Individual Lab Report - 5

Autonomous Reaming for Total Hip Replacement

HIPSTER | ARTHuR

Gunjan Sethi

Team C:

Kaushik Balasundar | Parker Hill | Anthony Kyu
Sundaram Seivur | Gunjan Sethi

April 7 2022

Carnegie Mellon University
The Robotics Institute

Contents

1 Individual Progress

1.1 Robust Landmark Collection

The landmark collection pipeline from last progress review was further developed to bring its functionality close to the SVD test requirements, increase robustness and integrate with the ICP registration pipeline.

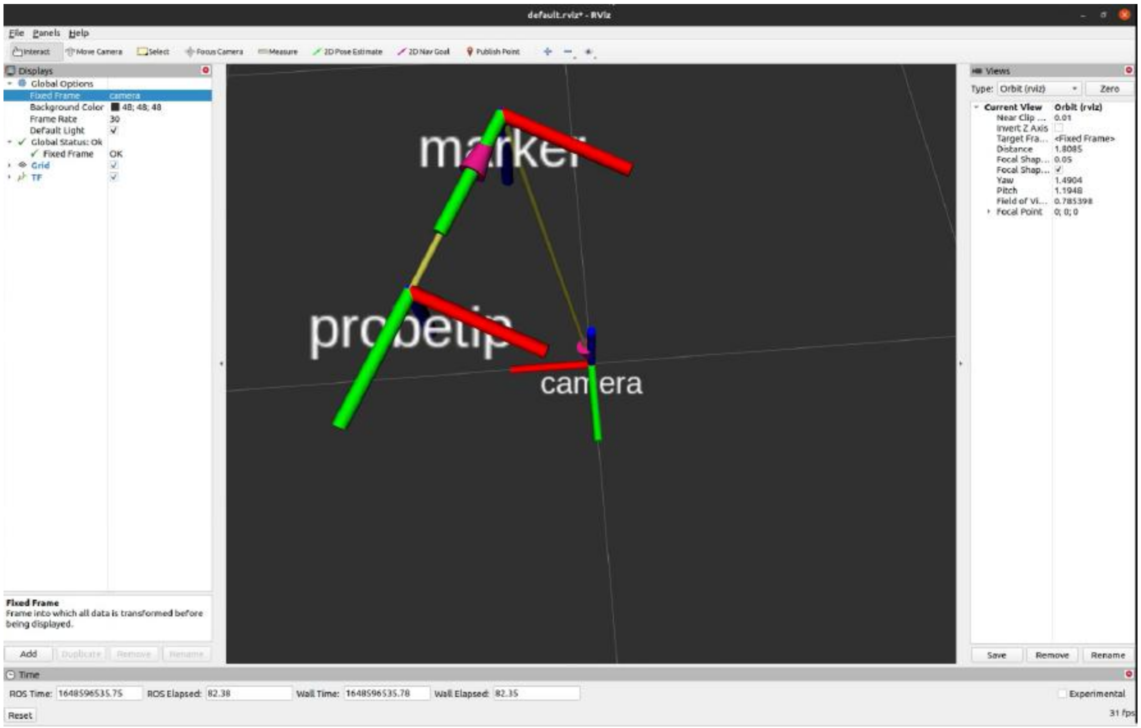
1.1.1 Integrating New Registration Probe

We received a new registration probe from the sponsors, along with its data sheet and CAD file. The probe is shown in Figure 1. After expressing concerns on obtaining the exact probe tip pose from the marker geometry, it was discovered that the specific transformation matrices that encode motion between various parts of the registration probe are present in the data sheet of the probe.
Once this registration probe was procured, the currently used probe was replaced. A new `.ini` file was generated keeping the data sheet as reference. After generating the new geometry file, it was initially verified with the Atracsys GUI and then loaded into the ROS Node. Both verification tasks were successful. The GUI was able to track the new marker geometry on the probe. The ROS Node was also able to track and publish the pose of the marker in realtime.

