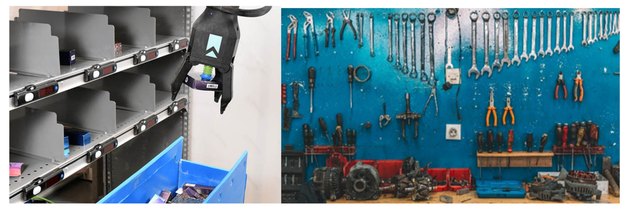
Robotic Tool Handler

Enis Zuo, Haixi Zhang, Xiaoshan Liu {azuo, hzh104, xliu119}@u.rochester.edu}
UNIVERSITY OF ROCHESTER, HAJIM SCHOOL OF ENGINEERING AND APPLIED SCIENCES
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Introduction



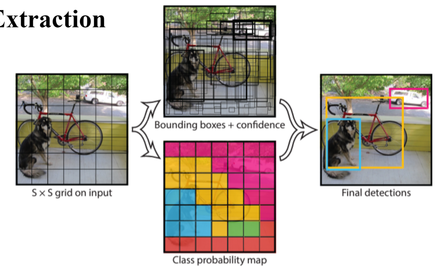
Project Goal: Develop a smart robotic arm to automate tool handling, boosting efficiency and safety in industrial settings.

Need: Tackles the significant labor shortage in skilled trades (As of 2021, there were 81,292 unfilled electrician jobs in the U.S.), enhancing productivity and reducing downtime.

Current Outcome: Algorithmic control of gripper, path planning of the robotic arm & object grasping in the visualizer, and user's voice command input understanding.

YOLO 2D Box Extraction

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w e^{t_w}$$
$$b_h = p_h e^{t_h}$$



PostProcessing: Camera Pixel to Cartesian Point Cloud

Require: Depth array (D) filtered by bounding box, camera intrinsic parameters (fx, fy, cx, cy)
  for each row V from 1 to $D_{(rows)}$ **do**
    for each column U from 1 to $D_{(cols)}$ **do**
      $Z = D[V, U]$
      $X = (U - cx) * Z / fx$
      $Y = (V - cy) * Z / fy$
      `points.append([X, Y, Z])`

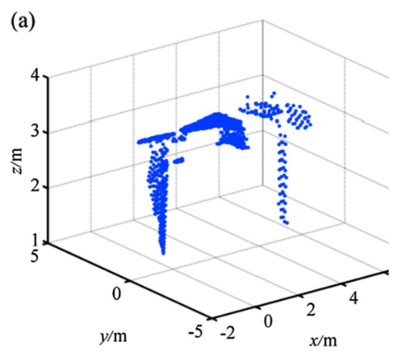
Perception Module

- **Uses Intel RealSense D405 RGBD camera and YOLOv5 2D object detector**
- **Converts 2D pixels corresponding to the detected bounding box to 3D point clouds in the camera frame**
- **Uses RANSAC to segment the table from objects**
- **Calculates object centroids and table height**
- **Transform the Frame & visualize point clouds in RViz**

Algorithm 2 Iterative RANSAC for Segmenting Objects from Table

Given:
  D – A set of data points.
  N – The minimum number of points required by the model.
  n – The maximum number of iterations.
  ϵ – The threshold for a point to be considered an inlier.

for `iterations = 1` **to** n **do**
  `maybeInliers = random_sample(data, N)`
  `maybeModel = fit_model(maybeInliers)`
  `nInliers = 0`
  for `point in data` **do**
    `distance = calculate_distance(point, maybeModel)`
    if `distance < ` ϵ **then**
      `nInliers += 1`
  if `nInliers > bestInliers` **then**
    `bestInliers = nInliers; bestModel = maybeModel`
    `bestOutliers =` D`\bestInliers`
return `nInliers, outliers`

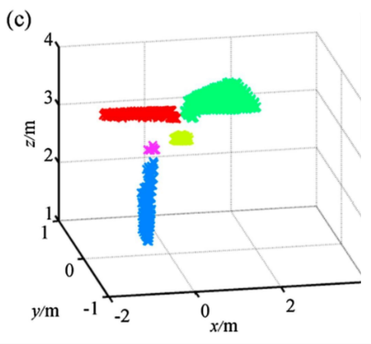


Acquire plane's normal vector of the sampled set with cross-product

$$\vec{n} = \begin{pmatrix} A \\ B \\ C \end{pmatrix} = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{pmatrix} \times \begin{pmatrix} x_3 - x_1 \\ y_3 - y_1 \\ z_3 - z_1 \end{pmatrix}$$

