Vision-Based Autonomous Grasping and Placing for NAO Robot

Student: Tzu-Ching Chao, Professor: Kai-Yew Lum
Department of Electrical Engineering, National Chi Nan University

I. Motivation

This project develops an autonomous object grasping and placing system using NAO robotics, integrating YOLOv8-based computer vision, motion control, and arm coordination. The system detects objects in real-time, estimates 3D coordinates without depth sensors, and performs grasping actions using inverse kinematics and touch sensor feedback with a 5-DOF robot arm.

II. Methodology and System Architecture

There are five main contents of the system:

- YOLOV8 Vision: For 3D localization and recognition via image processing.
- Forward and inverse kinematics: For precise arm path planning.
- Arm Control: Tactile feedback mechanism to refine gripping for adaptive and stable grasping in response to object contact.
- Motion Control: To move the NAO to detect and align with the target.
- Grasp planner: To design relay and final grasp positions for stable object gripping.

III. Technical Implementation Overview

1. Based on YOLOv8 object recognition model training.

The NAO robot's built-in camera captures object images in the Webots simulation environment to create a training dataset.
A total of five object types are selected:

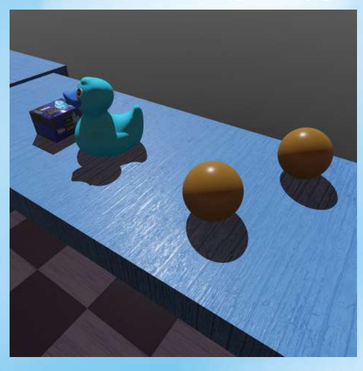

Fig.1(a) Original Image

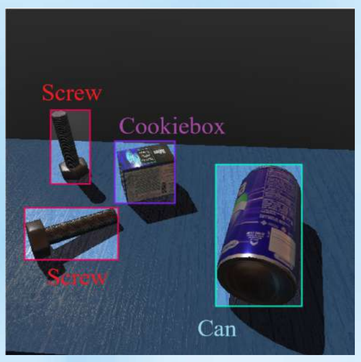

Fig.1(b) Screw, Can, Cookie box

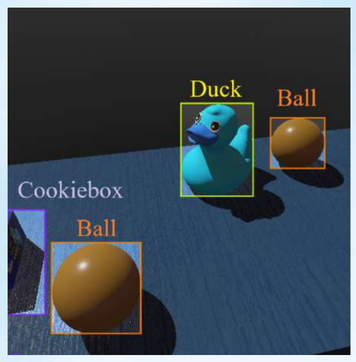

Fig.1(c) Duck and Ball

Figure 1. The collected image data used to label the objects before model training is shown. Fig. 1(a) Displays the original image with a size of 640×640 (unlabeled). Fig. 1(b) and Fig. 1(c) have labeled the objects in this collection, which consists of two sets of similar-looking objects (Can and Screw, Ball and Duck) and a square object (Cookie box) to increase the challenge of model recognition.

The following are strategies for collecting a dataset based on the environment:

- Object Placement: Learn about objects from different angles.
- Picture Complexity: Each image contains multiple objects.

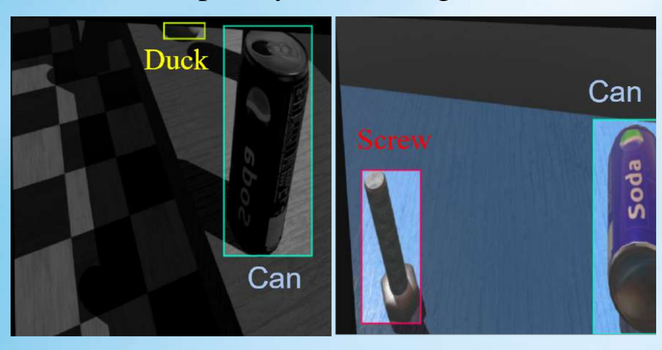

Figure 2. Preprocessed images for training to enhance model performance. Left: grayscale, rotated. Right: brightness-adjusted, rotated. Black padding ensures 640×640 resolution.

