# Object Information Estimation Using the TRI Hand

Ram Reddy and Alan Lu

May 28, 2025

## 1 Introduction

In this report, we explore improving robotic grasping, object recognition without vision, and 3D shape estimation using a soft tactile sensor robot hand based on air pressure [1]. Our approach includes implementing a dynamic pressure threshold for grasp detection and developing various hand motions to gather richer information about object size and shape. After data collection, we apply forward kinematics to approximate the 3D shapes of the grasped objects. Improving grasping without vision is important because vision may be occluded due to objects; having tactile data can also allow for grasping to be dynamic and more compliant with more materials, as the robot will not be reliant on just sight alone. This can help solve problems like slippage of objects and low-light environments and add more robustness to the robots' movement.

## 2 Experimental Design

In order to collect and estimate object shape, we kept the data collection setting as consistent as possible. For consistent data collection through time, we had simple dynamic calibration of the pressure sensors. At the start, before the hand moves, the program collects 10 sample pressures at a rate of 125 Hz. After collecting the initial data points, the program averages the data and adds a base of .3 to the thumb, pinkie, and index fingers, while adding 1.0 to the palm. We found that having offsets less than or greater than these thresholds would cause non-gentle grasping motions or lack of feedback from sensors. This was done as there are fluctuations in the read data, and the palm is more sensitive as it is more likely to be rested on by objects. If the measured values are above these thresholds, then the hand knows to stop the grasping motion. The thresholds are also calculated while the hand moves to compensate for any sensor changes, since the sensors automatically increase in value as they heat up. This also accounts for objects resting on the palm or any other finger for extended periods of time. To increase the task space coordinates, we implemented multiple different poses like tripod, grasp, wide, angled thumb, wide index, wide pinkie, and reset poses. The test poses, experimental data and coding are placed in the Google Drive link and the presentation slides.

Here is the pseudocode of our grasping algorithm:

**Algorithm 1** Grasping Control Flow

**Start** Get Initial Joint State  Initialize Pressure Thresholds  Calibrate Initial Pressure Values

**while** *true* **do**

  **if** *reset required* **then**

    Set `pose` to `reset pose`  reset = 0; calib_flag = 0

  **else**

    Set `pose` to `grasping pose`  reset = 1; calib_flag = 1; pose_index++

  Generate joint trajectory

  **if** *reset required* **then**

    Read current pressure values

  **else**

    Set pressures to minimum

  **if** *calib required* **then**

    Accumulate calibration values  **if** *calib_count == calib_count_reset* **then**

      Update pressure thresholds  and Reset calibration variables

    **else**

      Check pressure thresholds

  **else**

    Check pressure thresholds

  **if** *Any pressure* $\geq$ *threshold* **then**

    Set reset = 1  **break**

  **else**

    Publish joint positions  spinOnce and sleep

Figure 1: Objects Data was Collected On

## 3 Analysis

For each object we've measured, the joint states and tactile values are stored in a rosbag file, and the rosbag is converted to a CSV file using *rosbag_to_csv* [2]. Only the columns with interest in the CSV file, like the joint states of the 6 joints on the robot hand and the 4 pressure data from the tactile value array, are extracted into MATLAB and stored as an array. Among all the objects we've tested, the ball is considered the most ideal object since its size is appropriate for the hand and its shape is pretty straightforward and easy to analyze. First, the pressure data of the ball for each pressure sensor on the hand is given below.

