CAoCM

SS24

Semiautonomous Telerobotic Auscultation with Surface Normal Estimation

X

Group 3

Team member

- Delun Zhang
- Jeff Lin
- Yuya Yuan
- Zikang Liu

Supervisor

- Sven Kolb

Content

Idea

We want to address the shortage in care professions worldwide with the use of a semi-autonomous robotic arm. This technology enables precise remote diagnostic procedures from arbitrary locations. On the patient side, integrating multi-sensors on the robot ensures and improves patient safety. On the clinician side, sharing the diagnostic workflow with external experts facilitates better collaboration and expertise sharing.

Achievements

We developed a robotic diagnostic system integrated with auscultation device and investigated the accuracy of surface normal estimation methods.

Workflow

In this part, we are going to introduce how to implement this robot arms.

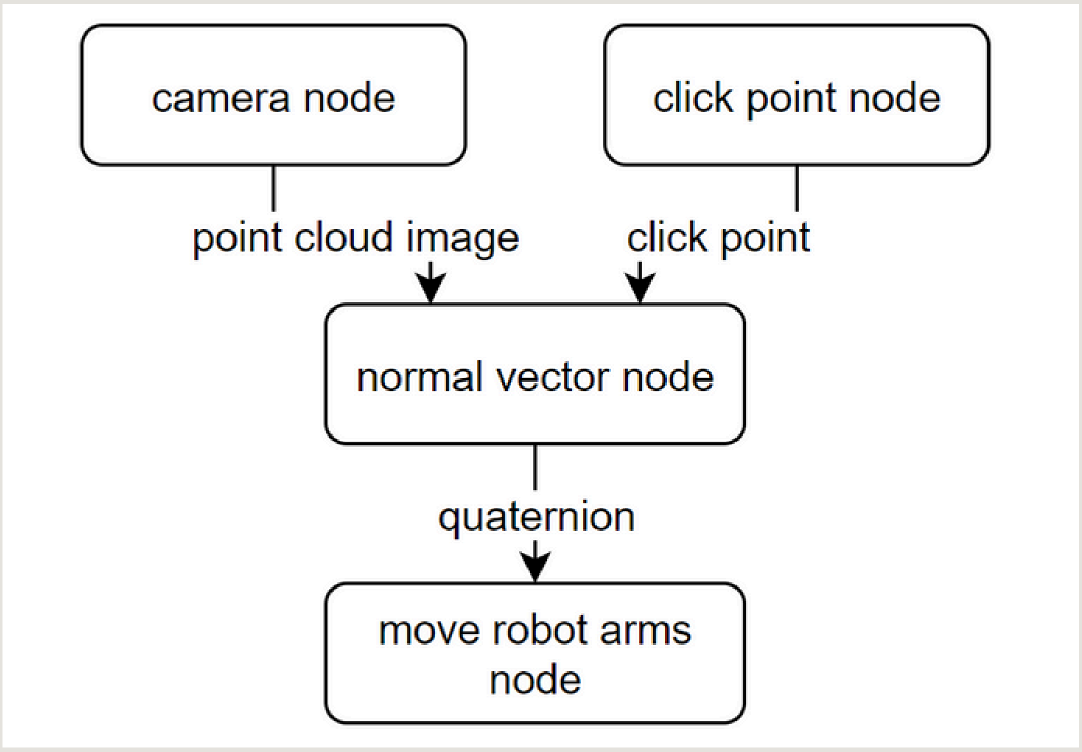
First, we use stickers to mark the point we want to test on the surface. By doing this, we can easily evaluate our algorithm at the same point.

After clicking the point you want in the monitor, the code will calculate its normal vector and pass it to robotic arms.

Robotic arms will then move to the clicked point and make sure to be perpendicular to the point.

Pipeline

In this part, we are going to explain how our code works. First, the camera node will take images of the patient continuously, and send the point cloud images to the normal vector node. After receiving normal vector nodes, the node will then wait for clicked point coordinates. After receiving both requried data, the normal vector node will calculate the desired point, which is the point closest to the clicked point. The reason why we are doing this step is because we have done the downsampling, which will eliminate the existing point. After that, we calculate the normal vector of the desired point and send it to the move robotic arm node. The robotic arms will move to the clicked point and with the direction of normal vector.



