of robot software development [open robotics,].

2.1.1 Nodes in ROS

The core of a ROS-based system is a group of many programs (so-called nodes) that operate simultaneously and communicate with each other [josh, 2021]. Each node resembles a traditional command line program, but is supported by ROS libraries, making it a recognized entity within the ROS ecosystem. Nodes are typically specialized and focus on individual functions, such as computing trajectories or processing sensor data, thereby contributing to the collective intelligence of the system.

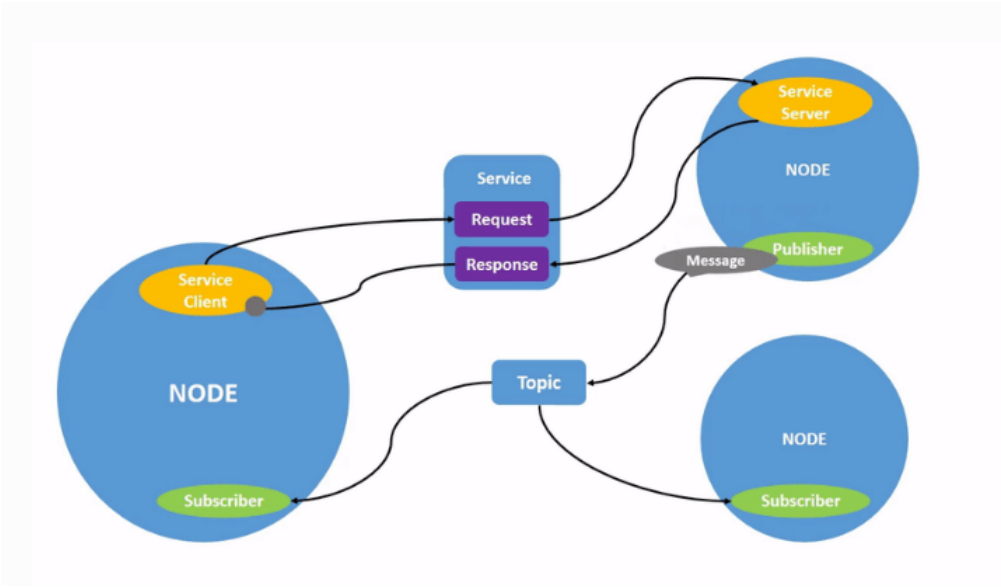

Figure 2.1: Schematic representation of nodes interacting through topics and services [noa, b]. Subsequent sections will provide an expanded discussion on topics.

2.1.2 Topics and Communication

Communication within a ROS network is facilitated through topics and messages. A topic serves as a named bus where nodes, referred to as publishers, broadcast messages. These messages can then be received by other nodes, known as subscribers, which are tuned into the specific topics [josh, 2021].

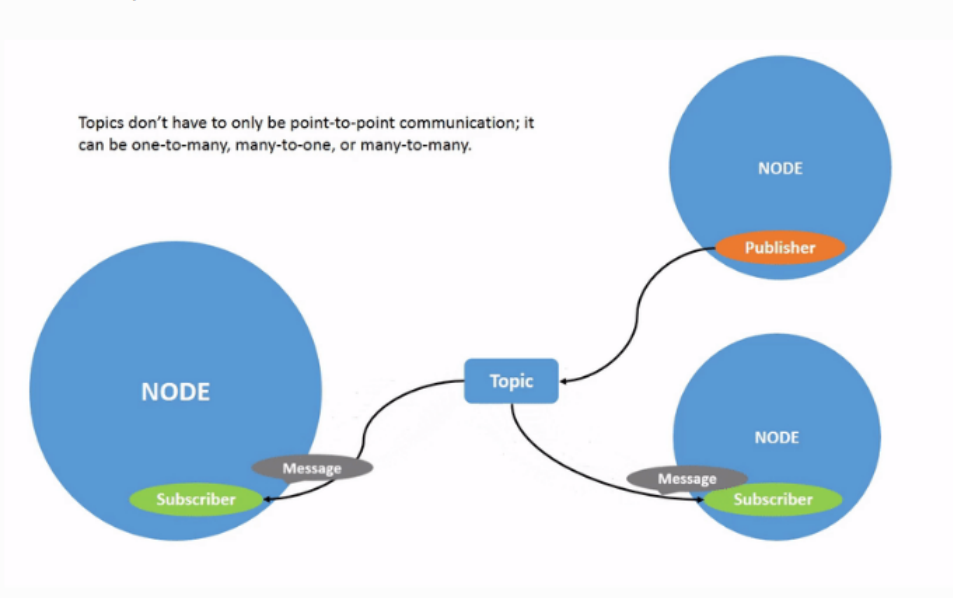

Figure 2.2: Illustration of node-to-node communication via topics, highlighting the flow of data from publishers to subscribers [noa, c].

2.1.3 Launch Files

The repetitive nature of robotics projects requires running the same nodes frequently. Starting each node manually in a new terminal can quickly become tedious, especially for complex projects. ROS addresses this challenge using a scripting mechanism known as a "launch file." This system streamlines the setup process by allowing the user to configure and start multiple nodes at the same time.

2.1.4 Packages

Structurally, ROS organizes components into packages. A package is a coherent collection of related files, including nodes, configuration files, robot models, libraries, and more. This packaging system improves the manageability and scalability of robot software, making it easier to both develop and deploy.

Figure 2.3: Overview of the package organisation [josh, 2021].

2.2 RVIZ

RVIZ enables the graphical visualization of data in 3D and the interaction with the robot by sending control commands to the visualized objects. Hence, from within RVIZ, one can direct the movement of a robot or other parts of its mechanics. By integrating these two aspects, RVIZ provides the capability to observe how the robot operates within its environment. Essentially, RVIZ acts as a window to perceive through the robot's "eyes" and a medium to instruct it on how to respond.

2.3 MoveIt

MoveIt2 is a system designed for precise robotic motion planning. It calculates the exact sequence and angles of each joint movement to guide a robot's end effector—like a gripper or tool—to a specific target. This process is complex and requires a sophisticated approach to ensure that the arm moves efficiently, follows a collision-free path, and adheres to the robot's physical limitations.

As an example, let us imagine that a robot arm needs to pick up an object (see Fig. 2.4). Each joint must move in harmony to position the end effector accurately. Such an endeavour requires a detailed plan, a set of instructions for every joint, specifying not only the final position but also the route to get there. This includes specific angles, speeds, and the timing of each joint's movement in relation to the others, ensuring a smooth and error-free reach to the target.

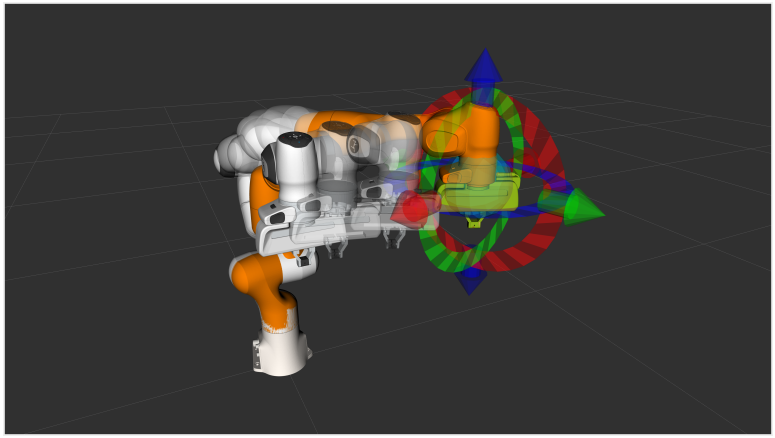

Figure 2.4: Path planning in MoveIt2 on a robotic arm: the user can manipulate the end effector's position using on-screen arrows. MoveIt2 then calculates the optimal route by determining the necessary joint positions to transition from the starting position to the final position set by the end effector's placement.[noa, a].

Creating such a motion plan is a complex challenge, one that MoveIt2 is adept at solving. The tool specializes in determining the paths joints should follow to move an arm from one position to another and then manages its execution, communicating the plan to the robot's joints through ROS. This ensures that the robot arm executes the planned movements accurately and safely, essential in environments where precision and safety are paramount.

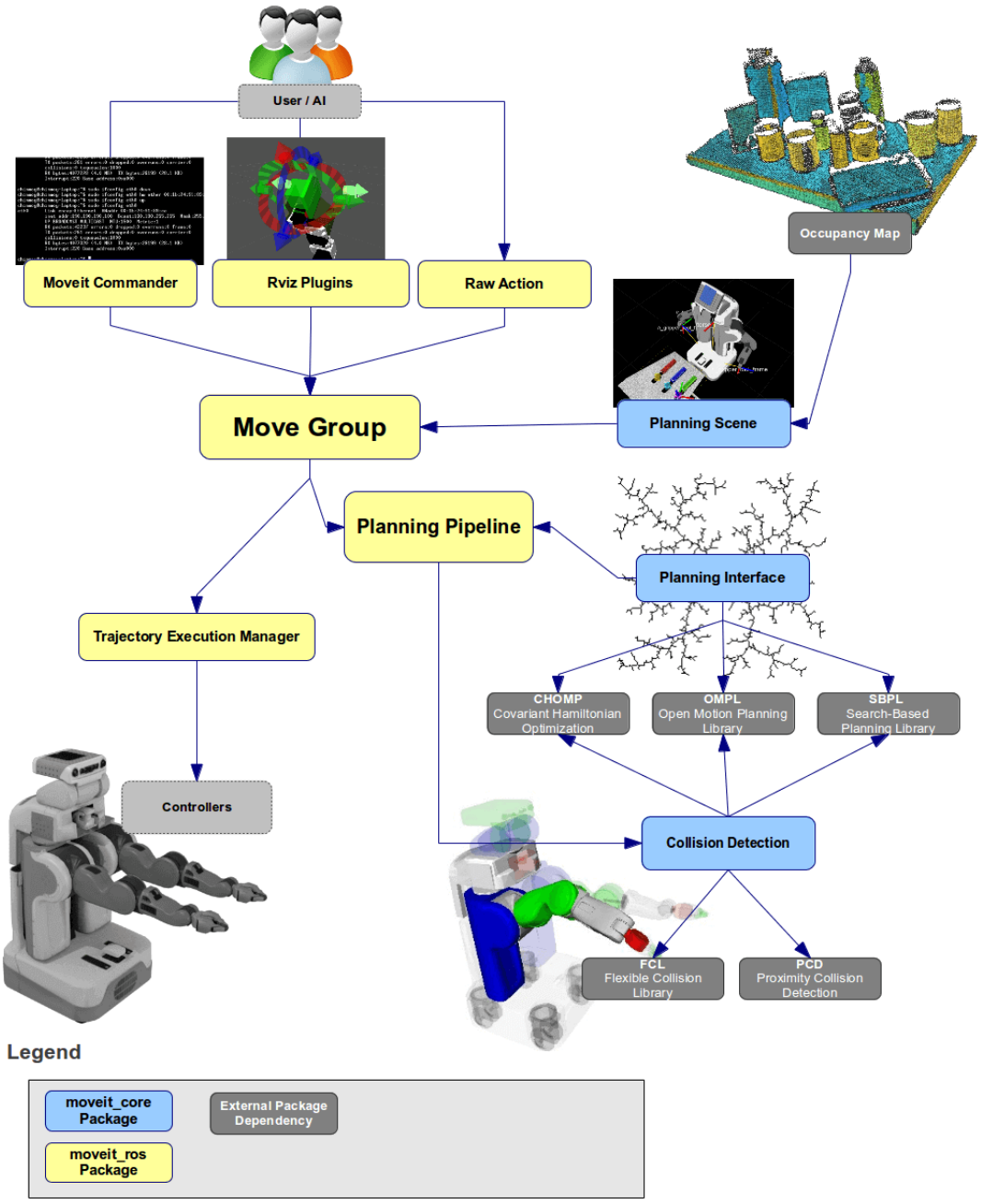

Figure 2.5: System Architecture: Quick High Level Diagram of MoveIt [Robotics, a].

