Computer Integrated Interventional Systems Laboratory

Johns Hopkins University

Hongyi Fan

Bridging 3D Slicer to Simulation Software via ROS

Research Project Report

Table of Content

Background

One of the most commonly used tools for research and prototyping in medical imaging is 3D Slicer, which is an open-source platform. In the field of robotics, the standard development framework is the open-source middleware suite called the Robot Operating System (ROS). In the past, various attempts have been made to connect these two tools, but they all relied on middleware and custom interfaces. The fact that there is not yet a successful integration of the two tools brings difficulties to research and developments related to robotic intervention.

On the other hand, robotics simulation software like Asynchronous Multi-body Framework (AMBF) is a powerful robot simulation tool that is integrated with ROS.

In this project, we proposed to develop a ROS module for 3D Slicer which allows direct usage of ROS packages within the software and builds real-time communication with robotic simulation software.

Project Goals

1. To create a ROS module in 3D Slicer that is directly loadable by the software. The module should maintain a ROS node while running, and establish communications with the ROS core including subscribing, publishing the messages, and reading parameters from the ROS parameter server.

2. Establish communication between 3D Slicer and robotics simulation software via ROS. The selection of simulation software can be arbitrary as long as it uses ROS, preferably also Unified Robot Description Format (URDF). AMBF and RViz are chosen to be used for this project.

3. Given the published robot state in ROS, the 3D Slicer should be able to load robot link meshes and sync the robot to the simulation in real time.