Mathmatical Methods

The process of estimating surface normal vectors from point cloud data can be divided into two main steps.

Step 1: Neighborhood Search. For the clicked point, Pi, we need to identify its neighboring points. This can be achieved through either fixed radius search or k-nearest neighbors search.

Step 2: We calculate the covariance matrix of these neighboring points and perform eigenvalue decomposition. The eigenvector corresponding to the smallest eigenvalue, indicates the direction with the least variance in the data. This vector is the normal vector.

$$\mathcal{C} = \frac{1}{k}\sum_{i=1}^{k} \cdot (\boldsymbol{p}_i - \overline{\boldsymbol{p}}) \cdot (\boldsymbol{p}_i - \overline{\boldsymbol{p}})^T,\ \mathcal{C} \cdot \vec{\mathrm{v}_j} = \lambda_j \cdot \vec{\mathrm{v}_j},\ j \in \{0, 1, 2\}$$

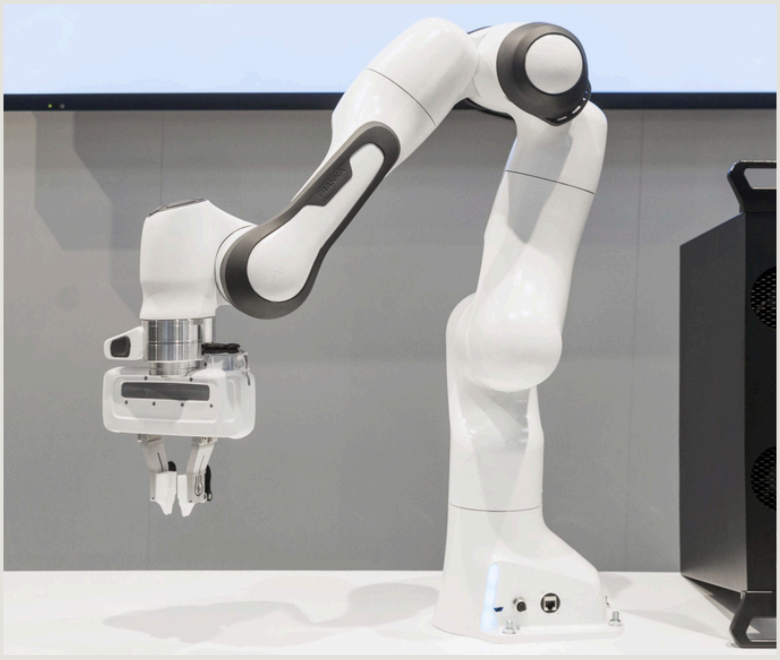
Formula to calculate covariance matrix C

Evaluation

To evaluate our algorithm, we first select a flat plane or human models and mark several check points on each surface.Then we manually manipulate the actuator to align the probe with the check points and record the probe's end position. And we also use our algorithm to predict the placement of the probe and control the robotic arm accordingly to get the probe's end position data and repeat several times. After we get the data, we can first calculate the standard deviation to assess the algorithm’s variability. Second, the error percentage to evaluate the error size of the algorithmse. Besides, we can use t-tests to find whether the surface type and camera's perspective affects the variability and performance of our algorithm.