Select plane with point distances to the plane without normalization

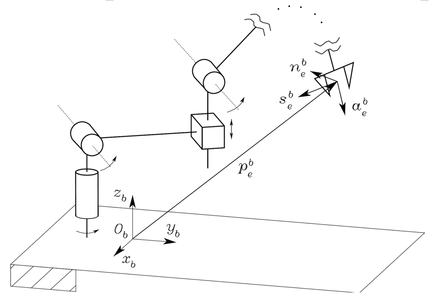
$$|Ax_j + By_j + Cz_j + D| \leq \epsilon$$



Planning Module

$$T_e^b(q) = \begin{bmatrix} n_e^b(q) & s_e^b(q) & a_e^b(q) & p_e^b(q) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_e^b(q) = A_1^0(q_1)A_2^1(q_2)\cdots A_e^{e-1}(q_e)$$

where $A_e^{e-1} = \begin{bmatrix} R_e^{e-1^T} & o_e^{e-1} \\ 0^T & 1 \end{bmatrix}$



Forward Kinematics
Transform objects detected by camera from camera frame to world frame.

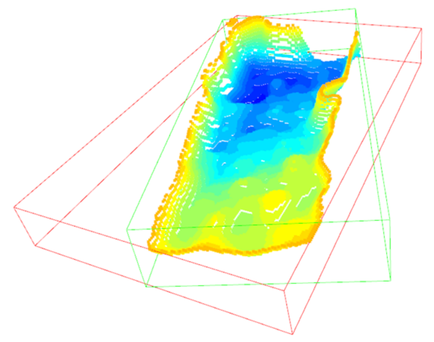
Each transformation is dependent on robot setup (urdf) and joint's angle.

Inverse Kinematics
Given gripper global position, solve joints angles in the robotic arm to achieve desired position.

$$q^* = \arg\min_q \|X_d - X(q)\|^2$$
$$\Delta q = -(J^T J + \lambda I)^{-1} J^T e$$
$$q_{\text{new}} = q + \Delta q$$

Iteratively solve until the error $e = X_d - X(q)$ is sufficiently reduced.

Acquire Point Cloud (object)'s Orientation



Covariance Matrix Computation:

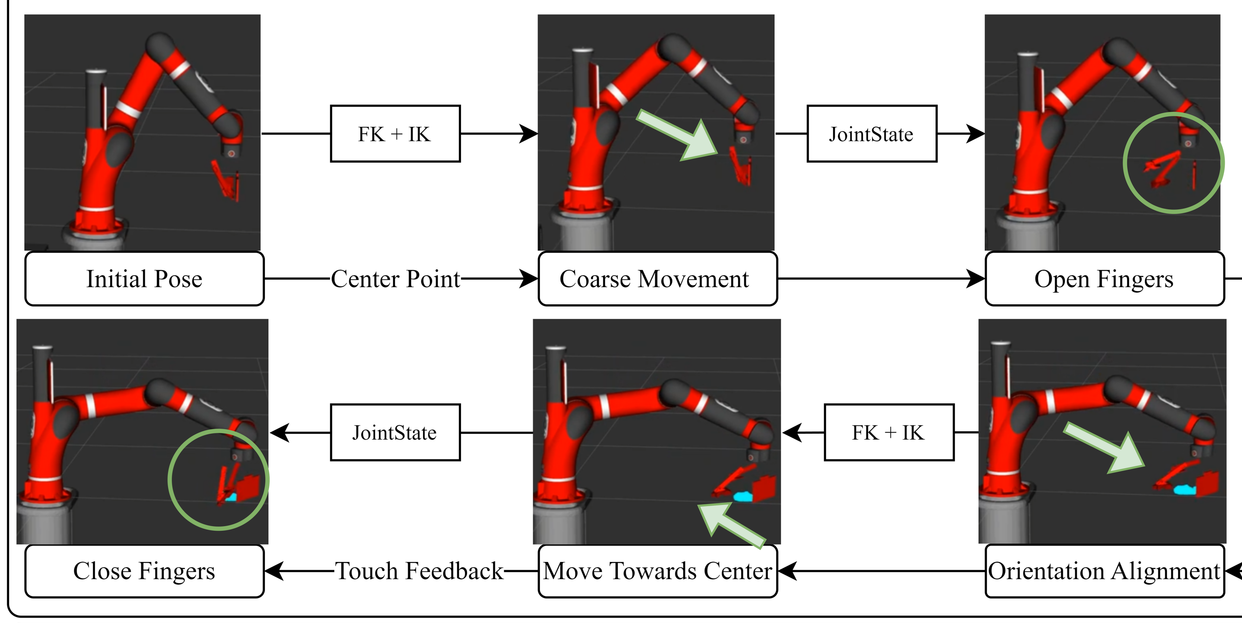
$$\Sigma_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} (z_{ij} - \bar{z}_j)(z_{ik} - \bar{z}_k)$$