Technical Approach

Architecture Overview

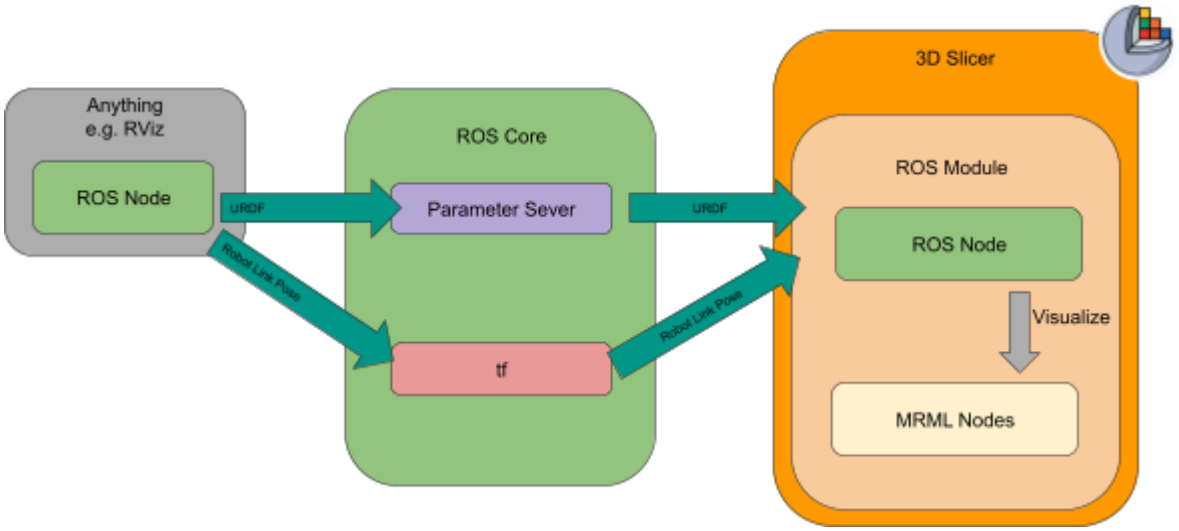

Figure 1: High-level structure of ROS Module with general simulators like RViz

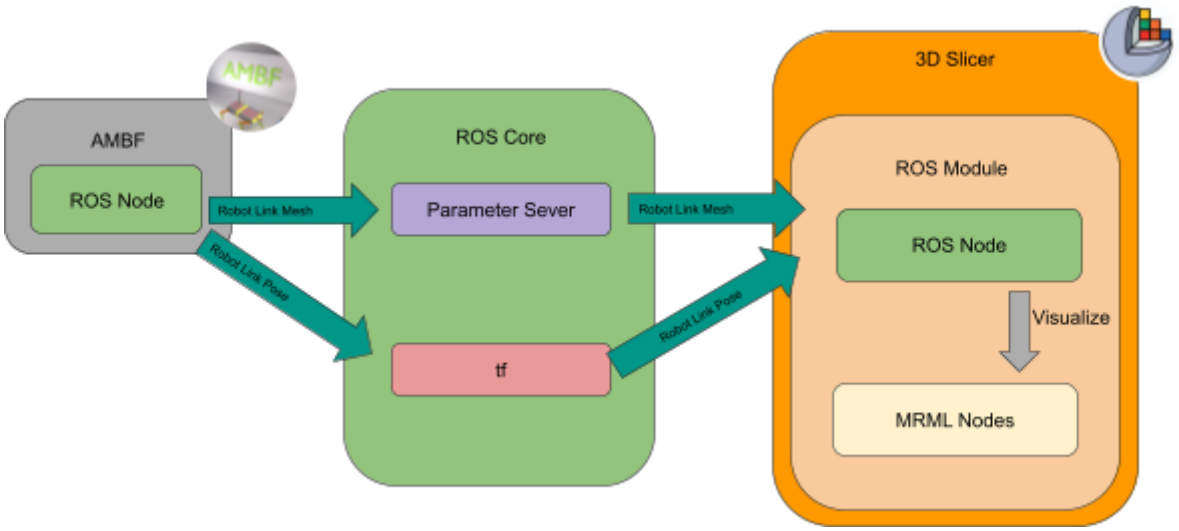

Figure 2: High-level structure of ROS Module with AMBF

For AMBF specifically, while the overall structure remains vastly unchanged, it does not provide robot descriptions in URDF format. Instead, we let AMBF publish the mesh file paths for each link to the parameter server, and the parameter is then parsed within the ROS module for mesh loading.

ROS Module in 3D Slicer

As mentioned in the previous section, the 3D Slicer platform offers interfaces that allow developers to create extensions and implement new features.

An extension module leaves several parts for the developer to implement. Mainly, the code is distributed over two spaces: Widget and Logic. The Module Widget manages the functionalities associated with GUI components and the Logic class maintains the processing of the module and its interaction with the MRML scene.

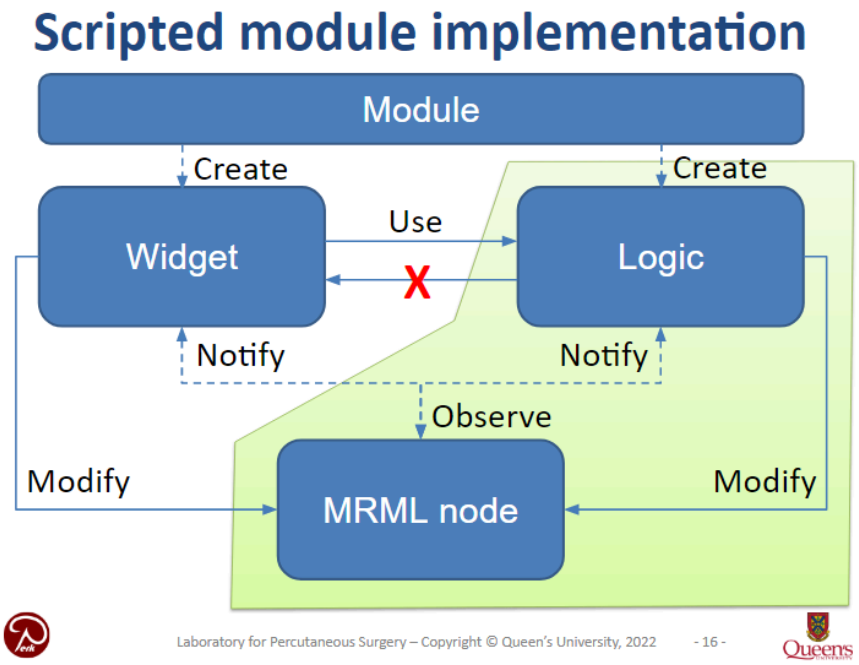

Figure 3: 3D Slice Module implementation provided by Perk Lab[2]

In our case, the functionality of real-time synchronization is achieved by setting up a timer at 200 Hz inside the module Wiget. Each timer timeout event triggers an update to the robot model inside the MRML scene.