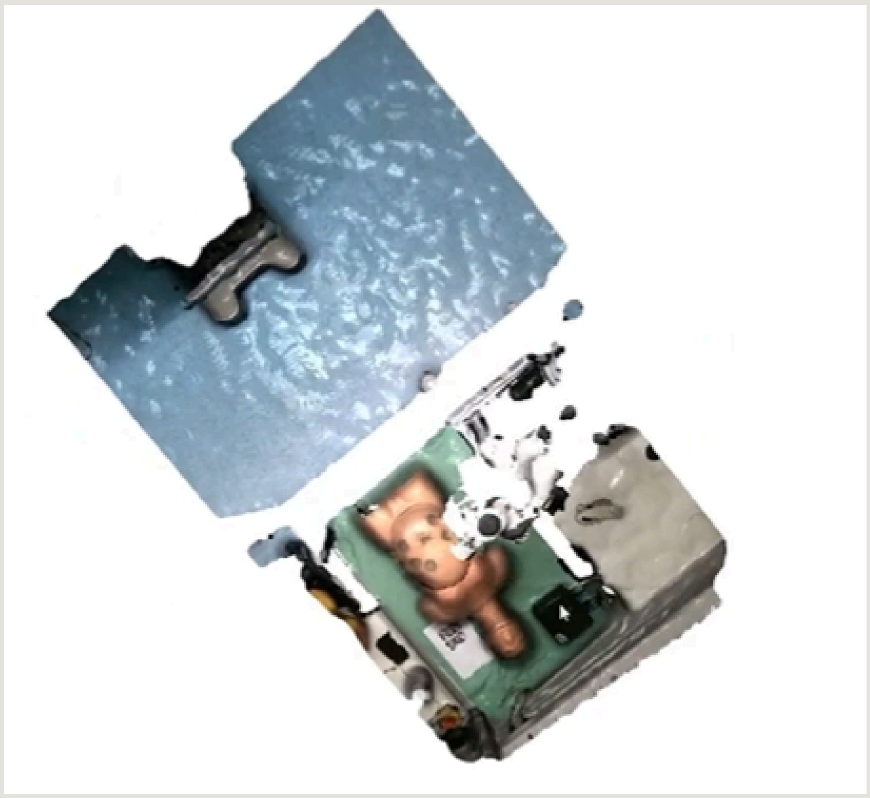
Hardware Setup

We use Franka Robotic arms to complete this project.