The global MoveIt pipeline represented in Fig.2.5 outlines the detailed functions of MoveIt. The **User/AI** represents the input to the system, which could be a human operator or an artificial intelligence algorithm that provides instructions or commands to the robotic system. It is going to send information to the **MoveIt Commander** (in python), the **RVIZ Plugins** (through a GUI) and send **raw actions**. The **MoveIt Commander** is a command-line interface that allows users to send commands directly to MoveIt for execution. The **RVIZ Plugins** can be used within RVIZ to visualize and interact with the robot's environment and planning scene. Finally, **Raw Action** are raw commands or actions that are sent to the robot, which have not been processed or planned yet. All those orders comming from

the MoveIt commander, RVIZ plugins and raw actions, are directed to the **Move Group** (see Fig.2.6). This node acts as a unifying element, bringing together different components to provide a set of ROS actions and services available for user engagement [Robotics, a]. It receives input from the user or AI and communicates with both the **planning pipeline** and the **controllers**. The move_group node retrieves three types of data from the ROS installation server:

- URDF (Unified Robot Description Format): The node looks for the robot_description parameter to get the robot's URDF, which describes its physical configuration [Robotics, a].
- SRDF (Semantic Robot Description Format): It looks for the robot_description_semantic parameter to access the robot's SRDF, which is typically designed by the user through the MoveIt configuration wizard (see Chapter 4 section 4.2.3 MoveIt and RVIZ) and details semantic attributes and constraints of robots [Robotics, a].
- MoveIt Configuration: Node that collects additional configuration details tailored to MoveIt, such as global boundaries, kinematics, and motion planning parameters. These configurations are created by the MoveIt configuration wizard and are saved in the configuration directory of the robot's MoveIt configuration package on the ROS installation server [Robotics, a].

The **Planning Pipeline** is the path the system uses to plan the robot's motion. It involves several potential planning algorithms:

- CHOMP (Covariant Hamiltonian Optimization for Motion Planning): An optimization-based motion planner that uses gradient descent to minimize a cost function [Robotics, a].
- OMPL (Open Motion Planning Library): A library that provides various sampling-based motion planning algorithms [Robotics, a].
- SBPL (Search-Based Planning Library): A planning library that utilizes graph search algorithms [Robotics, a].

The **Collision Detection** checks for potential collisions between the robot and its environment during planning. It utilizes two main libraries for collision detection:

- FCL (Flexible Collision Library): A library for collision and proximity queries [Robotics, a].
- PCD (Proximity Collision Detection): A method for detecting when objects are close to each other [Robotics, a].

On the other hand, the **Move Group** also has to take into account the Occupancy Map, a representation of the environment where the robot operates, including obstacles and free space, often used for navigation and obstacle avoidance. Once a trajectory is planned, the **Trajectory Execution Manager** is responsible for executing the trajectory, interfacing with the robot's controllers to move it along the

planned path. The **Controllers** are the hardware-level interfaces that receive commands from the Trajectory Execution Manager to control the robot's actuators and move it as planned.

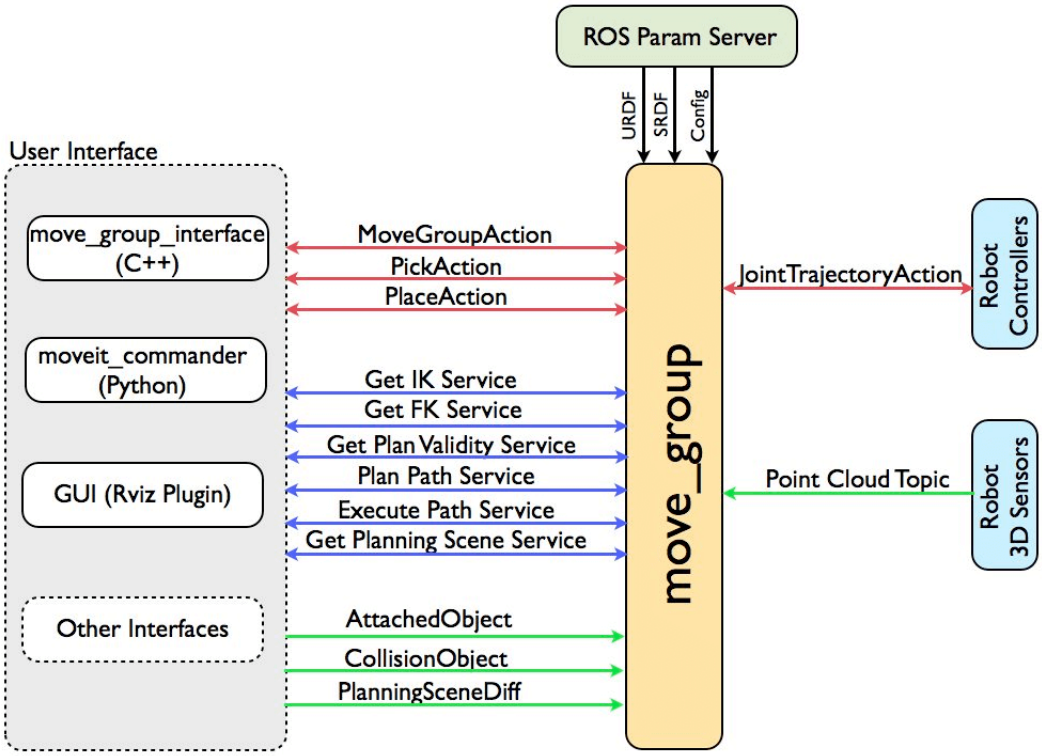

Figure 2.6: System Architecture: Move_group node [Robotics, a].

Robot Interface

The Robot Interface section of the MoveIt architecture is responsible for the interaction between the 'move_group' node and the robot hardware, which occurs through ROS topics and actions. Following, is a detailed explanation of each component mentioned:

Joint State Information

The 'move_group' node subscribes to the '/joint_states' topic to obtain the positions of the robot's joints [Robotics, a].

Transform Library

'move_group' employs the ROS TF (transform) library to monitor the robot's transform data, which includes the robot's pose (the position and rotation of a robot's end effector) in a global context [Robotics, a]. However, 'move_group' does not publish transform information itself; this should be done by a 'robot_state_publisher' running on the robot [Robotics, a].

Controller Interface

The node communicates with the robot's controllers using the 'FollowJointTrajectoryAction' interface, a ROS action that requires a corresponding server on the robot to respond to these action calls [Robotics, a]. The 'move_group' node acts as a

client, requesting action services from the server on the robot [Robotics, a].

Planning Scene

'move_group' relies on the Planning Scene Monitor to maintain a model of the environment and the robot's current state, which includes the robot itself and any objects it is carrying [Robotics, a]. This planning scene is a crucial aspect for collision checking and ensuring safe navigation within the robot's operating space [Robotics, a].

2.3.1 Motion Planning

MoveIt is designed to be adaptable and supports integration of various motion planning tools through a plugin interface. This flexibility allows MoveIt to communicate seamlessly with multiple motion planning libraries [Robotics, a]. Communication with these motion planners is made easy via the ROS action or the service provided by the 'move_group' node[Robotics, a]. By default, move_group is configured to use the Open Motion Planning Library (OMPL) through the interface set up by the MoveIt Setup Assistant.

2.3.2 Kinematics

The RobotState class includes the ability to perform forward kinematics and calculate the Jacobian. For inverse kinematics, MoveIt by default uses a plugin that uses a numerical solver from the KDL library based on Jacobians. Inverse kinematics involves specifying a target position and orientation for the end effector of a robotic arm and computing the necessary joint angles to achieve this arrangement. Essentially, it is the process of determining the settings of the robot's joints that result in the end effector reaching the desired location.

2.3.3 Collisions

Collision detection in MoveIt is primarily based on the Flexible Collision Library (FCL), one of the selectable collision detection backends. It provides an integrated collision environment by merging features like CollisionRobotFCL and CollisionWorldFCL.

CollisionRobotFCL

The robot's link geometry, including axis-aligned bounding boxes (AABBs), is calculated once and stored, in their local frames [Robotics, 2019]. To self-check collisions, a new FCL collision manager is created and all robot links are added to it [Robotics, 2019]. This process is efficient because the geometry is already stored and only needs to be referenced. First, large phase collision testing is performed using stored AABBs, which are updated with their general position to quickly identify potential collisions [Robotics, 2019]. If the wide phase indicates a potential collision, narrow phase collision checking is performed to detect collisions between overlapping pairs in detail.

CollisionWorldFCL

This component manages the environment or "world" in which the robot operates. It maintains a persistent collision manager containing all objects in the environment, organized in a tree structure for quick collision checking [Robotics, 2019]. When testing for collisions between the robot and the environment, each link in the robot is tested separately against objects in the world [Robotics, 2019]. The robots' pre-computed geometries stored in CollisionRobotFCL are used, updating their general pose based on the robot's current state to determine whether there will be collisions with objects in the world [Robotics, 2019].

Simply put, CollisionRobotFCL focuses on detecting collisions between the robot's parts, while CollisionWorldFCL manages the robot's interactions with objects in its environment. Both components leverage efficient data structures and algorithms to quickly evaluate potential collisions and provide accurate results to inform robot movement planning decisions.

2.4 Gazebo

Gazebo is an open-source 3D robotics simulator, developed by Opend Robotics. It should be noted that he tool is not by default a part of ROS (see Fig.2.7). It simulates how an object will behave in the real world, by simulating the mass, the inertia, the friction, etc. The tool allows to upload digital simulations of the robot and then experiment on it by applying various forces, taking things such as friction into account. As testing on real hardware can be expensive and time-consuming, having a good simulation environment is a valuable tool in robotics. Gazebo offers a lot of opportunities in the surgical field as well, like among others, testing on real patient anatomy and training before the real clinical practice.

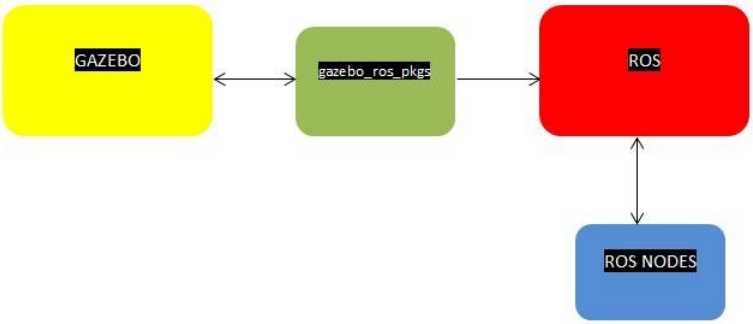

Figure 2.7: Relationship between Gazebo and ROS [Okoli et al., 2019].