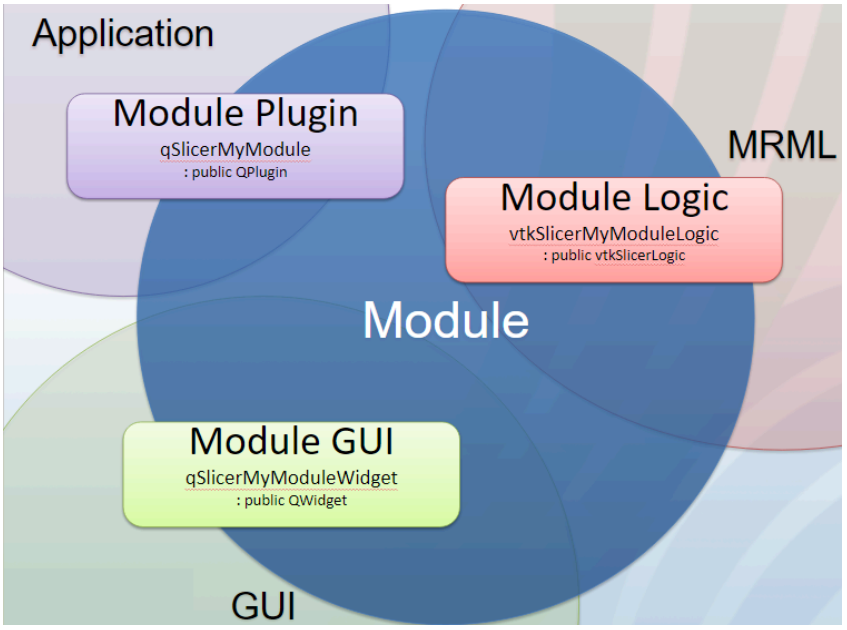

Figure 4: Module components diagram provided by Kitware[3]

In 3D Slicer, the environment is called an MRML scene. MRML stands for Medical Reality Markup Language. The data and elements in the scene are managed in form of MRML nodes.

The goal of our work is to provide real-time visualization of a robot. To achieve this, the module parses the robot's description from the ROS parameter server and creates an MRML model node for each link using the link's mesh file. An MRML display node is then added for each model node to visualize them in the scene. In order to synchronize the robot's motion, the module creates an MRML transform node for each model node and subscribes to the tf topic to receive updates on the robot's link poses. Whenever there is a new update on the tf topic, the module parses the tf transforms and updates the corresponding MRML transform nodes accordingly.

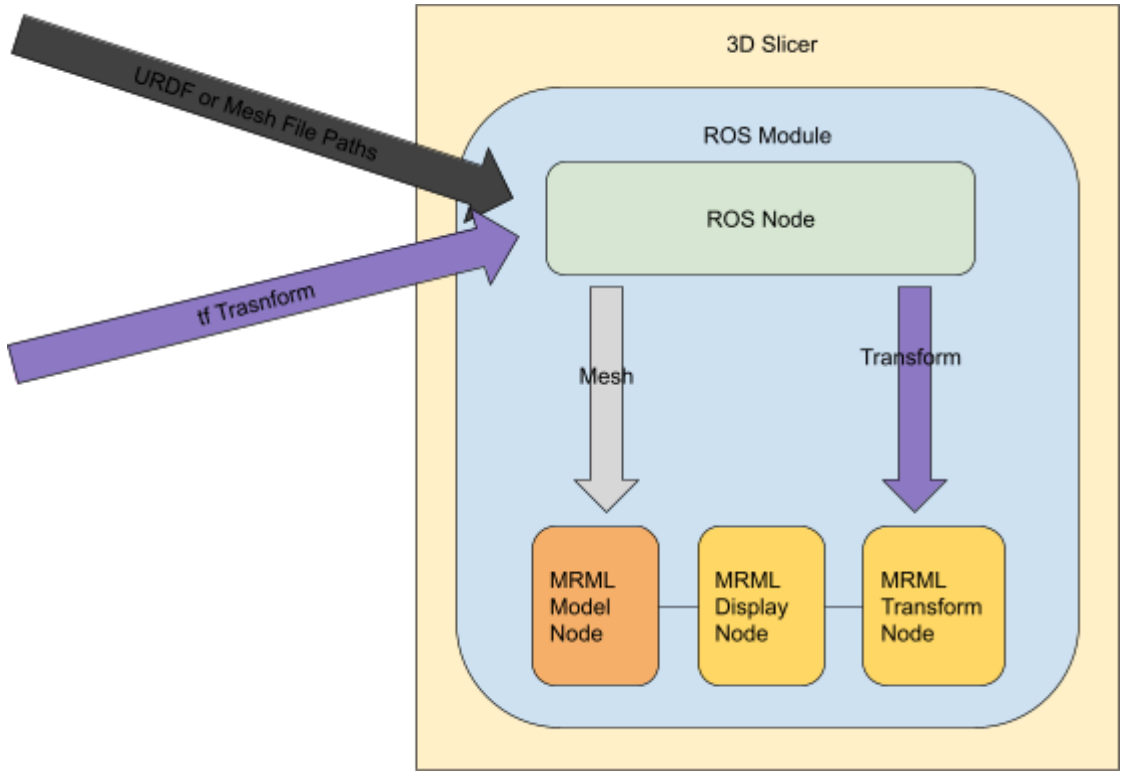

Figure 5: Lower level structure of ROS Module in 3D Slicer

Simulator

RViz

RViz is a tool that enables users to display and interact with robot data in three dimensions using the Robot Operating System (ROS). It can be used to visualize various types of information, such as point clouds, laser scans, and geometric data. Robot developers and researchers often use RViz to test and analyze their algorithms and systems.