Figure 1: New Registration Probe

1.1.2 Marker to Probe Tip Transformation

The registration probe data sheet consisted comprehensive information about the various transformation matrices within the probe body. Taking the same as reference, these transformations were applied to the detected marker pose in order to translate to the tip of the probe that touches the pelvis. This task ensured that inspite of the several orientation probabilities of holding the registration probe, the system reliably collects the same point from the probe tip.
To verify this, the three frames- camera, marker and probe tip were visualized on RViz. Kaushik held the probe at various orientations and the probe tip point, when not translated on the pelvis, always remained constant. Figure 2 shows the visualization of the three frames on RViz.



Global Options
Fixed Frame: camera
Background Color: 48; 48; 48
Frame Rate: 30
Default Light: ✓
Global Status: Ok
Fixed Frame: OK
Grid: ✓
TF: ✓

Fixed Frame
Frame into which all data is transformed before being displayed.

Type: Orbit (rviz)
Current View: Orbit (rviz)
Near Clip ...: 0.01
Invert Z Axis: ☐
Target Fra...: <Fixed Frame>
Distance: 1.8085
Focal Shap...: 0.05
Focal Shap...: ✓
Yaw: 1.4904
Pitch: 1.1948
Field of Vi...: 0.785398
Focal Point: 0; 0; 0

ROS Time: 1648596535.75
ROS Elapsed: 82.38
Wall Time: 1648596535.78
Wall Elapsed: 82.35

Figure 2: Camera, Marker and Probe-tip Visualized on RViz

1.1.3 User Input-Based Sparse Pointcloud Collection

Once the new registration probe was integrated and the probe-tip pose was obtainable, tests on pointcloud collection were performed. During the initial calibration phase, the surgeon will not continuously collect points. The registration algorithm may require only 5-10 points to be collected as an initial guess. For this purpose, ”Hit Enter to Collect” functionality was implemented. The user can place the probe at a desired point, hit ”Enter” and a point at that position is recorded.

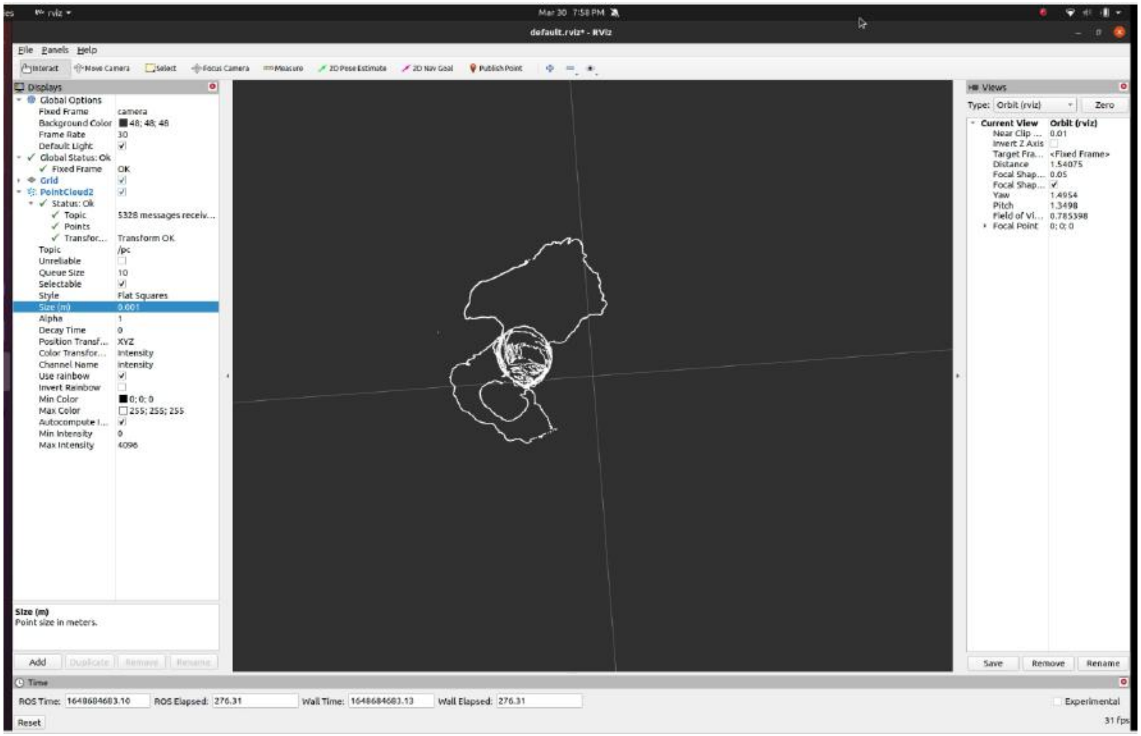
1.1.4 Continuous Dense Pointcloud Collection