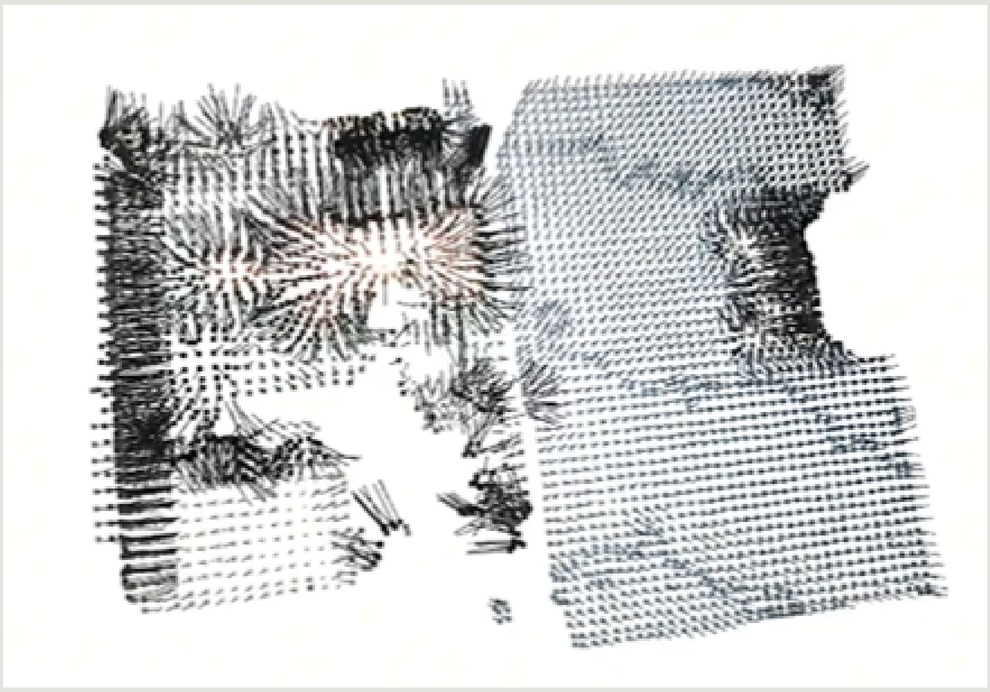


Results

- **3D Image captured by the depth camera**

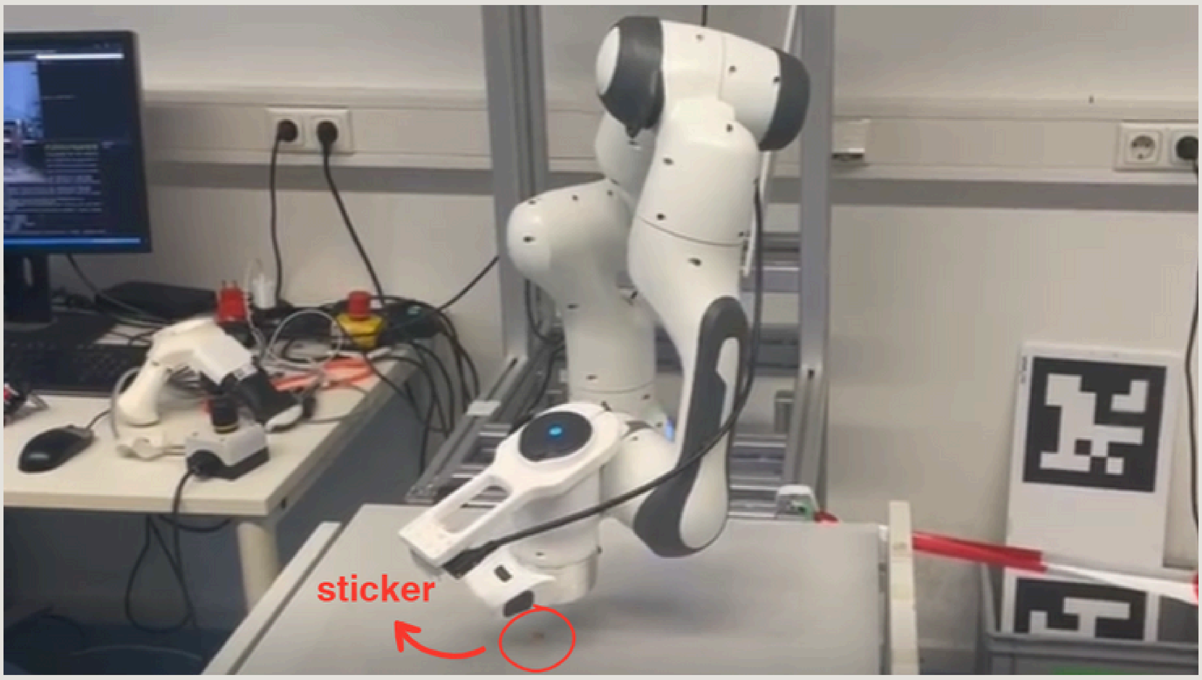


- **Visualization of the normal vector after implementing the downsampling method**