In this project, we use RViz to confirm the ability to connect simulators and 3D Slicer through ROS. We load a UR5 robot in RViz, which automatically publishes the robot's URDF file to the ROS parameter server. When we launch RViz with a UR5 simulation, a joint controller window appears, allowing the user to manipulate the robot's joint angles. The pose of each link is then published to the tf topic.

With the default setup, the ROS Module in 3D Slicer can synchronize the robot.

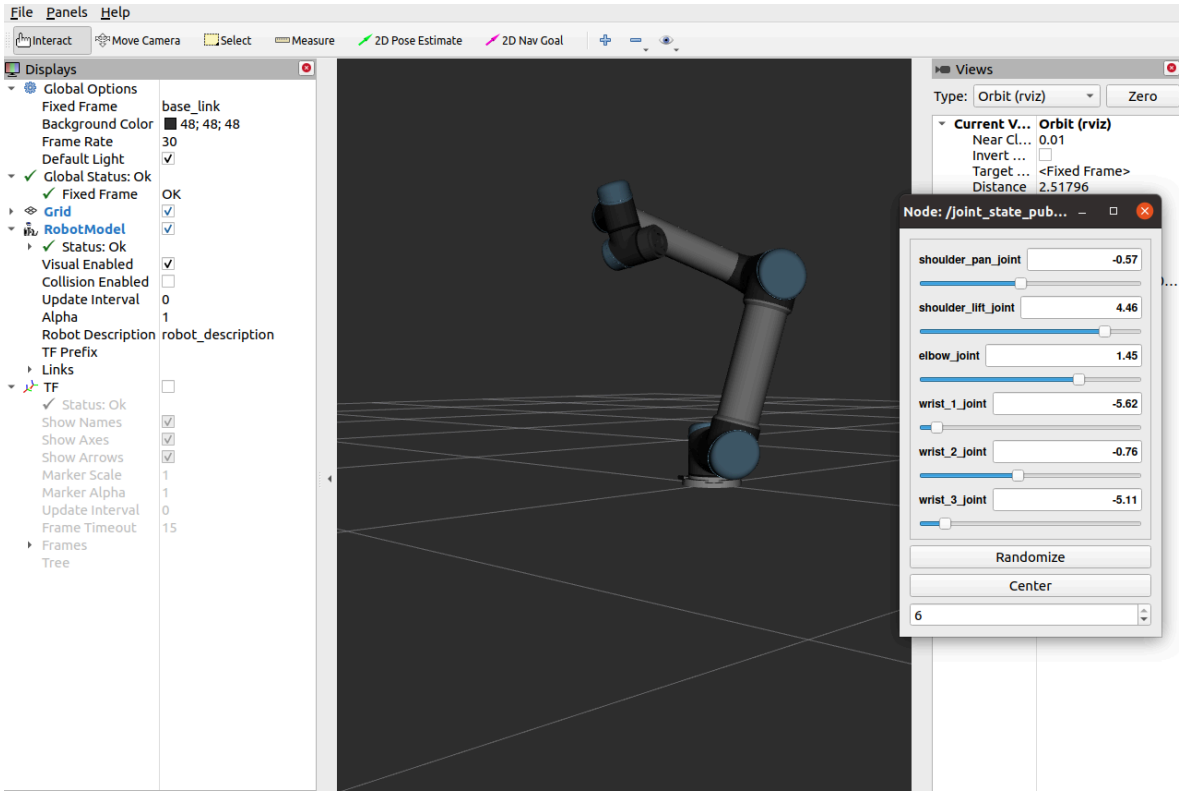

Figure 6: UR5 Simulation with robot state publisher

AMBF

AMBF is an open-source, 3D versatile simulator for robots that was developed in 2019. This multi-body framework provides real-time dynamic simulation of multiple bodies such as robots, free bodies, and multi-link puzzles, along with real-time haptic interaction with various input devices. The framework also integrates a real surgeon's console, whether haptic or not, to control simulated robots in real time.