After the initial guess, the surgeon will collect a dense pointcloud with sometimes, over a thousand points. For this, continuous dense pointcloud collection was implemented. The current tests were performed by collecting 6000 points at 30 FPS. The results of the tests are shown in Figure 3. Here one has collected the points from all over the acetabulum and the outline of the pelvis.

Pointcloud collection soon turned into a draw-in-the-air/3D art activity and Figure 4 shows off some of the fun pointcloud we collected.

1.2 Integration with Registration Pipeline

The collected pointcloud was saved as a `.pcd` file and handed over to Kaushik for testing with the ICP-based pointcloud registration pipeline. The results of the same are outlined in his report.



Global Options	
Fixed Frame	camera
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	☑
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	☑
PointCloud2	☑
✓ Status: Ok	
✓ Topic	5328 messages receiv…
✓ Points	
✓ Transfor…	Transform OK
Topic	/pc
Unreliable	☐
Queue Size	10
Selectable	☑
Style	Flat Squares
Size (m)	0.001
Alpha	1
Decay Time	0
Position Transf…	XYZ
Color Transfor…	Intensity
Channel Name	intensity
Use rainbow	☑
Invert Rainbow	☐
Min Color	■ 0; 0; 0
Max Color	☐ 255; 255; 255
Autocompute I…	☑
Min Intensity	0
Max Intensity	4096

Size (m)
Point size in meters.

Current View	**Orbit (rviz)**
Near Clip …	0.01
Invert Z Axis	☐
Target Fra…	<Fixed Frame>
Distance	1.54075
Focal Shap…	0.05
Focal Shap…	☑
Yaw	1.4954
Pitch	1.3498
Field of Vi…	0.785398
Focal Point	0; 0; 0

ROS Time: 1648684683.10 ROS Elapsed: 276.31 Wall Time: 1648684683.13 Wall Elapsed: 276.31

Figure 3: Dense Pointcloud Collection: 5328 points/30 FPS

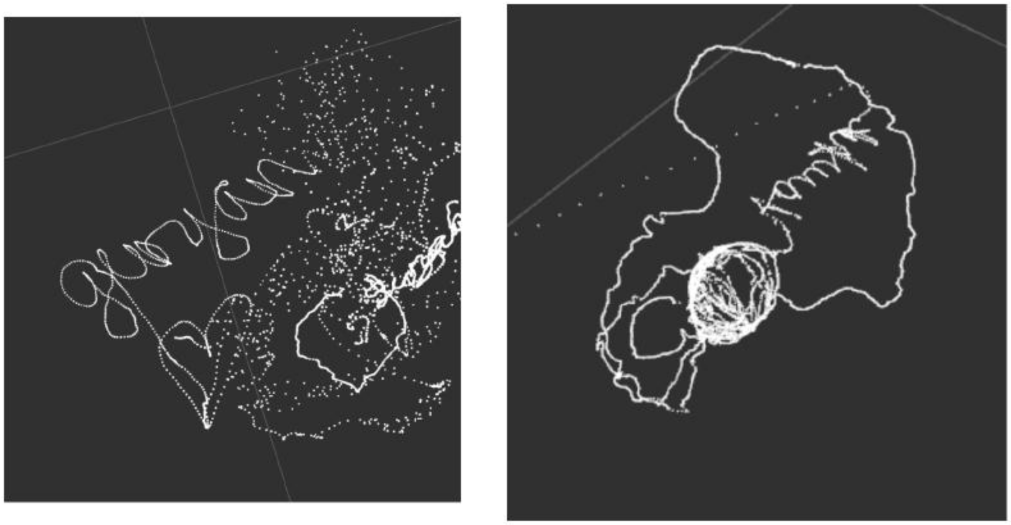


Figure 4: Pointcloud Art (Left) ”Gunjan” written amongst confetti. (Right) Pelvis with ”Kaushik” engraved