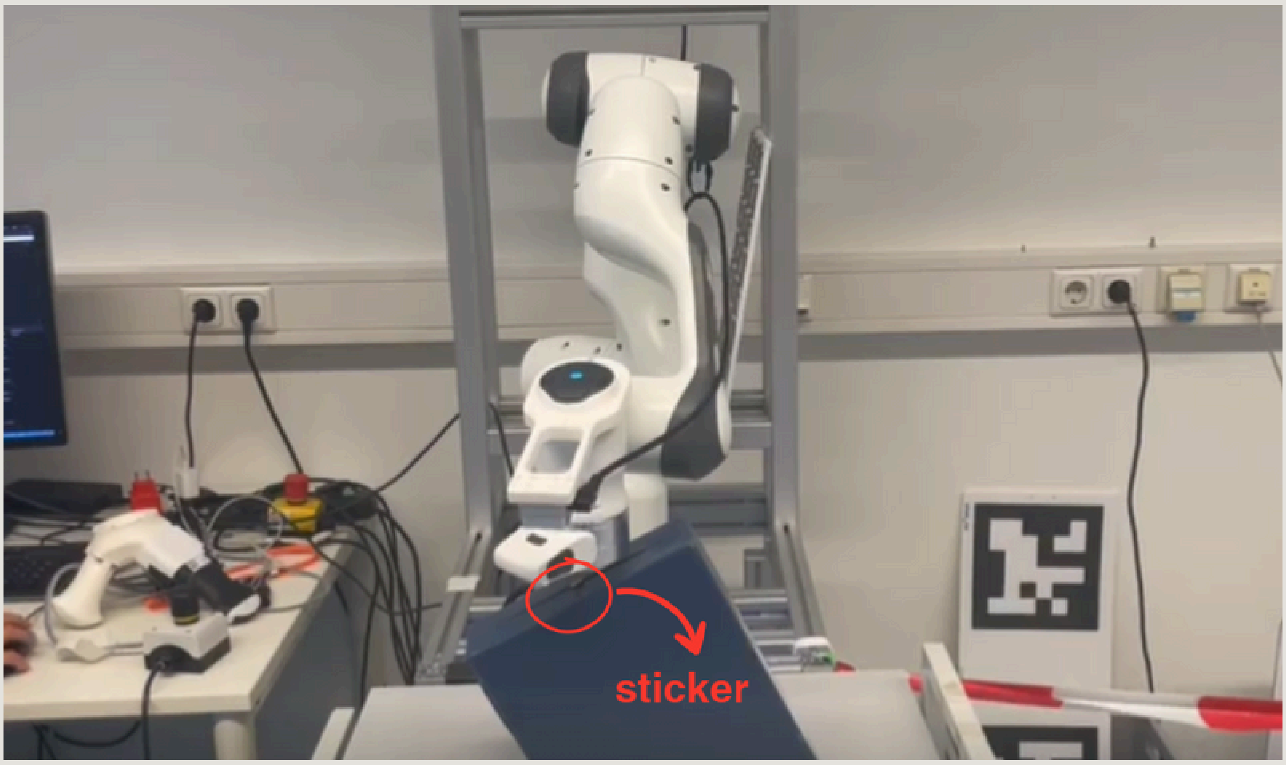
Robotic Arm Performance

- **Horizontal Surface**



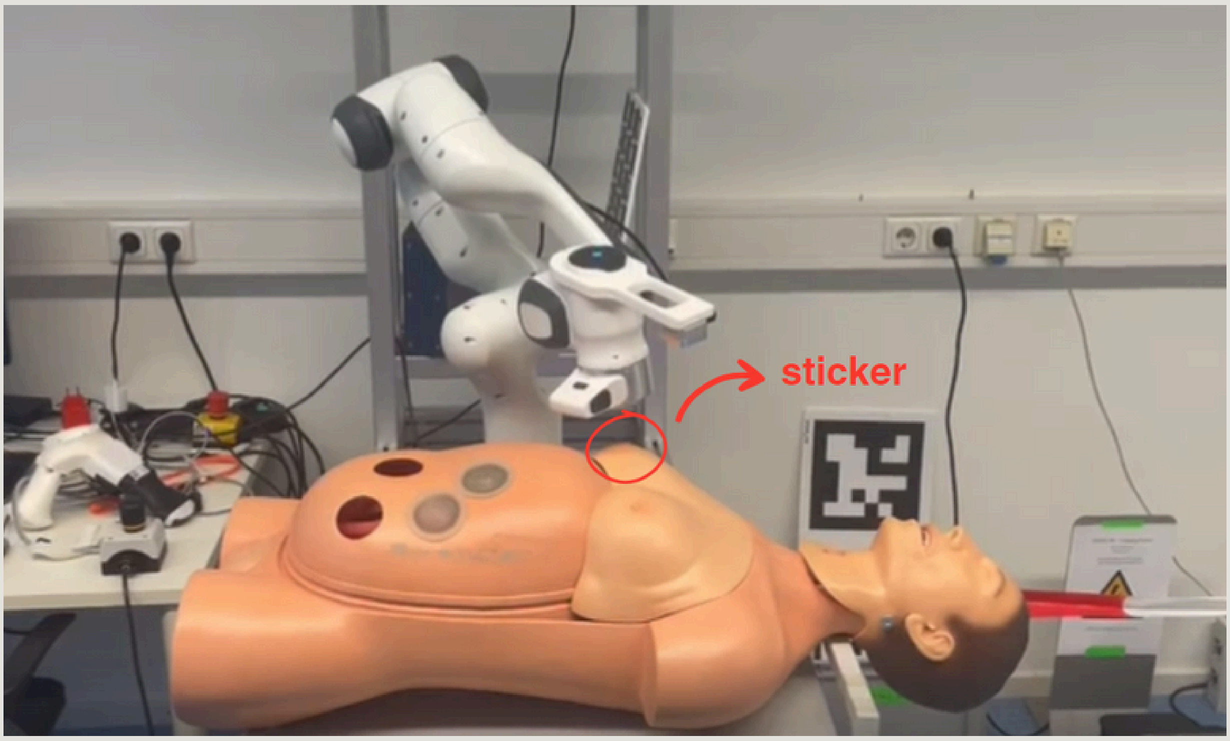
In the first experiment, we test the capability of the arm on a horizontal surface. After clicking the desired point, the arm moves to the point successfully.