2.4.1 Difference between Gazebo and RVIZ

Morgan Quigley, one of the primary creators of ROS, makes a clear distinction between RVIZ and Gazebo in his book "Programming Robots with ROS." He explains that while RVIZ provides a visualization of the robot's perception of its environment, Gazebo simulates and visualizes the actual events occurring within that

environment.

Chapter 3

Project context

3.1 Overview of the Multi-Year, Multi-Team Research Project Components

This section examines the collaborative efforts in this multi-team (John Hopkins University and SNRLAB), multi-year research project realised before this thesis. It highlights the complex interactions between technology and medicine, highlighting the development and deployment of sophisticated robotic systems designed for precision medical procedures.

3.1.1 Robot architecture

Robot description

The smart needle robot used for animal studies is a 3 degrees-of-freedom robot (X, Y, Z), as shown in Fig.3.1. It uses 3 rails, the X and Z direction allowing the needle to correct its trajectory, and the Y direction being the insertion depth. The rails are steered by a motor, guiding the needle to the desired placement.

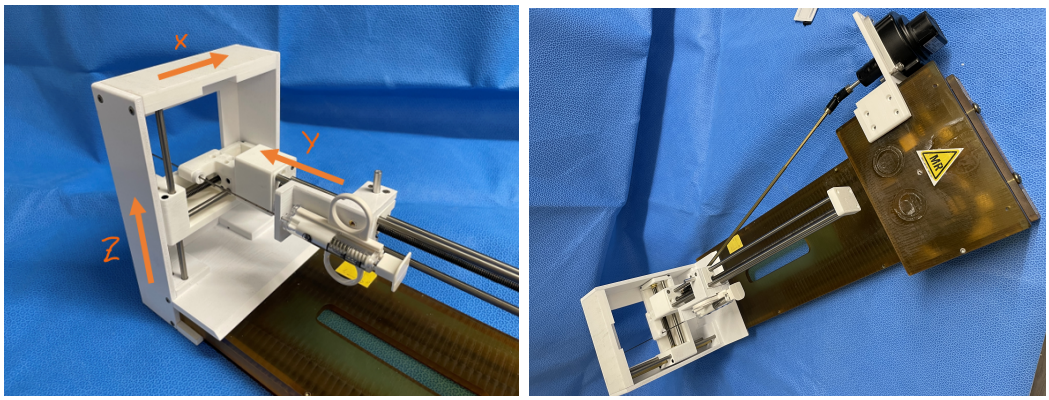

Figure 3.1: Robot for prostate biopsy. The first image shows the coordinates used and the 3 rails. The second image shows the motor that receives the information to move the needle.

A sensorized needle using fiber Bragg grating (FBG)

While entering a tissue, due to needle-tissue interaction and elastic skin contact, the needle will deflect to some extent, like shown on Fig.3.2. Needle deflection causes large targeting errors. For precise needle placement in tissue, it is advantageous to track the needle's path during needle insertion. Fiber Bragg grating (FBG) sensors proved to be effective for that kind of job. An FBG sensor is a fiber-optic strain gauge, made of a series of grated mirrors called Bragg reflectors, that have a reflective index that changes periodically, with a period Λ, as seen in Fig.3.3 [Yong Bai, 2014]. FBG reflects particular wavelength light that is in face with and transmits all others, as seen in Fig.3.3. When the sensor undergoes a change in stress, the distance between the Bragg reflectors changes, allowing different wavelengths to be reflected. This shift in wavelength is proportional to the strain the fiber experiences [Fatschel et al.,]. FBG's are placed along the needle to measure bending strain and estimate the curvature of the needle, which reflects the interaction between the environment and the needle. The output will be an array of 400 points, representing the needle shape. One of the positive features of FBG is that they are MRI compatible. This device has been implemented by John Hopkins University, who are close collaborators on the Smart Needle robot.

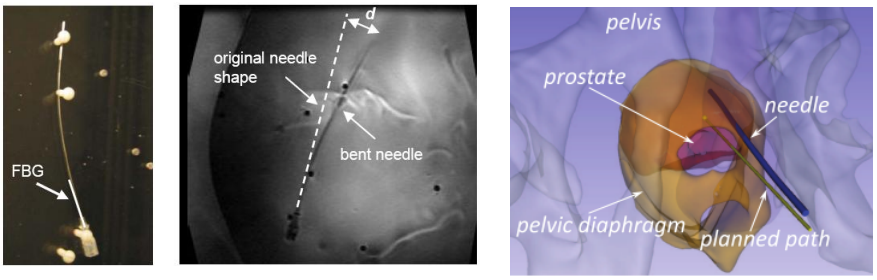

Figure 3.2: Needle deflection. Image a and b: MRI scan of a needle, using FBG sensor to estimating the curvature [SanthiElayaperumal, 2019]. Image c: [Bernardes et al., 2023] 3D models of the prostate. One can observe the planned path and the needle deviated from it, because of interactions between the needle and the tissue. It causes a large targeting error.

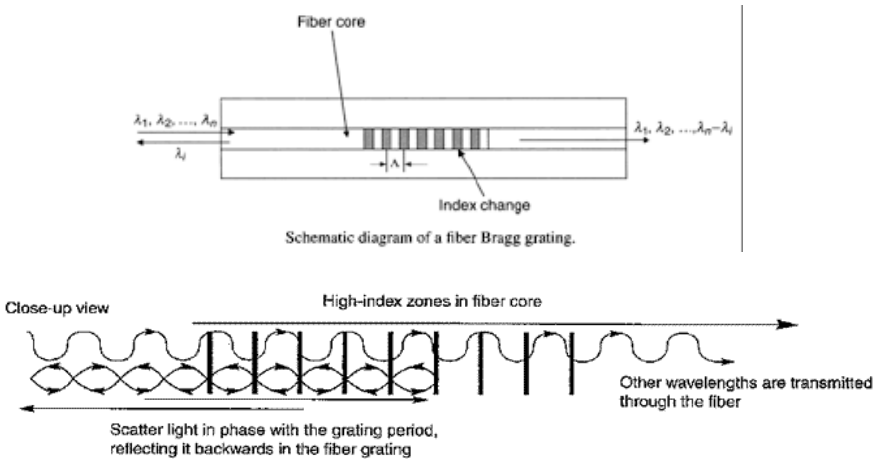

Figure 3.3: Upper image: Simple FBG, with λ the wavelength of the light, and Λ the period of the Bragg reflectors. Lower image: Light reflected and transmitted in function of the grating period, following the equation: $\lambda_b = 2n_e\Lambda$, where λ_b is the reflected wavelength and n_e is the refractive index [fbg,].

3.1.2 Integrated Communication System

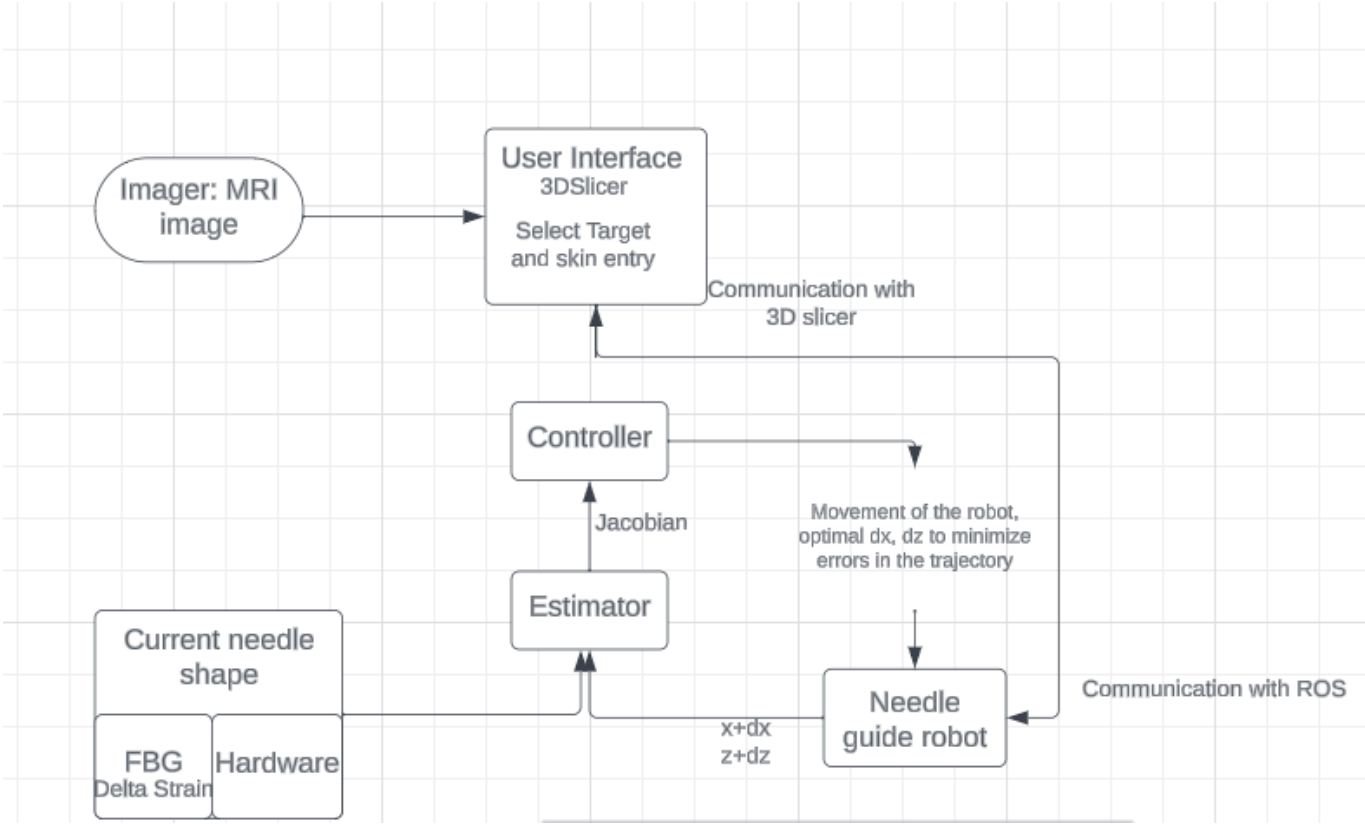

Figure 3.4: Schematic overview of the different components of the "Smart needle" robot.

User interface

The User Interface is a critical component as it is the main link to the operating physician. It inputs patient-specific parameters such as the skin entry point. Additionally, it integrates data from medical imaging, specifically MRI scans of the prostate, visualized in 3D Slicer software. This integration allows physicians to clearly define targets for the biopsy procedure. The User Interface then communicates these details to ROS (Robot Operating System) through IGTLBridge, a tool facilitating data transfer between 3D Slicer and ROS. The coordinates are given in the fiducial frame (the Z frame).

ROS2 needle guide robot

This module is central to the robotic operation, and responsible for the precise movement of the biopsy needle. It actively publishes the needle's position after each step insertion, ensuring continuous tracking. The system incorporates a feedback loop, wherein it evaluates position accuracy through encoder readings (stage control node) and dynamically adjusts the needle's path (needle pose sensor node). The updated position data is then published (stage state builder node) and relayed back to the 3D Slicer for real-time visualization.