1.3 Detecting Multiple Markers

In the full system setup, the camera must be capable of tracking the marker on the pelvis, registration probe and robot arm end-effector. Therefore, the perception pipeline must handle these multiple marker geometries frames in parallel without introducing latency into the system. To test the same, a proof-of-concept ROS node `MultiFrameHandler` was developed that builds on top of the current `FrameHandler` node. The current node handles only one frame at this time. The new node was loaded with the geometry files of both the registration probe and the pelvis marker. It was successfully able track and publish both 6DOF poses without introducing latency at 52FPS.

1.4 Software Architecture Design: Perception and Sensing

Since most of the initial development of the perception and sensing subsystem is complete, the team looks forward to formalizing the code by improving its style, maintainability and scalability. To do this, in collaboration with Kaushik, a software architecture plan was developed that outlines the various functions, classes and how these interact with each other temporally.
A rough sketch of our ideas is presented in Figure 5. Although this is important, it is currently not our primary focus to refactoring the code. We will perform this code refactoring over the Fall months.

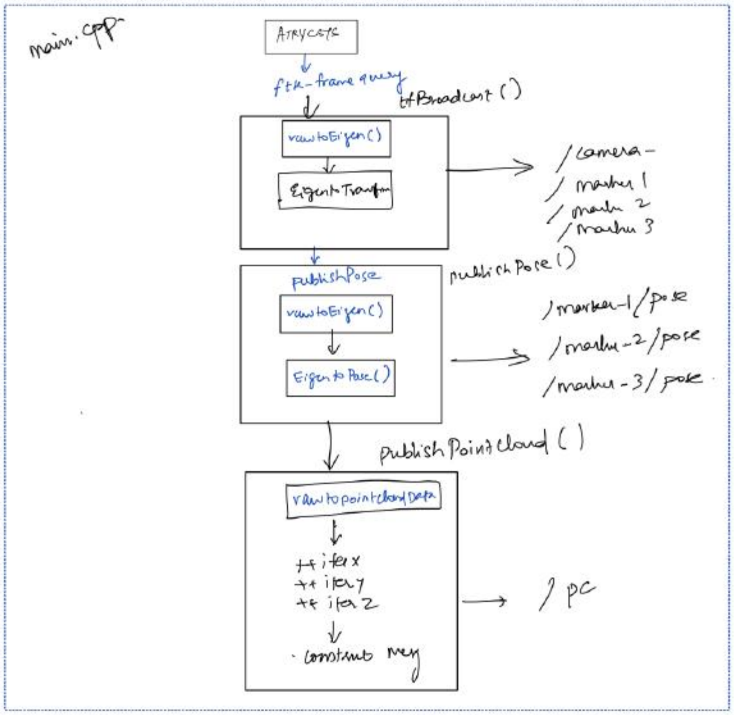


Figure 5: Rough Sketch of Percpetion and Sensing Subsystem Software Architecture

2 Challenges

2.1 Workspace Arrangements

The biggest challenge so far has been to arrange the workspace/demo space in a manner that complies with all software and hardware tests. Several times, the camera loses visibility of the markers and is unable to track them. Moreover, the placement of these markers in a way that does not interfere with each other's geometries is crucial. All these placements must also comply with the robot arm.

2.2 Hyperparameter Tuning for Pointcloud Collection

The pointcloud collection functionality has been implemented. However, we are still speculating on the best frequencies and number of points for the task of registration. While 6000 points/30FPS works well for a dense pointcloud, the registration pipeline might not require this level of density. And since the pointcloud collection is a one-time calibration process, it must not be too time-consuming. To tackle this issue, we will perform several tests and document them. We will pick the hyperparameters that increase surgeon experience while also giving accurate and timely registration results.