Eigenvalue Decomposition: $\Sigma V = VD$

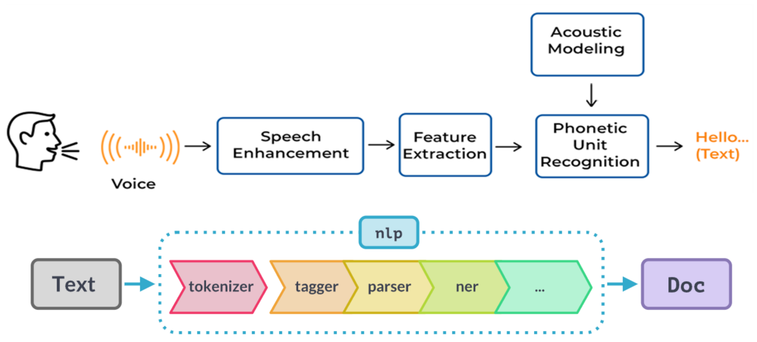
Selection of Eigenvectors with highest variance (energy) as major axis.

$$max(\frac{g_j}{g_p})|g_j = \sum_{k=1}^{p} D_{kk} \quad \text{for } j = 1, \ldots, p; g_p = \sum_{k=1}^{p} D_{kk}$$

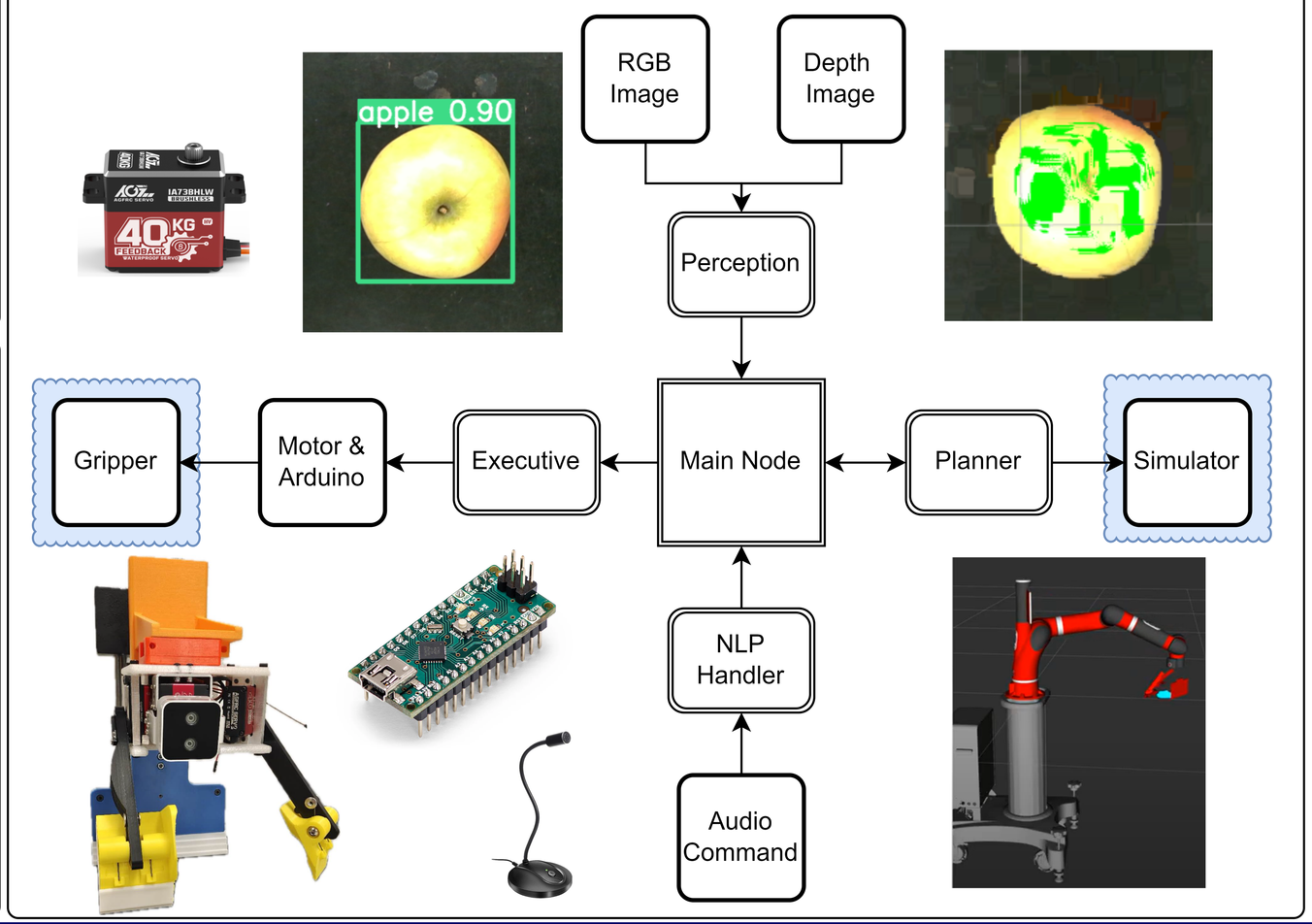
Planner Operation Flow



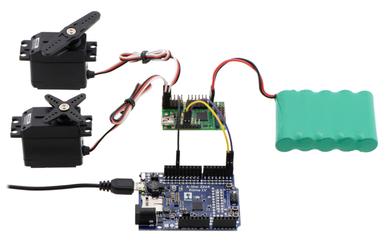
NLP Handler Module

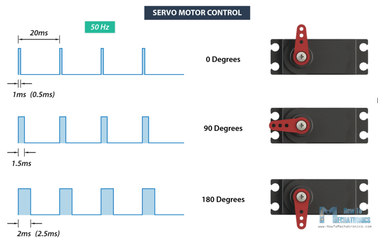


Project Architecture

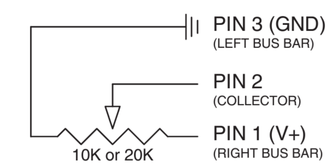


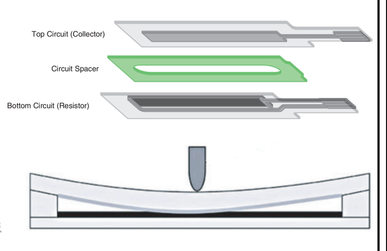
Gripper Motor





Touch Feedback





Conclusion

- Our project accomplished several key tasks in building a human-robot interaction system and demonstrated the feasibility of a tool handler.
- We successfully captured audio input and converted it into command messages, enabling voice-driven robot control. Using an RGBD camera for environmental perception, we accurately detected and segmented multiple objects in a 3D space.
- The system can calculate both forward and inverse kinematics, allowing for planning in the robotic's base frame and precise robot movement.
- Further, it can understand the orientation of the target object and plan a grasp by integrating touch feedback.

Future Directions

- **Fine-Tuning YOLO Model**: to enable the system to detect a broader array of objects with greater precision.
- **Voice-Environment Interaction:** to enable robot to understand the command with respect to the perceived surroundings.
- **Refine Grasp Sampling:** to enable the gripper to pick up irregular tools.
- **Incorporate Force Feedback:** to enable the gripper to handle soft or deformable tools.
- **Docker Containerization**: simplify the setup process for users and ensure compatibility across different environments.