Needle shape model

This component is crucial for accurate needle shape. It consists of two main packages: hardware integration and the needle shape publisher. The hardware integration acquires data from the hyperion interrogator (Fiber Bragg Grating - FBG), translating this information into the physical shape of the needle. This is crucial for understanding needle deflection, which can impact the accuracy and safety of the biopsy. The model outputs a detailed representation of the needle shape, providing an array of 400 data points.

Trajectory control

This part is essential to put all of the previous parts together, ensuring the communication between all the different coordinate frames. Divided into two sub-parts —estimator and controller— this segment is pivotal for trajectory precision. The estimator calculates the needle's pathway, while the controller decides how to adjust movements to minimize errors. This system is integral in ensuring that the needle reaches the target accurately. By considering the current needle position, skin entry point, and desired target, it establishes a linear relationship using the Jacobian matrix. This relationship guides the needle from its current location to the target, constantly updating the needle guide robot with instructions for movement adjustments.

Chapter 4

Methodology

After having the tools and the context explained in the previous chapters, we can now focus on the heart of this thesis and its contribution to the broader research project: the design and development of the 3D simulation of the 3 degrees-of-freedom robot for prostate biopsy used in animal studies. The process involves modeling the robot and the prostate using URDF and stl files, based on MRI data. Key elements include simulating the robot's motion and a basic model of needle deflection in Gazebo, using ROS2 control to activate the actuators, and optimizing path planning with MoveIt, using the KDL plugin to solve the inverse kinematics problem.

4.1 Modelling

4.1.1 Describing a robot with URDF

The robot's physical information is called the robot description, and the URDF (Unified Robot Description Format) file is the tool used to store all the physical information or robot description in the same place, allowing the different software to access the information needed (see Fig.4.1).

Links and Joints

URDF describes a robot as a tree of links, that are connected by joints. The links represent the physical components of the robot, and the joints represent how one link moves relative to another link, effectively defining the location of the links in space.

Figure 4.1: Fragment of the code used to define the links and joints of the robot for prostate biopsy.

The *smart needle* robot's simulation in the URDF file is composed of 1 base and 3 links shaped as boxes representing the X, Y and Z rails, as shown in Fig. 4.2. Its needle is segmented in 9 parts, forming an articulated needle model, to simulate the deflection. Five links are used to simulate the needle shape, and four other tiny links are used to connect the five main links together, in order to allow a continuous motion for the needle, as showed in Fig. 4.3. The joint used to connect the base (in orange on Fig. 4.2) to the Z rail is a fixed joint, but the joints used to connect the link Z with the link X, the link X with the link Y are prismatic joints (see Fig.4.4). The joint used to connect the needle to the link Y is also a prismatic joint. The joints used to connect the 9 links of the needle are the combination of 2 revolute joints (see Fig.4.4), one revoluting around the X axis and the other one revoluting around the Y axis. Combining all those links and joints, allows the needle to reach the target in the prostate, and to correct its path moving along the links Z and X.

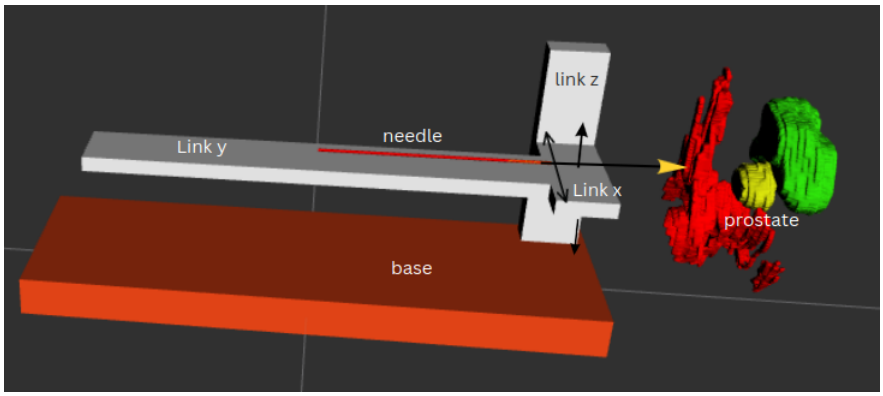

Figure 4.2: Full structure of the robot (in white), the needle (in red) and the prostate (in yellow). The arrow represent the direction of movement. In green, the bladder, and in red, other anatomical structures adjacent to the prostate, like the corpus spongiosum, the urethra of penis, the obturator internus, the rectum, etc.

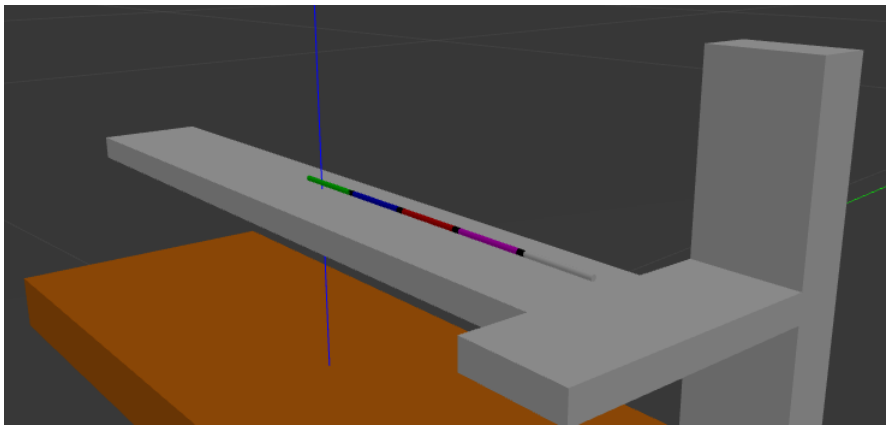

Figure 4.3: The articulated needle with 9 segments. Each segment allows a range of movement around the X and Y axis, allowing the simulation of the needle curvature.

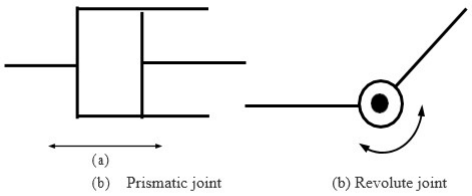

Figure 4.4: Prismatic and revolute joints.[VV et al., 2020].

4.1.2 Prostate modelling

The process of modelling a patient-specific prostate for use in surgical simulations and robotic surgery begins with obtaining Magnetic Resonance Imaging (MRI) of the patient's prostate. MRI is a medical imaging technique that produces detailed images of internal body structures and is particularly effective for visualizing soft tissue like the prostate. Once the MRI images are acquired, they are imported into the

program 3D Slicer, within which a crucial process called segmentation is performed. Segmentation is the process of partitioning a digital image into multiple regions or segments to simplify or change the representation of an image into something more meaningful and easier to analyze. In the context of prostate modeling, segmentation involves delineating the prostate as well as adjacent anatomical structures.

After segmentation, the segmented prostate and the adjacent anatomical structures are exported as stl (stereolithography) files. The stl file format is widely used for 3D printing and computer-aided design and is compatible with many different types of 3D modelling software. This format describes only the surface geometry of a three-dimensional object without any representation of color, texture, or other common model attributes. These stl files are then used to create the prostate model in the URDF file.

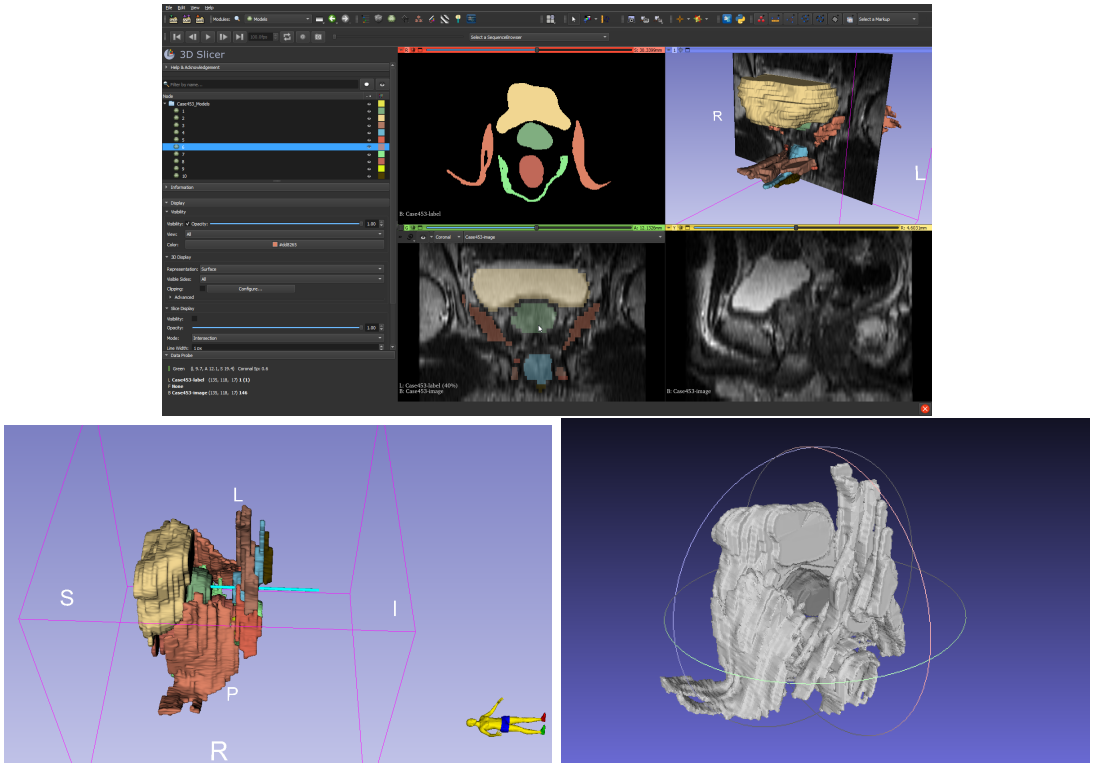

Figure 4.5: The upper image represents the MRI image in 3D slicer and the segmentation done based on this image. The left image represents the 3D segmentation of the prostate taken from 3D slicer, showing how the needle should be inserted inside the prostate. The right image represents the stl file that the URDF file uses as input to construct the prostate and the anatomical structures adjacent to the prostate

4.2 Simulation

4.2.1 Simulation with gazebo

In the previous section, the usage and development of a URDF file to describe the structure of a robot was explained. However, Gazebo, the robot simulation environment, works differently. It mainly uses SDF (Simulation Description Format) files,

similar to URDF but with a wider scope [Newans, 2022b]. Unlike URDF which focuses only on robotic structures, SDF can include robots, environments, and other models within that environment. Fortunately, there is no need to generate both URDF and SDF files for our robot simulation. Gazebo is equipped to seamlessly convert URDF files to SDF format. However, this automatic conversion does not include all the elements necessary for the Gazebo to function optimally. Additional components such as Gazebo tags, plugins, and inertial properties need to be integrated. To import gazebo-specific data into URDF, we use the Gazebo tag. These tags allow Gazebo to create SDF files while having no consequences for other systems that interpret URDF. Regarding the interaction between Gazebo and external systems such as ROS, plugins are essential. These are additional code libraries that, once installed and referenced in our URDF, will be executed by Gazebo as needed. Finally, Gazebo requires each link in the simulation to be associated with an inertial tag. This tag is essential for accurately simulating the inertia of each component, contributing to a more efficient and realistic simulation experience.