3 Team Work

Following are the tasks accomplished by the team members since the previous ILR.

- ***Kaushik Balasundar*** worked with Gunjan to obtain the registration marker's tip pose using the market geometry and the pose of the probe center obtained from the camera. He then worked with her to obtain the pointcloud of the pelvis using this probe, and drafted a software architecture for the perception sub-system. After this, he worked on using the acquired pointcloud to develop a method to obtain the initial guess for registration, and further refined the pipeline to register the pointcloud of the acetabulum with the 3D CAD model of the pelvis. The registration was then evaluated quantitatively. He also assisted Sundaram and Anthony in the integration of planning and controls sub-systems by writing the effort commander interface.

- ***Parker Hill*** redesigned the end-effector reamer to be shorter and more robust than his initial prototype. These designs were then 3D-printed and attached to the arm, allowing for the hardware system to be finalized for the Spring Validation Demonstration. He also worked on the motor control PCB, ordering all the parts, soldering them to the PCB, and testing the PCB for efficacy. While the PCB is not finalized, he was able to extend the motor wires and connect them to a power supply to actuate the reamer head, allowing for testing of the hardware system to be conducted on the sawbone pelvis.

- ***Anthony Kyu*** worked on transferring the MPC code from offline simulation to online real-time simulation in Gazebo. This task involved writing MPC update functions as well as restructuring the code to be modular and more efficient. He then integrated the MPC code into ROS using RobotOS.jl, writing an MPC solver node to solve the optimal control problem, a simulation node for internal testing before integration, and a controller node to send torques to the effort controllers. Collaborating with Sundaram and Kaushik, Anthony then integrated the controller and MPC nodes with the trajectory planning nodes, Gazebo sensor nodes, and the Gazebo effort controllers, enabling real-time simulation testing of the controllers in Gazebo. Because the MPC had trouble performing well in real-time, Anthony also developed a PD Impedance Tracking controller in parallel to use as a fallback should the MPC not be viable by the SVD. Anthony also collaborated with Parker, providing tips on how to set up and wire the power distribution system.

- ***Sundaram Seivur*** worked on completing the motion planning pipeline for which he compiled a custom message type on ROS. In this message he compiled trajectory messages which stored the joint states, a pose array message which stored the Cartesian positions and orientations, and a point array message which stored the cartesian velocity. He wrote a node to plan a trajectory using the Pilz motion planning pipeline and publish the generated trajectory to a topic. He collaborated with Anthony and Kaushik for integrating the planning and controls subsystems. He also collaborated with Parker to ideate hardware designs and their interaction.

- ***Gunjan Sethi*** worked on extending the functionality of the perception pipeline and testing for robustness and reliability. She worked closely with Kaushik to integrate the new registration probe into the current pointcloud collection pipeline, obtain the probe tip pose and publish pointclouds at several frequencies. She developed user-input based sparse pointcloud collection and continuous dense pointcloud collection. Both functionalities were thoroughly tested. The point cloud collection pipeline was integrated and tested with the registration pipeline with the help of Kaushik. Further, she wrote test scripts to track multiple marker geometries as a proof-of-concept. Kaushik and Gunjan also discussed a revised software architecture for the perception and sensing subsystem.

4 Plans

For future work, the following (individual) tasks have been planned for the MRSD project.

4.1 Pelvis Error Detection

The perception subsystem must detect error in pelvis pose (as the patient moves) and indicate the same to other subsystems. This task will involve building on top of the multi-frame handler node and constantly checking the pelvis marker pose to detect if the change has hit a threshold.

4.2 Integration with Planning Subsystem

Major efforts will be steered towards integrating the planning and controls subsystems with the perception and sensing. This will allow the team to showcase a complete pipeline of the system to some extent. The integration of the subsystems will be based on writing ROS topics and publisher/subscribers to communicate between various nodes. The same will be tested and evaluated based on the SVD metrics.

4.3 SVD Test Preparation + Testing

The next two weeks will involve stress testing all the components and ensuring the tests are well-rehearsed. Some variability in the tests will be introduced to ensure user engagement.