Labeled preprocessed and original images form a 1852-image dataset. The dataset is trained on YOLOv8x (*epochs*=1000, *imgsz*=640, *batch*=8, $lr0 = 0.01$, $lrf = 0.01$, *warmup*=3) with early stopping and adaptive learning rate decay for object detection. Finally, the training was stopped at the 542nd epoch, and the learning rate decayed from 0.001 to 0.0004.

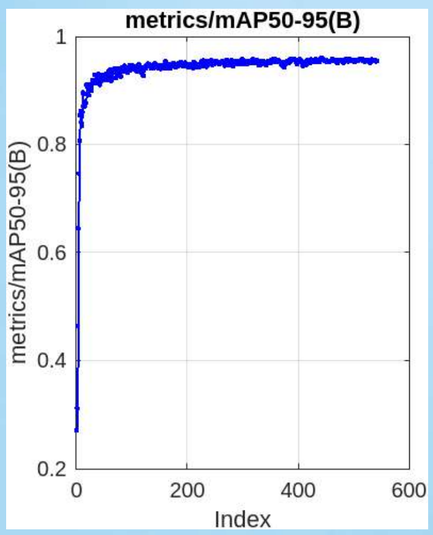

Figure 3. YOLOv8x mAP50–95 (B) – average precision across IoU 0.50–0.95. The metric rises steadily, stabilizing near 0.95, reflecting robust detection with early stopping and adaptive learning rate decay.

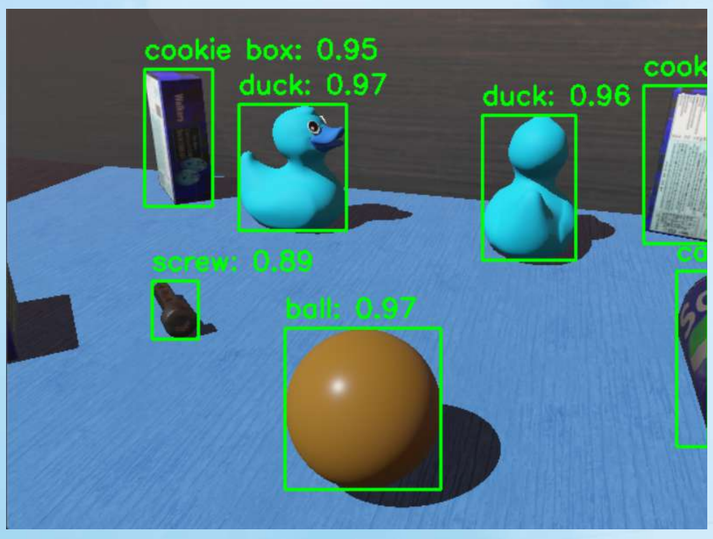

Figure 4. Real-time object detection with YOLOv8x on NAO robot in complex scenes, accurately identifying objects even when partially obscured, and displaying object names and confidence scores.

2.From object image borders to torso coordinates:

Based on the pinhole camera modeling principle and coordinates conversion.