- **Inclined Surface**



In the second experiment, we chose an inclined surface. After clicking the desired point, the arm also moves to the point successfully. However, we can see from the picture that the robotic arm isn’t precisely perpendicular with the surface.

- **Complex Surface**



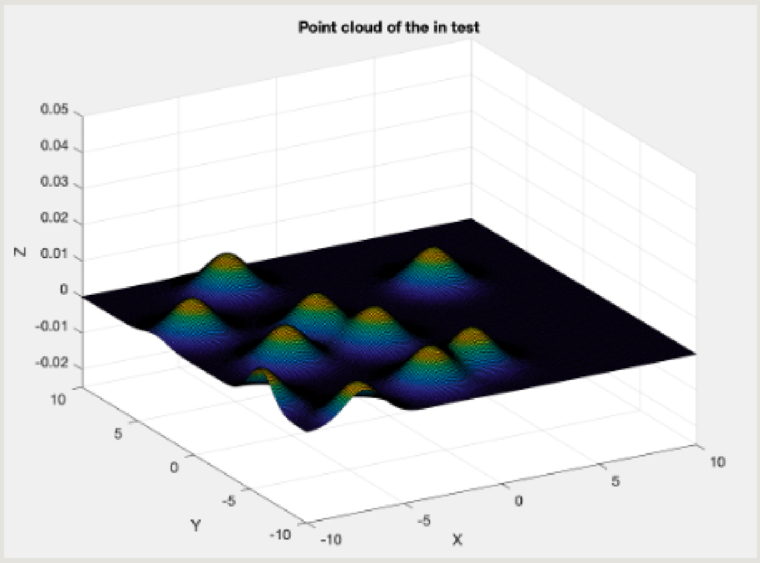
In this experiment, we chose a complex surface, such as human model. In order to test if our codes can change the direction of the arm, we also manually move the arm towards the sky as its original state. After clicking the desired point, the arm moves to the point successfully.

Analysis

- Error before applying Surface Normal Estimation(SNE): There are some noises when obtaining point cloud from depth camera

- Error during applying SNE: Downsampling vs Clicked Point
  - If voxel is large: Surface will be smoother, but contain less details
  - If voxel is small: Surface will contain more details, but may have some noise
  - <u>Best voxel we tested: 0.04</u>

- Error after applying SNE: The robotic arm requires quaternion datatype for the path planning. However, the transformation from 3D vector to quaternion may cause error.



noises of the depth camera captured the image of flat surfaces

Our work

We upload all our work, which includes code and all the slides presented to supervisors, on gitlab. The below is the link.

https://gitlab.lrz.de/6g-life_MITI/semiautonomous-telerobotic-examination-suite/cacom-auscultation

Reference

1. “Estimating Surface Normals in a PointCloud” [Online]Avaliable:https://pcl.readthedocs.io/projects/tutorials/en/master/normal_estimation.html
2. “Smoothing and normal estimation based on polynomial reconstruction” [Online]Avaliable:https://pcl.readthedocs.io/projects/tutorials/en/master/resampling.html#smoothing-and-normal-estimation-based-on-polynomial-reconstruction
3. “Normal Estimation Using Integral Images” [Online]Avaliable:https://pcl.readthedocs.io/projects/tutorials/en/master/normal_estimation_using_integral_images.html#normal-estimation-using-integral-images