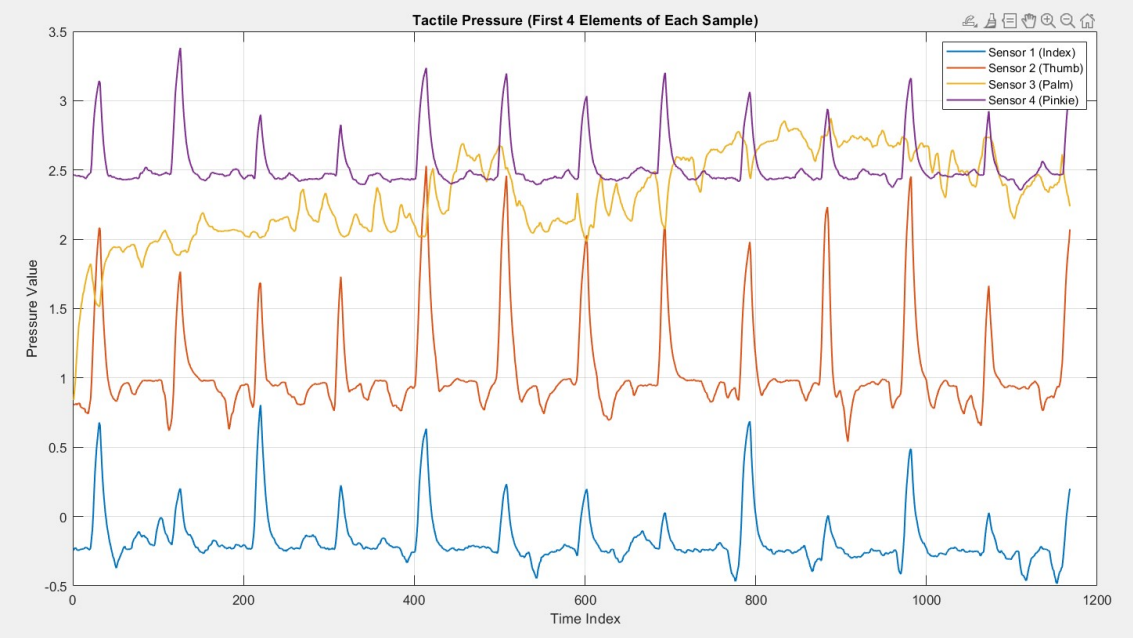


Figure 2: Pressure for the Ball Grasping Motion

Clearly, there are many peaks on these curves for sensors 1, 3, and 4. That's when the contact between the fingertip of the robot hand and the ball takes place, since a sudden increase in pressure is observed. The time indices of these peaks are stored in an array using the MATLAB built-in signal processing local maxima detecting functions, which will be used to calculate the end effector task space coordinate for the robot hand. Now, with these time indices, the joint states of contact points are extracted from the total joint state vector, and the plot with the contact points marked with red dots is given in the figure below:

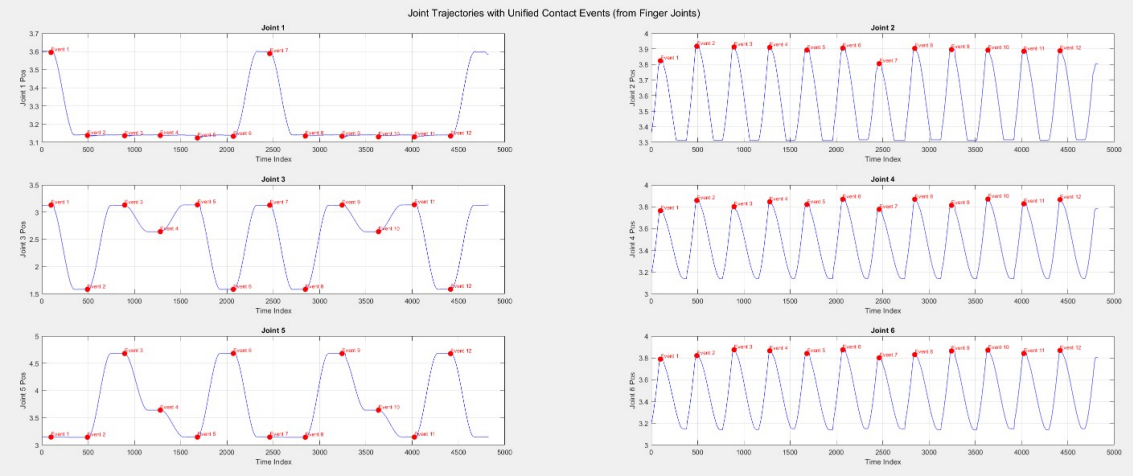


Figure 3: Joint trajectories for the Ball Grasping Motion

To calculate the forward kinematics of the robot hand, we are using the MATLAB Robotics Toolbox with the URDF file of the hand generated from the given workspace. The URDF file contains the complete kinematic structure of the hand, including link dimensions, joint types, joint limits, and the hierarchical relationships between links, which allows us to compute the fingertip positions using joint state inputs. The end effector coordinates plotted with 3 dimensions are shown in the figure below:

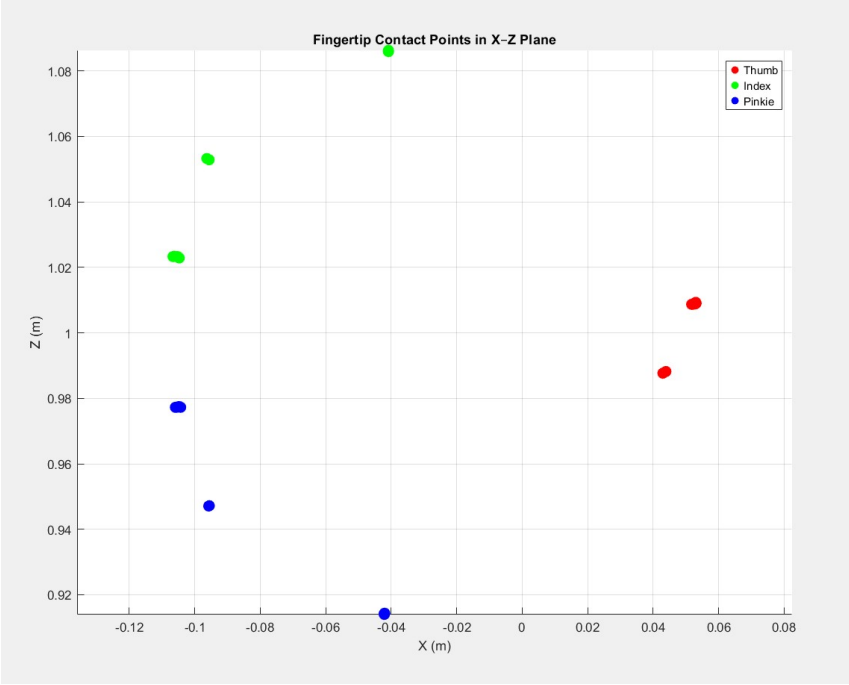


Figure 4: End effector task space contact points

As shown, the contact points show a circular shape, which is what we expected to see when the robot hand is grasping a ball. The diameter of the circle cut at the ball around

the contact that's parallel to the ground is estimated to be 16.2 cm by averaging the furthest point pairs in both directions. A similar procedure is applied to other objects, and their actual length, estimated length, and percentage error are shown in the table below.

Table 1: Estimated vs Actual Contact for Different Objects

| **Object** | **Actual (cm)** | **Estimated (cm)** | **Error (%)** |
|---|---|---|---|
| Ball | 14.0 | 16.2 | 15.7 |
| IZZE Bottle (Vert) | 8.3 | 12.8 | 54.2 |
| IZZE Bottle (Horiz) | 8.3 | 11.6 | 39.8 |
| Bowl | 11 | 13.5 | 22.7 |
| Bowl (Plastic) | 11 | 14.2 | 29.1 |
| Cup (Horiz) | 8.5 | 12.6 | 67.4 |
| Cup (Vert) | 7.5 | 12.8 | 70.6 |
| Box (Horiz) | 14 | 15.2 | 8.6 |
| Box (Vert) | 12 | 14.7 | 22.5 |

## 4 Discussion

Although we fixed many of the issues with data collection, there are still a few issues that affect the results. In our algorithm, we stop the joint trajectories after any of the finger or palm thresholds are above the calculated thresholds. This decreases accuracy as it is hard to map and calculate the size of the object, as some of the joints may not actually be touching the object, while others are. To mitigate this, in our analysis, we cross-referenced the pressure data to see which fingers actually were touching the object; however, this still has some error. In the future, it would be better to only reset the pose after all the fingers are touching the object instead of just one. What's more, throughout working on this project, we faced many hardware and driver issues. We were never completely able to solve the faulty connection between the computer and the hardware, as it always kept disconnecting, and this slowed progress — over 50% of the time we are trying to reconnect the motor and pressure sensor by unplugging and reconnecting the USB probe, as they always lose connection unexpectedly.

Additional sources of error come from the physical setup and sensing fidelity. From the table above, it seems like larger objects tend to produce smaller relative error, as the proportional impact of finger misalignment decreases. The effective end-effector location is within the fingertip structure, not precisely at the contact point, introducing spatial offset when calculating contact locations. Furthermore, the grasp may not fully stabilize the object during closing, resulting in underestimation of size. Tactile sensing accuracy is also affected by uncertainty in the pressure sensor calibration and potential kinematic inaccuracies from the URDF model. Also, the plots of the end effector for all the objects look pretty similar due to the fact that there are only three end effectors, and they gener-

ally all give a circular-shaped task space coordinate distribution. Significant differences could not be observed unless the fingertips of the hand are more than 3, such that they give more shape information for a single grasping setup.

Future improvements could include increasing the number of fingers for more robust grasp coverage, adding more actuators per finger for better shape conformance, and mechanically integrating the hand with a robotic arm. This would allow task-space registration from the base of the arm to the fingertips, enabling accurate measurement of contact points on larger or external objects using full-body forward kinematics. Integrating the robot hand to the arm will be considered the next step of our project since it will allow us to interact with a wider range of object geometries and orientations, enhance spatial awareness through full-body kinematic mapping, and enable more complex manipulation tasks beyond static grasps—ultimately improving the precision and utility of tactile-based object estimation in real-world scenarios.

## 5 Conclusion

In conclusion, improving robotic grasping without vision is feasible. With a rudimentary algorithm, we were able to approximate the size of symmetrical objects greater than 10 cms within 20 percent of error. There are many avenues of improvement due to possible inaccuracies of data collection such as as not closing all of the joints on the object. Other improvements with the URDF model can also drastically increase accuracy. Overall, it was a great learning experience, and further research can be done to improving grasping and visualization of the object without vision.

## References


[1] S. Taylor, K. Park, S. Yamsani, and J. Kim, "Fully 3d printable robot hand and soft tactile sensor based on air-pressure and capacitive proximity sensing," in *Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, Yokohama, Japan, May 2024, pp. 18 100–18 105. DOI: 10.1109/ICRA57147.2024.10610731.

[2] A. Sakai, *Rosbag_to_csv*, https://github.com/AtsushiSakai/rosbag_to_csv, Accessed: 2025-05-15, 2016.