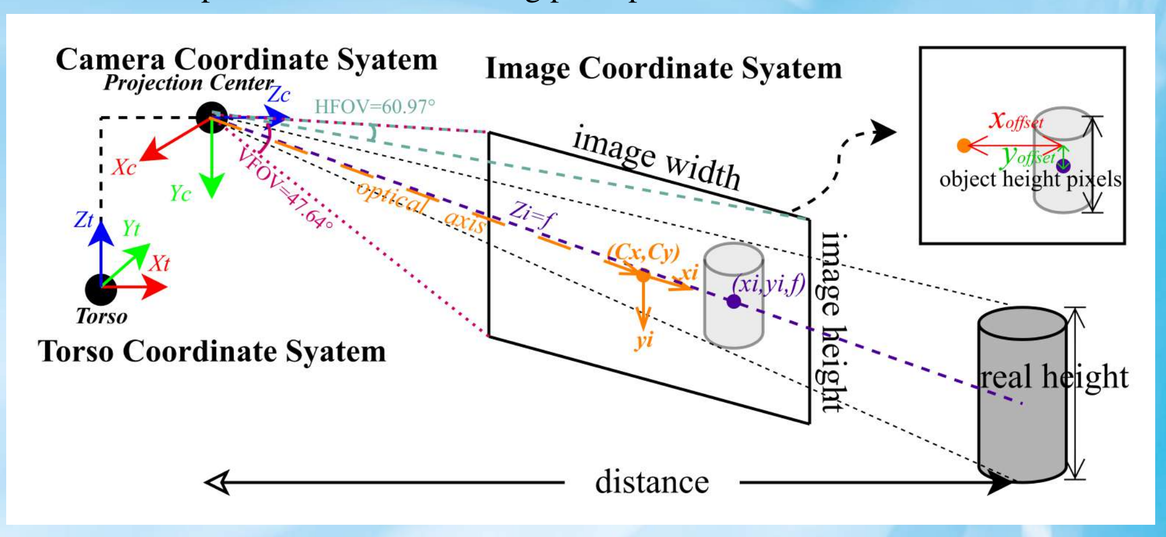

Figure 5. Model of a pinhole camera with focus and camera coordinate system.

The object angle is calculated for the center bounding box with an offset:

$$x_{offset} = center_x - \frac{\text{image\_width}}{2},\ y_{offset} = center_y - \frac{\text{image\_height}}{2}$$

The horizontal and vertical angles are:

$$\text{horizontal\_angle} = \arctan\left(\frac{x_{offset}}{f_x}\right), \text{vertical\_angle} = \arctan\left(\frac{y_{offset}}{f_y}\right)$$

The distance (depth) is calculated as:

$$\text{distance} = \frac{f_y \times \text{real\_height}}{\text{object\_height\_pixels}}$$

The 3D coordinates in the camera coordinate system are:

$$x_{cam} = \text{distance} \times \tan(\text{horizontal\_angle}), y_{cam} = \text{distance} \times \tan(\text{vertical\_angle})$$
$$z_{cam} = \text{distance}$$