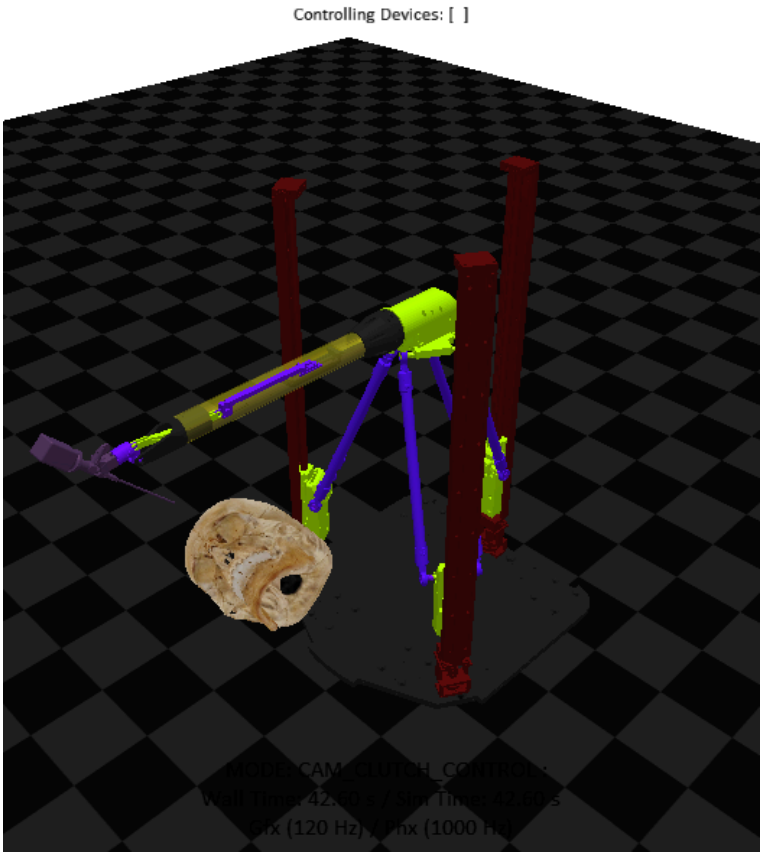

Figure 7: AMBF Simulator with Galen surgical system

In this project, we achieve the synchronization of the Galen Surgical Robot between the AMBF simulator and 3D Slicer.

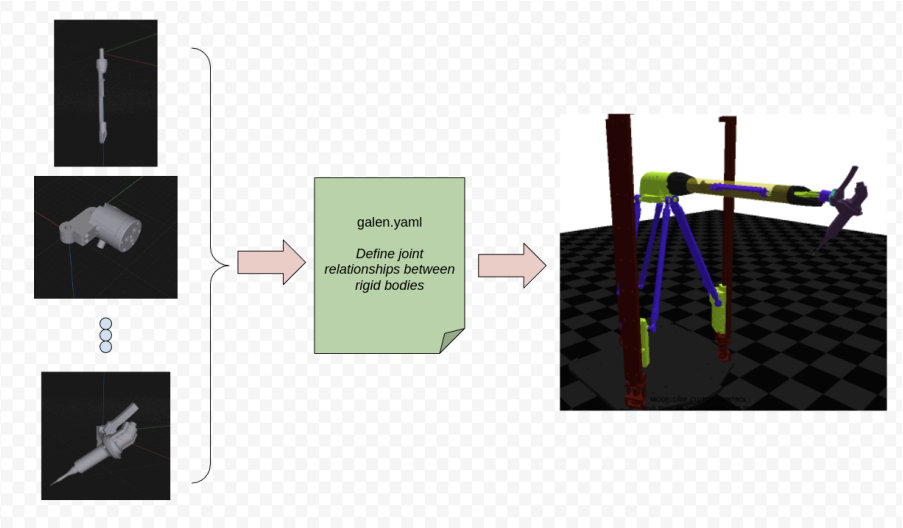

Figure 8: ADF defines Galen Surgical Robot using individual components

AMBF allows developers to write plugins to achieve custom functionalities.

Differing from the previous simulator, AMBF does not use URDF for robot description. It uses a YAML-based description file called ADF. To address this difference, the AMBF plugin designed for this project publishes the robot's link and the path to their mesh files in a special format to the ROS parameter server.

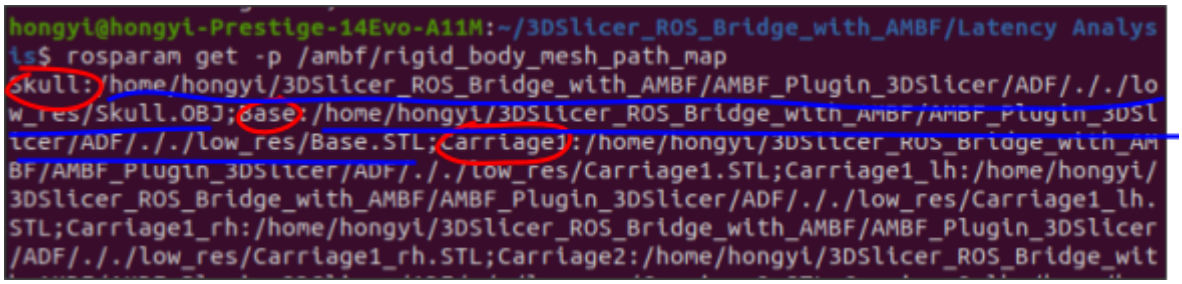

Figure 9: AMBF publishes the mesh file path for each rigid body, separated by semicolons

The ROS module exclusively searches and parses the robot mesh paths parameter from the ROS parameter server. The parameter only contains the name of the link and its mesh file path because AMBF automatically handles the joint offsets before querying its pose. On the contrary, ROS Module has to parse and apply the offset before updating the pose of the links.