4.2.2 ROS2 control

The nature of robot operation includes processing input (from the operator or environmental sensors), making decisions, and activating actuators. At the heart of this process is the Controller Manager, a key component that coordinates the integration of hardware drivers and controllers using the plugin system [Newans, 2022a]. For a robot to be compatible with ROS2_control, it needs a "hardware interface" (also called a hardware component). This code interacts with the physical hardware and normalizes it for ROS2_control [Newans, 2022a]. The hardware interface acts as a bridge, simplifying the complexity of the hardware into two main interfaces: the command interface and the status interface [Newans, 2022a] (see FIG. 4.6). Command interfaces are things that can be manipulated, like actuators, while state interfaces are things that can only be observed, like sensors. The only controllable variable in the simulation is the position of the joints, so we will have various hardware interfaces, that will have as many command interfaces as we need to control our robot. Meaning the joints XY and ZX and needle insertion, to control the needle placement along the X, Y, Z axis and also each joint being part of the articulated needle model ($link\_i\_joint$ and $link\_i\_aux\_joint$, for i between 2 and 5) as seen in Fig.4.7. Regarding the robot's state interfaces, we can measure both the velocity and the position of the joints, using the encoders . It will have 2 state interfaces, position and velocity for each joint described in the command interface, as seen in Fig4.8.

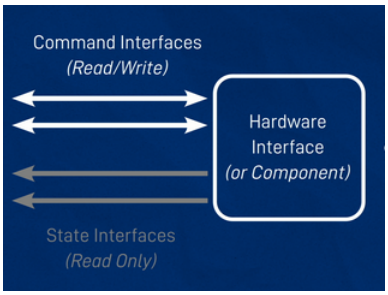

Figure 4.6: Command interfaces and state interfaces' communication with the hardware interface [Newans, 2022a].

```
command interfaces
        joint_needle/position [available] [claimed]
        joint_xy/position [available] [claimed]
        joint_zx/position [available] [claimed]
        link_2_aux_joint/position [available] [claimed]
        link_2_joint/position [available] [claimed]
        link_3_aux_joint/position [available] [claimed]
        link_3_joint/position [available] [claimed]
        link_4_aux_joint/position [available] [claimed]
        link_4_joint/position [available] [claimed]
        link_5_aux_joint/position [available] [claimed]
        link_5_joint/position [available] [claimed]
```

Figure 4.7: Command interfaces of the simulation, on Linux. On the simulation, the only parameter the can be controlled is the position of the joints. We will therefore have various hardware interfaces, that will have as many command interfaces as we need to control the robot, meaning the joints XY and ZX and needle insertion. This will control the needle placement along the X, Y, Z axis and also each joint being part of the articulated needle model ($link\_i\_joint$ and $link\_i\_aux\_joint$, for i between 2 and 5).

```
state interfaces
        joint_needle/position
        joint_needle/velocity
        joint_xy/position
        joint_xy/velocity
        joint_zx/position
        joint_zx/velocity
        link_2_aux_joint/position
        link_2_aux_joint/velocity
        link_2_joint/position
        link_2_joint/velocity
        link_3_aux_joint/position
        link_3_aux_joint/velocity
        link_3_joint/position
        link_3_joint/velocity
        link_4_aux_joint/position
        link_4_aux_joint/velocity
        link_4_joint/position
        link_4_joint/velocity
        link_5_aux_joint/position
        link_5_aux_joint/velocity
        link_5_joint/position
        link_5_joint/velocity
```

Figure 4.8: State interfaces of the simulation, on Linux. The simulation has 2 state interfaces: position and velocity, for each joint described in the command interface.

Controllers serve as the main interface between the ROS ecosystem and ROS2_control. They monitor ROS objects for control inputs, which may include joint positions or

the velocity of a moving object. Using specific algorithms, these controllers determine the appropriate speed or position of the actuator and transmit this information to the appropriate hardware interfaces [Newans, 2022a]. Additionally, the controller has the ability to publish command responses or status data received from the hardware to ROS topics. They have the ability to facilitate the flow of information in both directions, or even both at the same time [Newans, 2022a].

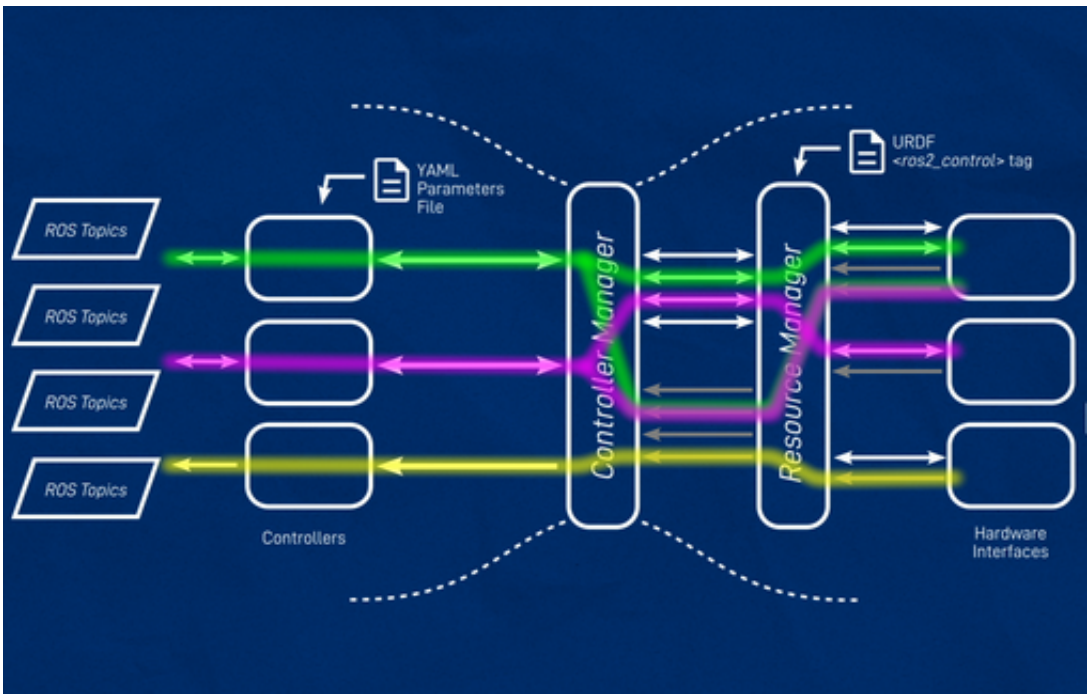

Figure 4.9: The arrows in the diagram serve as visual guides to indicate the direction of information exchange between different components. The controller is configured to receive input from ROS objects, which they analyze and interpret, ultimately generating commands. These commands are sent to the hardware interface through the resource management intermediary. The resource manager compiles a set of hardware interfaces, presenting them as a common set of command and status interfaces to the controller. Meanwhile, the controller manager orchestrates this interaction by instantiating the necessary controllers defined in the YAML settings file, ensuring that they correspond to the appropriate hardware interfaces identified in the YAML settings file. detailed in URDF. [Newans, 2022a].

It is the controller manager's responsibility to align requested controllers with the appropriate command and status interfaces provided by the resource manager [Newans, 2022a]. To configure these controllers, we create a YAML file detailing the necessary settings and include it in the Controller Manager. After successfully loading these configurations, we have the ability to launch or stop the controller as needed.

4.2.3 MoveIt and RVIZ

Setup Assistant is a user-friendly tool that simplifies the creation of the files required for MoveIt ROS2 packages, as shown in Fig.4.10. By providing a URDF file, users can configure their packages (see Fig.4.11) efficiently without manual

code editing. To build the complete package, one must go through and complete the sections described by the configuration wizard. Although not all sections are detailed here, the important sections relevant to this project will be covered, starting with the "self Collision" section. The self-collision matrix generator streamlines movement planning by disabling collision checking between link pairs that are always conflict-free [Robotics, b]. It distinguishes link pairs that collide continuously, pairs that never collide, link pairs that are in the robot's default position, or link pairs that are adjacent to each other in the kinematic chain. By adjusting the sampling density, the number of robot poses tested for self-collision is determined [Robotics, b]. To speed up collision matrix generation, collision evaluation is performed concurrently [Robotics, b]. The subsequent "planning groups" section is an integral part of defining the semantics of individual robot components, such as to differentiate an arm or an end effector. For this project, there were two important planning groups: the main structure of the robot, including the base_link, the links for the X, Y, Z axes and the needle itself and the second group contains the needle tip (the end effector). Given the sequential needle topology, kdl_kinematics_plugin/KDLKinematicsPlugin serves as a suitable kinematics solver, with more details available in the "Kinematics and Dynamics Libraries" section. The “end effector label” identifies the needle tip as an end effector, a part located at the end of a robotic arm designed to interact with the environment. In the "MoveIt Controllers" section, it should be noted that MoveIt requires a trajectory controller with the FollowJointTrajectoryAction interface to execute the planned trajectories, transmitting the resulting trajectory to the robot's ROS 2 controller . The MoveIt Controllers framework helps create automated controllers that the MoveIt Controller Managers will use. After the setup wizard requirements are met, the required files are created, including the initialization and configuration files, as shown in Figure4.11. The simulation can be visualized in RVIZ as seen in Fig.4.12, but we can also use the Gazebo simulation if we want to incorporate the physical parameters such as mass, inertia, etc. Indeed, there is an easy way to connect RVIZ and Gazebo, by just including Gazebo in the launch file.