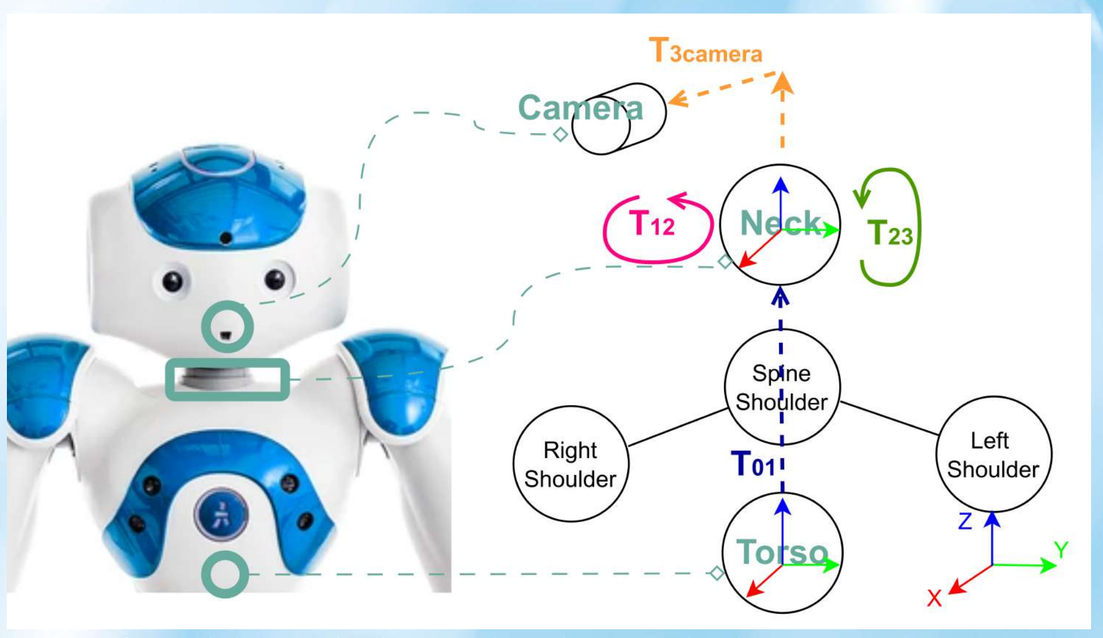

Figure 6. Schematic diagram of matrix conversion from camera to torso.

The conversion positions of the matrices are defined as follows.

- **T_{01} – Torso to Head Base Translation**
  Translates from the torso to the neck joint along the Z-axis (torso height offset).
- **T_{12} – Head Yaw Rotation**
  Rotates around the Z-axis based on the robot's yaw angle (left/right head rotation).
- **T_{23} – Head Pitch Rotation**
  Rotates around the Y-axis based on the robot's pitch angle (up/down head movement).
- **$T_{3camera}$ – Head to Camera Translation**
  Adjust the tilt rotation by panning the physical offset to the camera's coordinates.

The complete transformation matrix is given by:

$$\mathbf{T}_{0,\text{camera}} = \mathbf{T}_{01} \cdot \mathbf{T}_{12} \cdot \mathbf{T}_{23} \cdot \mathbf{T}_{3,\text{camera}}$$

$$\text{Object coordinates (originating from the torso)} = \mathbf{T}_{0,\text{camera}} \cdot \left(\begin{bmatrix} z_{\text{cam}} & -x_{\text{cam}} & y_{\text{cam}} & 1 \end{bmatrix}\right)^{\top}$$

3.Kinematic Approach to Robot Arm Control in NAO:

- Forward Kinematics: *Homogeneous transformation matrices* , for computing hand position by multiplying joint transformations.
- Inverse Kinematics: *Numerical iteration* with a *Jacobian matrix* and *gradient descent*, to adjust joint angles, for reaching a target position.

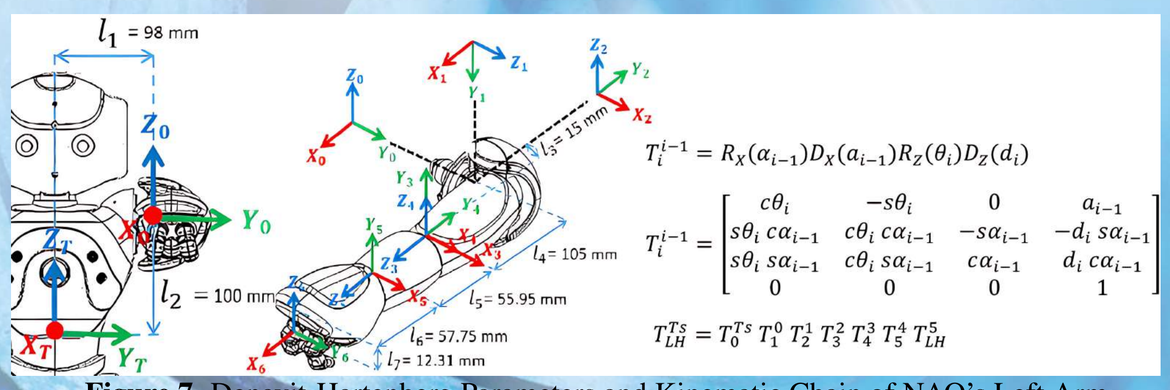

Figure 7. Denavit-Hartenberg Parameters and Kinematic Chain of NAO's Left Arm

國立暨南國際大學 電機工程學系
Department of Electrical Engineering, National Chi Nan University