The AMBF simulator plugin inquires about the pose of each rigid body in the simulation world and publishes them to the tf topic.

Results

The ROS Module in 3D Slicer was able to synchronize with both RViz and AMBF free of error, and without noticeable latency.

Robot Synchronization with RViz

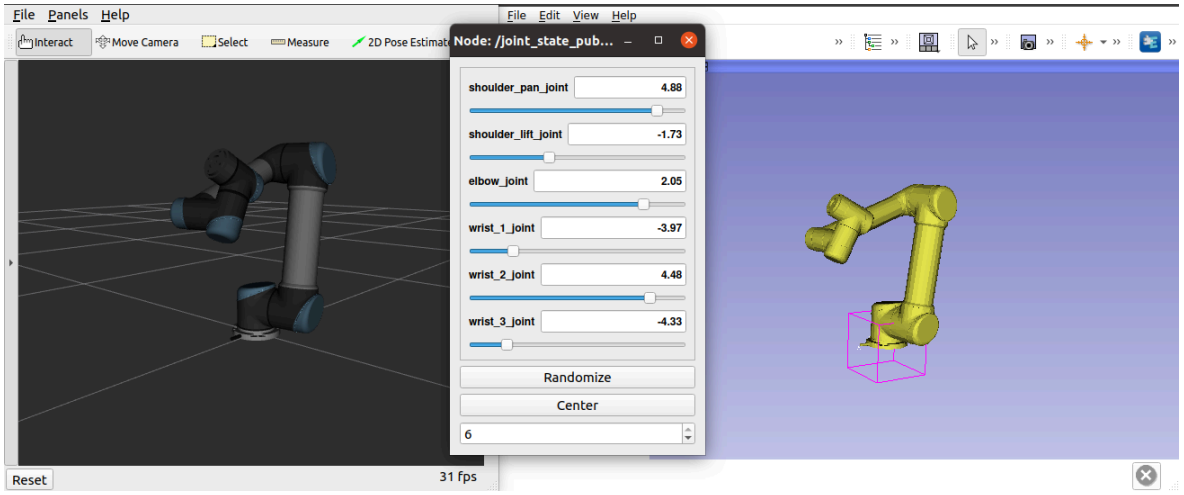

Figure 10: Robot synchronization with RViz

Robot Synchronization with AMBF

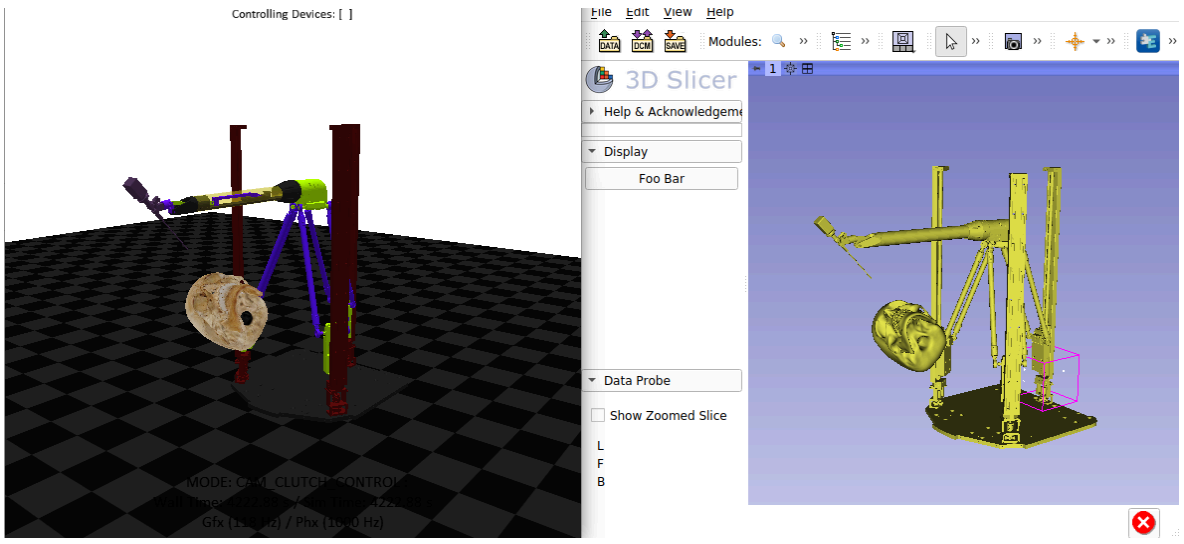

Figure 11: Galen Robot synchronization with AMBF

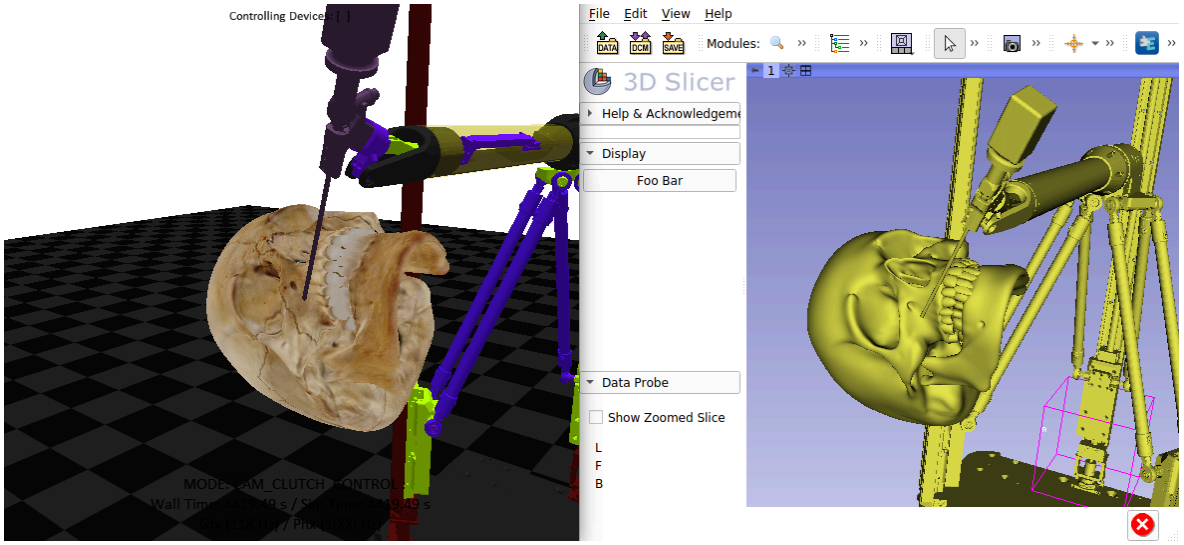

Figure 12: Galen Robot synchronization with AMBF. Note the dynamics feature in AMBF is preventing the tool from passing through other rigid bodies.

Performance Evaluation

In order to assess the performance of the system, I configured the simulator to include the AMBF software and a Galen Surgical Robot model, which consists of 25 rigid bodies and one skull model. The 3D Slicer ROS module updates at a maximum rate of 200Hz. During the evaluation, I manually generated motion in AMBF by manually manipulating the robot, and for each update, I recorded the time difference between the timestamp of the first rigid body's TF transforms and the timestamp of the end of the robot model update in 3D Slicer.

The evaluated result is as shown below:

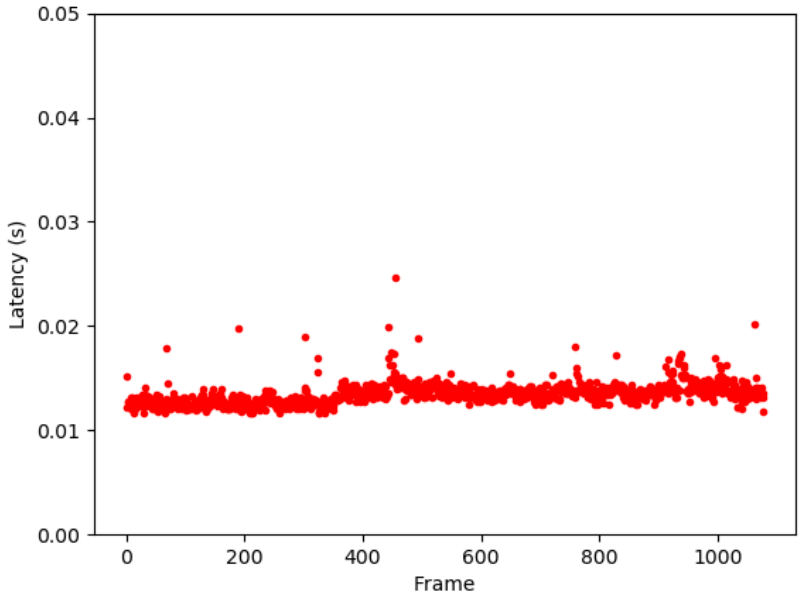

Figure 13: Latency vs Frames plot

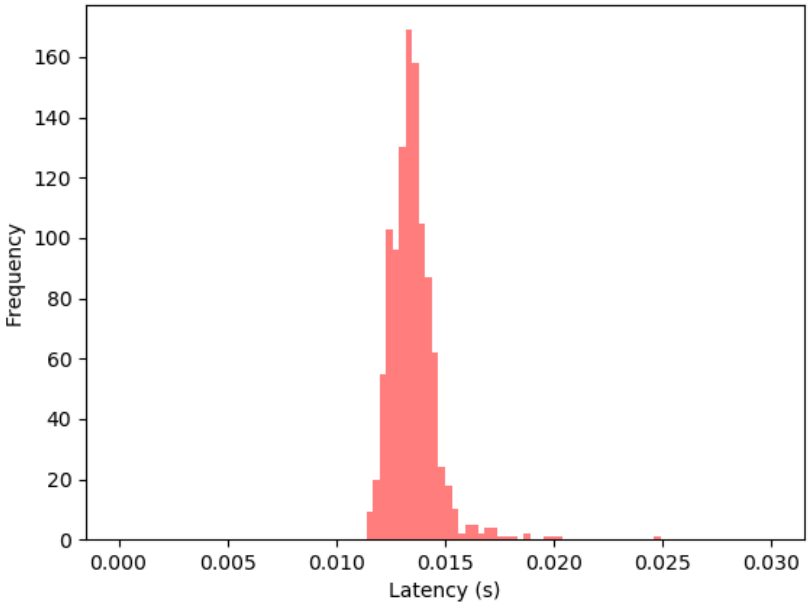

Figure 13: Latency histogram

Statistics (Unit - Second):

Mean: 0.0136355261192251

Median: 0.013400077819824219

std: 0.0043685298720561976

Since average delays in TF are < 1ms. We can say that the source of delay is mainly from robot model updates in 3D Slicer.

The statistics show that delay of the ROS module is stable at around 13ms.

Significance and Conclusion

In this project, I developed a ROS module for 3D Slicer that includes a ROS node. The module is able to parse URDFs from the ROS parameter server, load meshes for each link, and reconstruct the robot within the 3D Slicer scene. When working with a simulator like RViz, the module is able to synchronize the robot's motion in real-time. Additionally, the module also works with AMBF. It reads the path of the mesh file for each link from the parameter server and updates the robot's motion based on the rigid body poses in the AMBF simulator. Overall, this ROS module effectively bridges 3D Slicer with simulators and allows for real-time synchronization of robot motion. It is an important contribution to the field of robotic intervention research and development that utilizes 3D Slicer.

There is space for improvement. At the current stage, the communication is One-Way from the simulator/robot to 3D Slicer. 3D Slicer, as a powerful tool that is capable of Registration, Segmentation, etc., does not provide any input or feedback for the virtual context. In the future development, having 3D Slicer as a source of input back to the simulator is an important direction.

Management Summary

Source Code

The source code for this project is controlled using GitHub. Resources along with the source code are located in a repository under the CIIS-Lab organization.

Link to the GitHub repository:

https://github.com/LCSR-CIIS/3D-Slicer_ROS_Module_with_AMBF

Documentation

The instruction for installing and building this project is located in the GitHub repository in the form of README markdown language.

The conclusive report (this document) is saved in the OneDrive folder.

Acknowledgment

Thank you to Dr. Adnan Munawar, Dr. Manish Sahu, and Prof. Russell Taylor for providing weekly feedback and guidance throughout the project.

Reference

[1] Connolly, Laura et al. “Bridging 3D Slicer and ROS2 for Image-Guided Robotic Interventions.” Sensors (Basel, Switzerland) vol. 22,14 5336. 17 Jul. 2022, doi:10.3390/s22145336

[2]PerkLab. “PerkLab/Perklabbootcamp: Materials for the Yearly Perklab Bootcamp Course.” *GitHub*, https://github.com/PerkLab/PerkLabBootcamp.git.

[3]Finet, Julien. “Loadable Modules.” *Kitware*, 1 Apr. 2019, https://www.slideserve.com/eytan/loadable-modules-powerpoint-ppt-presentation.

[4]Hongyi, Fan et al. “VR (Virtual Reality) Guided Skull-Base Surgery ”, 19 Apr. 2022, https://livejohnshopkins.sharepoint.com/:w:/s/CIS2-VRGuidedSkullBasedSurgery-RobotControlwithSimulationint/EeFmaBGXAnBIgUvAEsxJ7-EBHS3ugMrzRG3yTO4KrkwRVA?e=NZ9A5p