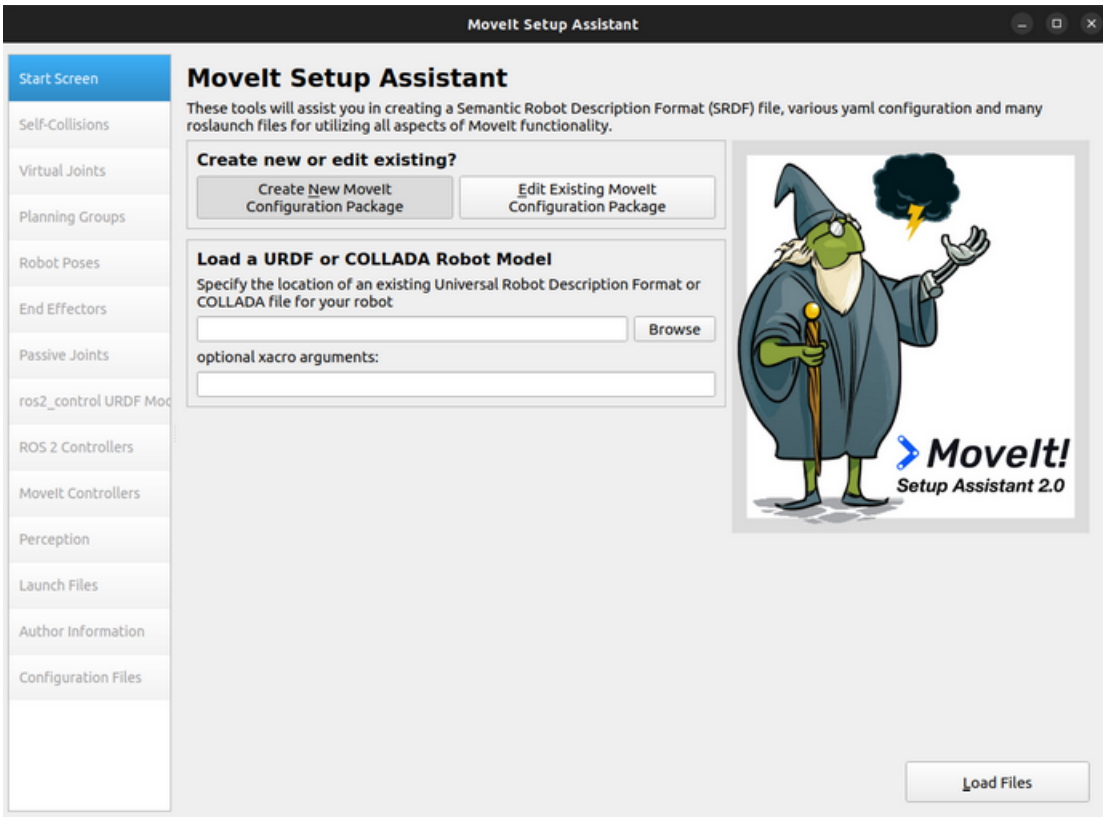

Figure 4.10: Setup assistant for MoveIt2

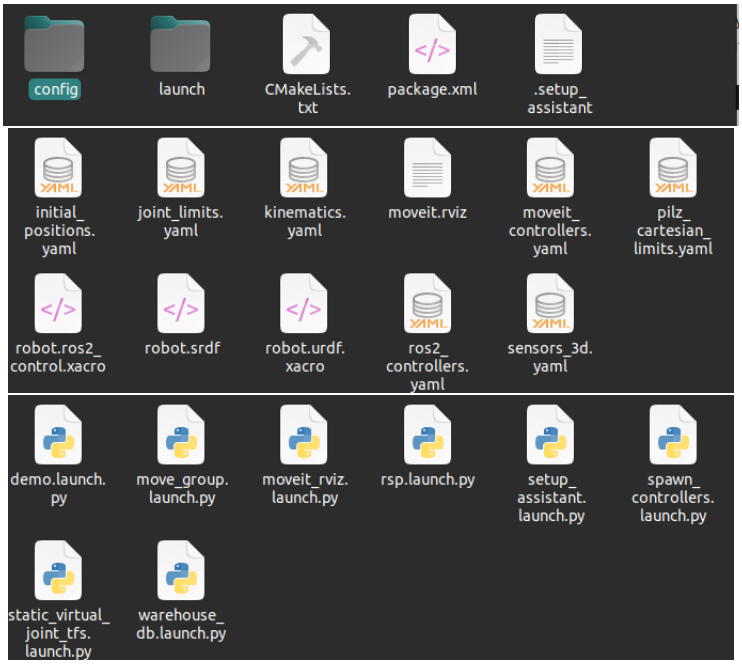

Figure 4.11: The top image shows the complete package assembled by the setup wizard, including all the files needed to build the ROS2 package. The middle image drills down into the contents of the configuration file, while the bottom image provides a view of the launch file.

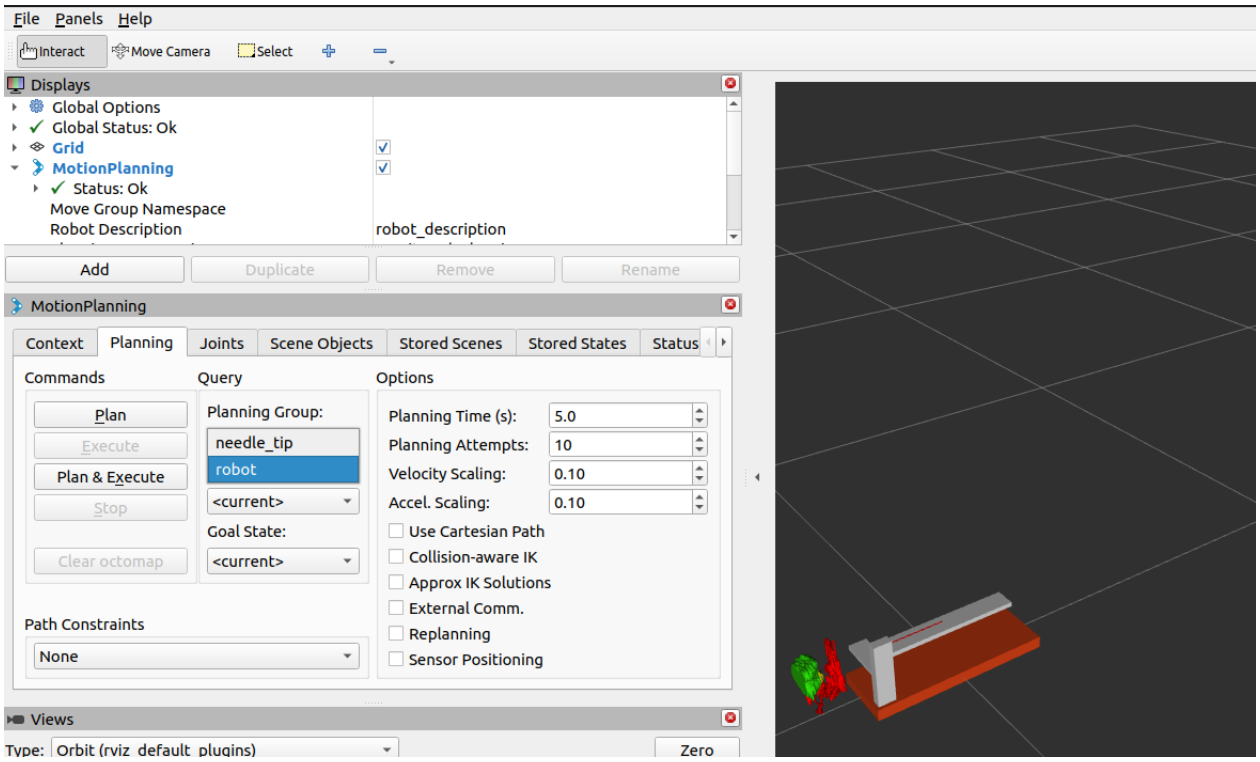

Figure 4.12: The image shows RVIZ with MoveIt functionalities added for motion planning. It is possible to select the planning group and define a random position for that group. It is also possible to play with each joint individually, to create a personalised end position for the robot.

Kinematics and Dynamics Library

The KDL kinematics plugin wraps around the numerical inverse kinematics solver provided by the Orocos KDL package [Robotics, b]. This is the default kinematics plugin currently used by MoveIt. It obeys joint limits specified in the URDF (and will use the safety limits if they are specified in the URDF). The KDL kinematics plugin currently only works with serial chains.

Chapter 5

Results

The robotic procedure was successfully simulated from selection of target up to the modelling of a trajectory to reach this target with a device that simulates the equipment available at SNRLAB (and starting from the MRI images typically acquired at Brigham and Women's hospital). The path planning simulation within MoveIt and RVIZ takes approximately 5 seconds.

5.1 Modelling

Based on visual validation, the URDF model correctly represents the robot structure. Any necessary adjustments to the link length can be easily made by manipulating the source code. As shown in Fig.5.1, the URDF modelling in RVIZ is visually consistent with the real robot. The accuracy of URDFs is an important aspect, as they serve as the basic input for the simulations performed in MoveIt and Gazebo. Mesh files can be used to further refine the appearance, but such improvements are more aesthetically pleasing than functionally accurate.

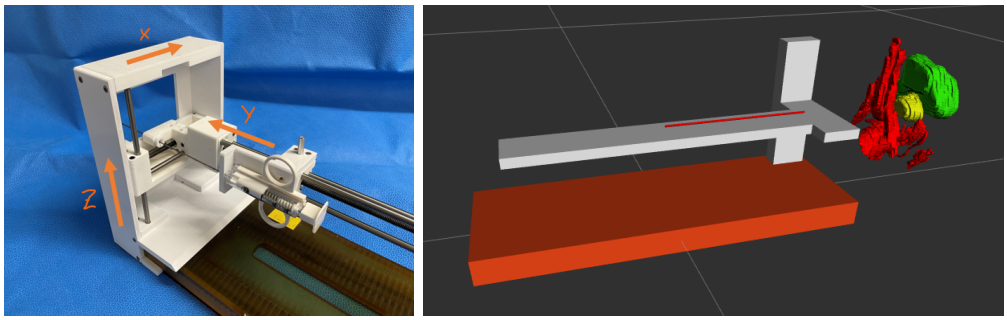

Figure 5.1: You can indeed see that the 3 links x,y,z are well represented, as well as the needle.

The accuracy of the prostate model is highly dependent on the accuracy and resolution of the MRI images and the rigor of the segmentation process. Although these factors are beyond the developer's control, comparative visual analysis of MRI images and 3D slicer segmentation shows acceptable reconstruction of the prostate, as shown in Fig.5.2. Despite the fact that the discretization is visible on the image, this model visually captures the important anatomical features of the prostate for a unique patient, making it a reliable representation for simulation and testing purposes.

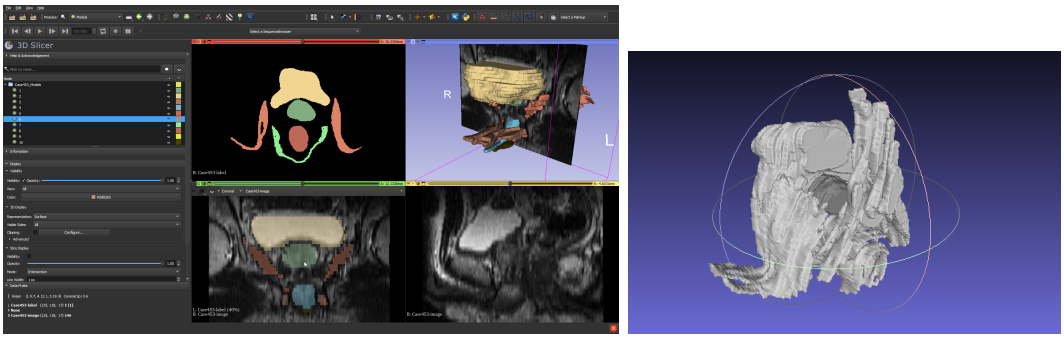

Figure 5.2: Comparison between the prostate image taken from the MRI image in 3D slicer (left) and the stl file inserted in RVIZ (right).

5.2 Simulation

5.2.1 Simulation with Gazebo

The Gazebo simulation is primarily to assess whether the needle within the robotic structure would realistically respond to external forces, with gravity being the main factor. In the gazebo simulation, unstable initial conditions were introduced for the needle. We could observe the articulated needle behave like a moving snake, therefore reacting to external forces.

5.2.2 MoveIt and RVIZ

There are three main ways to use path planning in MoveIt (there is also a fourth incomplete method that is described further in the appendix section). The first method is to set a random ending position (see Fig.5.3) for the robot and have MoveIt calculate the inverse kinematics needed to reach that position. Although this approach confirmed the robot's mobility and the effectiveness of the kinematics solver, it is not particularly relevant to this thesis, as random positions are unlikely to match the specific requirements of intraprostatic positioning. However, that first method served as a valuable preliminary step as it demonstrated the robot simulation's ability to move properly following the shortest path and made joint positions accessible via ROS2 commands.

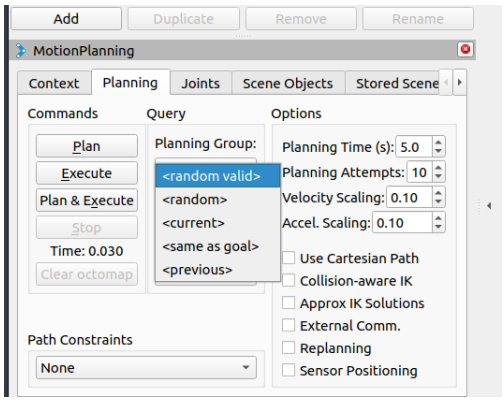

Figure 5.3: MoveIt interface on Rviz: defining a random valid goal for the planning group.

The second method is to manually define the position and orientation of each joint using the MoveIt interface as showed in Fig 5.4. Once a collision-free position is set, the corresponding path is executed, but it's not very practical to reach a specified target, and it will be of no use in real life, because the target is not the input, but the result of the robot's position.

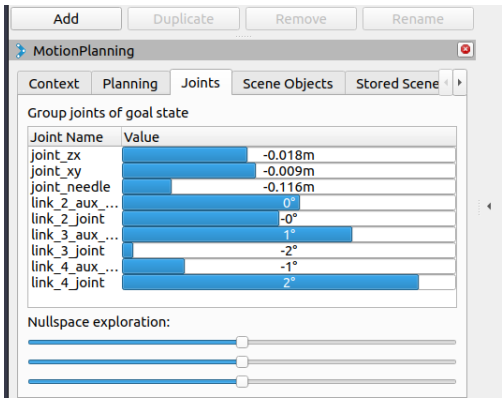

Figure 5.4: MoveIt interface on RVIZ, you can manually move the cursor to define new joint positions.

The third method, which is most relevant to this project, is manually guiding the needle tip (end effector) to the desired location in MoveIt. This is beneficial for our application because the end effector needs to be moved to a target and MoveIt manages the robot's joints accordingly. After movement, joint position values can be accessed via ROS2. It has been visually observed that for every random placement, or placement in the anatomical surroundings of the prostate (see Fig. 5.6 and Fig5.5), the path is always correctly (based on visual approval) planned and executed, choosing the shortest way and avoiding obstacles.

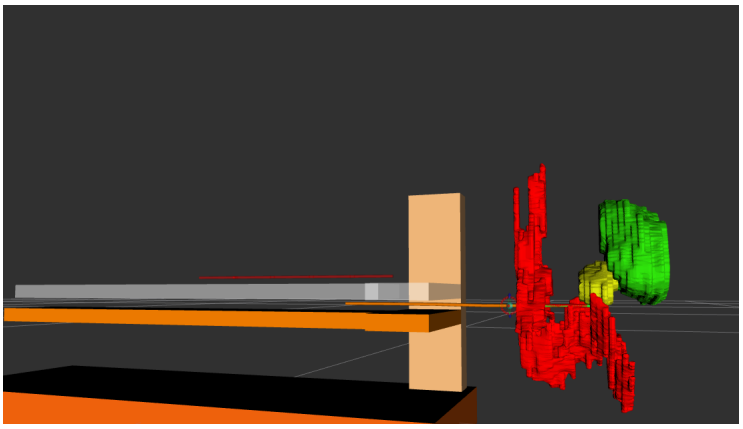

Figure 5.5: Visualisation on RVIZ, using MoveIt. Initial position showed in white, and finial position showed in orange. We can observe the needle tip reaching the prostate.

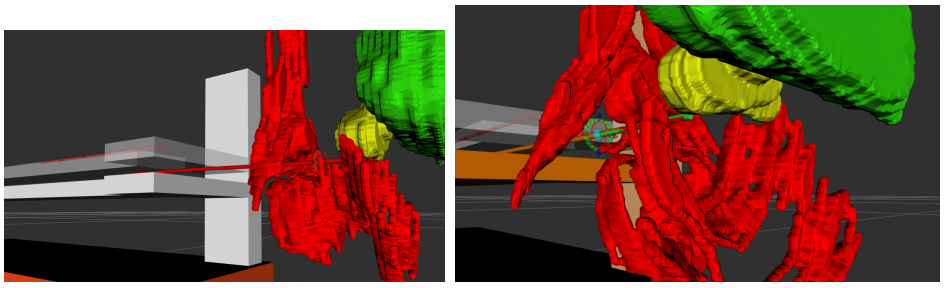

Figure 5.6: Needle penetration into the anatomical structure showed in RVIZ, with the end effector nearly reaching the prostate, following a path planned by MoveIt.

Chapter 6

Discussion

6.0.1 Simulation with Gazebo

The physical parameters of the needle in the simulation do not completely reflect reality. Currently, the needle is only divided into nine parts, which has its limitations. For a more accurate representation, we need to divide the needle into more segments. For comparison, the FBG sensor that provides data about needle shape outputs an array of 400 points. Therefore, to improve the model, ideally the needles in the simulation should have a similar level of segmentation. Additionally, the friction between needle segments and the deflection angle between them are not accurately represented. More comprehensive studies are needed to accurately determine these parameters. However, these are just variables in the code and can be adjusted after further research determines the correct values.
The updated simulation needs to incorporate dynamic anatomical adjustments to accurately model real-world scenarios. This includes adapting to variations in bladder filling, which can alter the spatial relationships in the pelvic region, affecting the trajectory and positioning of instruments. Similarly, changes in rectal filling must be accounted for, as they can impact the accessibility and visibility of the prostate. Additionally, the simulation should simulate the needle compression effect on the prostate tissue. This is critical because the deformation of the prostate due to needle insertion could shift the target location during the biopsy procedure, necessitating real-time recalibration and strategy adjustment.
Making these adjustments will significantly improve the accuracy of the model. The final test of this model is to compare the needle deflection in a real-world scenario where the needle is inserted into a physical phantom to its Gazebo-simulated counterpart with the same physical properties. The goal is to measure and validate the accuracy of the simulation against real results.

6.0.2 MoveIt and RVIZ

The third method (manually guiding the needle tip to the desired location in MoveIt) has two main challenges. First, it is difficult to manually (via the computer's cursor) guide the needle through the complex structure of the prostate, as shown on Fig 5.6. Second, it is difficult to define the exact numerical target of the needle target position. Ideally, this project would use target numbers defined in 3D slicer, based on MRI images to support MoveIt path planning for later application on real patients.

An attempt was made to create a method using C++ to assign a numerical target

for the end effector in MoveIt2. However, given the relatively recent development of MoveIt2, a significant lack of detailed documentation online has been encountered. This shortage of resources impeded our ability to successfully develop the code, leading to a persistent error that remained unresolved for a month. The appendix details this incomplete code and the challenges encountered during its development.

Chapter 7

New research opening

7.1 Related to the prostate biopsy robot "Smart template"

7.1.1 Needle insertion simulation

The combined capabilities of Gazebo and MoveIt open the door to significant advances in medical robotics: a simulation of needle insertion and deflection customized to each patient's unique anatomy. This master's thesis only visually proves path planning for a single patient's data. There is a need to prove that the method works for a numerous amount of patients using more rigorous techniques than visual validation. As final step, testing on real patients would be the ultimate validation. Simulating tissue-needle interaction is a complex task, that is not fully correctly represented by this current work, due to lack of data like friction, and due to lack of mathematical work to correctly modelize this interaction.

7.2 For the improvement of robotics in the clinical field

7.2.1 Closed loop robot correction

When developing a new robot or introducing an additional component to an existing robot, simulation plays a pivotal role in ensuring functionality and compatibility without incurring the high costs associated with manufacturing physical parts. This approach is particularly beneficial in circumstances where a robot's design does not align with specific anatomical requirements of a patient. For example, this method could be applied to identify the limitations of using a breast biopsy device for prostate procedures. In fact, reliance on simulation could have significantly reduced expenses by preemptively identifying design mismatches. The process can be efficiently managed through a closed-loop correction system, where feedback from simulations guides iterative design improvements. In this context, the initial robot design is typically conceptualized and stored in a CAD (Computer-Aided Design) file. CAD files are versatile digital formats that encapsulate detailed design information, making them ideal for initial design stages. These files are not just visual representations but contain precise measurements, material specifications, and

other critical engineering data. One key strategy in this process involves extracting a URDF file from the CAD file. By conducting simulations in Gazebo with the URDF file, it is possible to identify and rectify any discrepancies or design issues. Subsequently, these corrections can be applied back to the original CAD design. This iterative process – testing in Gazebo, making adjustments in the CAD file, and repeating – ensures that the final robot design is not only theoretically sound but also practically viable in real-world scenarios. This method not only saves costs but also significantly accelerates the development cycle, ensuring that the final product is well-tuned to its intended operational environment.

7.2.2 Path planning for other organs

Biopsies often present a complex challenge due to the variety of potential paths to reach the target area. This complexity is compounded by the need to navigate around vital structures and blood vessels, making precise path planning essential. Utilizing MoveIt significantly simplifies this task. MoveIt leverages advanced algorithms to automatically calculate and optimize potential paths, ensuring the most efficient and safest route to the biopsy target is selected. This optimization process involves evaluating multiple factors, such as the shortest distance, minimal invasiveness, and avoidance of critical anatomical structures. By doing so, MoveIt facilitates easier and more accurate access to the target area, potentially reducing the procedure's duration and enhancing its overall safety. This technological advancement in robotic-assisted surgeries represents a significant leap forward in medical precision, particularly in complex procedures where accuracy is paramount.

7.2.3 Training for medical staff

Gazebo simulations provide a safe and controlled environment for medical professionals, especially surgeons, to practice complex procedures with robotic assistance. This virtual training ground is essential for helping physicians understand the nuances of robotic surgery and how the robotic equipment works, control systems, and possible reactions in different surgical scenarios [Araujo and Pêgo-Fernandes, 2023]. Such simulations allow repeated practice without posing risks to real patients, promoting skill development and confidence in the use of robotic systems. Additionally, if the robot has autonomous capabilities, Gazebo also provides a platform to test and refine these capabilities. This dual aspect of training for both medical staff and the robotic system itself ensures a comprehensive readiness strategy. Healthcare professionals can learn how to work effectively with autonomous systems and understand their limits and capabilities, while optimizing the robot's autonomous capabilities for better synergy with human operators.

Chapter 8

Conclusion

In this thesis, a simulation platform was developed, to simulate a prostate biopsy robot and the optimal path planning to avoid organs at risk based on patient's anatomy, using tools like ROS, RVIZ, MoveIt, and Gazebo. This platform is now available for further use in the development of safer robotic systems. By creating an URDF file for robot structure representation and integrating prostate models derived from MRI images, a comprehensive and realistic simulation environment was established. This simulation allows for precise path planning and the execution of complex movements, essential for targeting biopsy sites within the prostate. This thesis opens the door to new possibilities in medical robotics, promoting the use of virtual simulations in medical procedures. It emphasises the potential of medical robotics in enhancing the precision and safety of medical procedures, particularly in prostate cancer diagnostics. The implications of this research extend beyond the academic sphere, offering promising insights for less invasive and more precise procedures.
The exploration of new research opportunities, particularly in the application of needle insertion simulation, and liver biopsy path planning, marks a significant stride towards the continuous improvement of medical robots. Additionally, the potential of these simulations in training medical staff presents an invaluable tool in educating and preparing healthcare professionals for the technological advancements in medical procedures.

Chapter 9

Appendix

9.1 Motion Planning using C++

This Appendix presents a C++ code (see Fig 9.1) implementation for robot motion planning using ROS2 and MoveIt2, defining a numerical target point for the end effector. The code is supposed to find the optimal planning path, based on the numerical target defined for the end effector. This code initializes the ROS environment and creates a node named "motion_planning". Next, a logger is set up to record important messages during execution. The core functionality of the code includes interfacing with the MoveIt2 framework using the MoveGroupInterface class, specifically for the Main_Manipulator group (every part of the robot, excluding the end effector). Robot movements can be defined and controlled through this interface. Next, the code specifies the target pose of the robot's end effector. Poses defined based on position in 3D space of the end effector. Once a target pose is set, the code attempts to create a motion plan for that pose. The plan method of MoveGroupInterface generates a plan and evaluates whether the plan was successful. If successful, the plan is executed and the robot moves according to the defined pose. If the plan fails, an error message will be logged to the logger that was previously set up.

In this thesis, a challenge with C++ code for robot motion planning was encountered. Despite careful implementation, the code did not work as expected, and the underlying problem remained unclear because the command line errors did not provide a clear explanation. One possible cause suspected was that the target pose definition was incorrect, especially the initial value set for the target. These values may not be suitable for applications beyond standard robotic arms (the example always used on the internet). However, due to the lack of specific documentation and guidance regarding the range and type of values required for different robot structures, this remained an unconfirmed hypothesis. This situation highlights the gap in resources and documentation available for advanced applications of robot motion planning using ROS2 and MoveIt2, especially for non-traditional robot configurations.

```cpp
#include <memory>

#include <rclcpp/rclcpp.hpp>
#include <moveit/move_group_interface/move_group_interface.h>

int main(int argc, char * argv[])
{
  // Initialize ROS and create the Node
  rclcpp::init(argc, argv);
  auto const node = std::make_shared<rclcpp::Node>(
    "motion_planning",
    rclcpp::NodeOptions().automatically_declare_parameters_from_overrides(true)
  );

  // Create a ROS logger
  auto const logger = rclcpp::get_logger("motion_planning");

  // Create the MoveIt MoveGroup Interface
    using moveit::planning_interface::MoveGroupInterface;
    auto move_group_interface = MoveGroupInterface(node, "Main_Manipulator");

    // Set a target Pose
    auto const target_pose = []{
      geometry_msgs::msg::Pose msg;
      // Set the orientation (rotation) using quaternion values
      //msg.orientation.x = 1e-6;
      //msg.orientation.y = 1e-6;
      //msg.orientation.z = 1e-6;
      //msg.orientation.w = 1.000000;

      // Set the position (translation)
      msg.position.x = 1e-6;
      msg.position.y = 1e-6;
      msg.position.z = 1e-6;
      return msg;
    }();
    move_group_interface.setPoseTarget(target_pose);

    // Create a plan to that target pose
    auto const [success, plan] = [&move_group_interface]{
      moveit::planning_interface::MoveGroupInterface::Plan msg;
      auto const ok = static_cast<bool>(move_group_interface.plan(msg));
      return std::make_pair(ok, msg);
    }();

    // Execute the plan
    if(success) {
      move_group_interface.execute(plan);
    } else {
      RCLCPP_ERROR(logger, "Planning failed!");
    }

      // Shutdown ROS
      rclcpp::shutdown();
      return 0;
    }
```

Figure 9.1: C++ code used to define a numerical value for the target.

Bibliography

[noa, a] Create Realistic Robotics Simulations with ROS 2 MoveIt and NVIDIA Isaac Sim | NVIDIA Technical Blog.

[noa, b] Understanding nodes — ROS 2 Documentation: Humble documentation.

[noa, c] Understanding topics — ROS 2 Documentation: Humble documentation.

[fbg,] What is Fiber Bragg Grating ? – Fosco Connect.

[lap, 2023] (2023). Laparoscopy. Page Version ID: 1187491124.

[Araujo and Pêgo-Fernandes, 2023] Araujo, P. and Pêgo-Fernandes, P. (2023). Robotic surgery training.

[Artur et al., 2019] Artur, S., Hongbing, L., Natalia, S.-M., and Evgeni, M. (2019). (PDF) Extending Gazebo simulator for surgical robotics: tissue and suture modeling.

[Bernardes et al., 2023] Bernardes, M. C., Moreira, P., Mareschal, L., Tempany, C., Tuncali, K., Hata, N., and Tokuda, J. (2023). Data-driven adaptive needle insertion assist for transperineal prostate interventions. *Physics in Medicine & Biology*, 68(10):105016.

[Birtch, 2021] Birtch, N. (2021). MRI for Biopsy: MRI-US Fusion vs. In-Bore.

[CDCBreastCancer, 2022] CDCBreastCancer (2022). What Is Screening for Prostate Cancer?

[Chang et al., 2013] Chang, D. T. S., Challacombe, B., and Lawrentschuk, N. (2013). Transperineal biopsy of the prostate–is this the future? *Nature Reviews. Urology*, 10(12):690–702.

[Clare M C and MD Peter R, 2023] Clare M C, T. and MD Peter R, Carroll MD MPH Michael S, L. M. (2023). The role of magnetic resonance imaging in prostate cancer.

[Cochrane Miller,] Cochrane Miller, J. MR/US Fusion Imaging as an Aid to Prostate Biopsy. *Massachusets General Hospital*, 14.

[Daniel et al., 2018] Daniel, J., Charlotte, F., Andreas, D., Victoria, A., and Una, M. (2018). The diagnostic test accuracy of rectal examination for prostate cancer diagnosis in symptomatic patients: a systematic review - PMC.

[Dimmen et al., 2012] Dimmen, M., Vlatkovic, L., Hole, K.-H., Nesland, J. M., Brennhovd, B., and Axcrona, K. (2012). Transperineal prostate biopsy detects significant cancer in patients with elevated prostate-specific antigen (PSA) levels and previous negative transrectal biopsies. *BJU International*, 110(2b):E69–E75. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1464-410X.2011.10759.x.

[Fatschel et al.,] Fatschel, A., Li, M., Li, G., and Iordachita, I. Flexible Needle Shape Reconstruction based on FBG Sensors.

[Florin et al., 2015] Florin, G., Butnariu, S., Voinea, G., Bogdan, T., Girbacia, T., and Pisla, D. (2015). A Virtual Reality System for Pre-Planning of Robotic-Assisted Prostate Biopsy. *Applied Mechanics and Materials*, 772:585–590.

[josh, 2021] josh (2021). Getting Ready for ROS Part 4: ROS Overview (10 concepts you need to know).

[Kim et al., 2002] Kim, Y., Roscoe, J., and Morrow, G. (2002). The effects of information and negative affect on severity of side effects from radiation therapy for prostate cancer | Supportive Care in Cancer.

[Lakhal et al., 2023] Lakhal, O., Chettibi, T., Belarouci, A., Youcef-Toumi, K., and Merzouki, R. (2023). Optimisation of path planning for minimally invasive interventions on prostate using MR-robot: Application to on-live pets. *IFAC-PapersOnLine*, 56(2):11621–11626.

[Mareshal, 2022] Mareshal, L. (2022). System integration and pre-clinical evaluation of adaptive needle steering for image-guided prostate intervention.

[Masoom et al., 2022] Masoom, S. N., Sundaram, K. M., Ghanouni, P., Fütterer, J., Oto, A., Ayyagari, R., Sprenkle, P., Weinreb, J., and Arora, S. (2022). Real-Time MRI-Guided Prostate Interventions. *Cancers*, 14(8):1860.

[Moreira et al., 2021] Moreira, P., Grimble, J., Iftimia, N., Bay, C. P., Tuncali, K., Park, J., and Tokuda, J. (2021). In vivo evaluation of angulated needle-guide template for MRI-guided transperineal prostate biopsy. *Medical Physics*, 48(5):2553–2565.

[Newans, 2022a] Newans, J. (2022a). Making a Mobile Robot #12 - ros2_control Concept & Simulation | Articulated Robotics.

[Newans, 2022b] Newans, J. (2022b). Making a Mobile Robot #3 - Concept Design Gazebo | Articulated Robotics.

[Okoli et al., 2019] Okoli, F., Lang, Y., Kermorgant, O., and Caro, S. (2019). Cable-Driven Parallel Robot Simulation Using Gazebo and ROS. In *CISM International Centre for Mechanical Sciences, Courses and Lectures*, pages 288–295. Journal Abbreviation: CISM International Centre for Mechanical Sciences, Courses and Lectures.

[open robotics,] open robotics. Why ros 2.

[Penzkofer et al., 2015] Penzkofer, T., Tuncali, K., Fedorov, A., Song, S.-E., Tokuda, J., Fennessy, F. M., Vangel, M. G., Kibel, A. S., Mulkern, R. V., Wells, W. M., Hata, N., and Tempany, C. M. C. (2015). Transperineal In-Bore 3-T MR Imaging–guided Prostate Biopsy: A Prospective Clinical Observational Study. *Radiology*, 274(1):170–180.

[Robotics, a] Robotics, P. Concepts | MoveIt.

[Robotics, b] Robotics, P. MoveIt Setup Assistant — moveit_tutorials Noetic documentation.

[Robotics, 2019] Robotics, P. (2019). Developer concepts | MoveIt.

[Sang-Eun Song et al., 2013] Sang-Eun Song, Tokuda, J., Tuncali, K., Tempany, C. M., Zhang, E., and Hata, N. (2013). Development and Preliminary Evaluation of a Motorized Needle Guide Template for MRI-Guided Targeted Prostate Biopsy. *IEEE Transactions on Biomedical Engineering*, 60(11):3019–3027.

[SanthiElayaperumal, 2019] SanthiElayaperumal (2019). BendingSurgicalTool < Haptics < TWiki.

[Society, 2019] Society, A. C. (2019). Surgery for Prostate Cancer | Prostatectomy | American Cancer Society.

[Society, 2023] Society, A. C. (2023). Tests for Prostate Cancer | Prostate Cancer Diagnosis | American Cancer Society.

[Stephen W. et al., 2023] Stephen W., L., Taylor L., S.-S., Anu R, I., Hussain, S., and William P., S. (2023). Prostate Cancer - PubMed.

[VV et al., 2020] VV, P., LM, S., and OV, Z. (2020). RESEARCH OF MECHATRONICS OBJECTS for the fifth year students for second (master's) level applicants educational and professional program "Applied Mechanics" industry 13 Mechanical Engineering specialty 131Applied Mechanics full-time and correspondence forms of study. LUTSK NATIONAL TECHNICAL UNIVERSITY Technological faculty Department of Applied Mechanics.

[Yong Bai, 2014] Yong Bai, Q. B. (2014). Fiber optic monitoring system.

[Özer and Hasbay, 2020] Özer, C. and Hasbay, B. (2020). Concordance between needle biopsy and radical prostatectomy specimens in prostatic carcinoma. *Acta Oncologica Turcica*, 53(3):396–401.

UNIVERSITE CATHOLIQUE DE LOUVAIN

Faculty of Sciences

Place des sciences, 2 bte L6.06.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/sc**