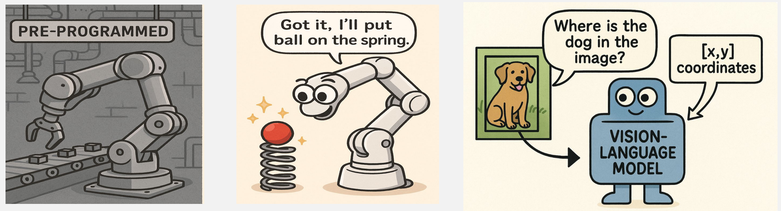
From Language to Action: A Vision-Language and Kinematics-Guided Framework for 6-DOF Robotic Arm Manipulation

Author: Zihao Yu[1,2,3] Course Advisor: Prof. Jason Trobaugh[1] Design Advisor: Prof. Dennis Mell[1]

1. Department of Electrical and System Engineering 2. Department of Computer Science and Engineering 3. Department of Physics, Grinnell College

Introduction & Educational Motivation

As robotics moves beyond pre-programmed routines into intelligent autonomy, there is a critical need to bridge vision-language understanding and robotic kinematics. This project tackles that challenge by developing an affordable, intelligent robot assistant capable of interpreting human language and executing real-world grasping tasks.



This capstone project is inspired by the excellent foundation offered in core courses in control systems, robotics, and natural language processing at Washington University in St.Louis, including ESE-4480, ESE-446, and CSE-527. These courses provide essential training in system modeling, feedback control, and AI fundamentals. Building on that groundwork, the project explores how modern **vision-language models (VLMs)** can be integrated into real-time robotic systems—an area not yet formally covered in the curriculum.

Key Contributions

The platform combines classical robotics with modern AI through the following **key contributions**:

(1) Hardware Integration:

A compact tabletop platform was built using the *Elephant Robotics myCobot 280 Pi* (6-DOF arm with onboard Raspberry Pi), an $6 *Adafruit Ultra Tiny Camera* mounted for top-down viewing, an *adaptive parallel-jaw gripper*, and a *3D-printed base secured with a G-clamp* for workspace stability.

(2) ROS 2 Control Framework:

Developed a modular ROS 2 (Humble) control system implemented in Python 3 on Ubuntu 22.04, enabling real-time communication across hardware components through custom nodes, services, and message types.

(3) 2D Hand-Eye Calibration:

Implemented a lightweight affine transformation method to convert camera pixel coordinates into robot-frame positions, enabling accurate visual grounding for grasp execution.

(4) Vision-Language Instruction Parsing:

Connected to *Qwen2.5-VL-Instruct* via the Aliyun API. The **VLM** is a multimodal AI system that processes both images and natural language inputs. It generates responses in an autoregressive manner, conditioned on visual context and user instructions. In this system, the VLM functions as both a **conversational interface** and a **visual perception module**—outputting structured commands (e.g., vlm_move(...)) and object bounding boxes directly from natural language prompts.

(5) Frame Prediction via Segmentation + PCA:

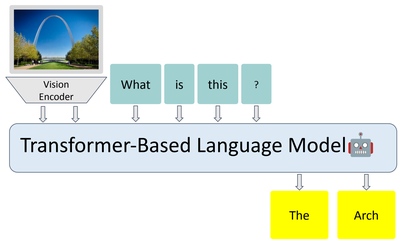
Introduced an efficient method combining *EfficientSAM* segmentation (prompted by VLM-detected bounding boxes) with *Principal Component Analysis* to estimate object-centric 2D grasp frames. These frames allow the gripper to align with the geometry of non-uniform objects for stable and interpretable grasping.

Vision Language Model

A vision-language model (VLM) jointly conditions on textual input $x = (x_1, \ldots, x_N)$ and visual features v extracted from an image to generate a response $y = (y_1, \ldots, y_T)$ autoregressively. The model estimates the conditional probability of the output sequence as:

$$P(y \mid x, v) = \prod_{t=1}^{T} P(y_t \mid y_{<t}, x, v)$$

where $y_{<t}$ represents all previously generated tokens. This formulation allows the model to reason over both language and vision inputs when producing each token in the response.



2D Hand-Eye Calibration

Camera coordinates (x', y') to robot coordinates (x, y) as an affine transformation:

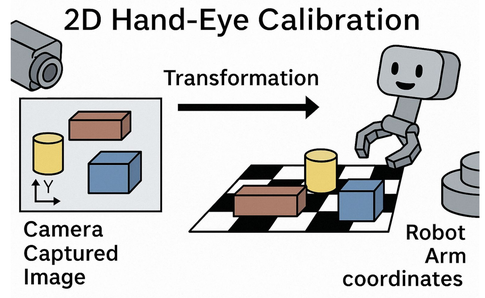
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Collect three coordinate pairs:

$$\underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{bmatrix}}_{\mathbf{M} \in \mathbb{R}^{6\times 6}} \cdot \underbrace{\begin{bmatrix} a \\ b \\ t_x \\ c \\ d \\ t_y \end{bmatrix}}_{\mathbf{u} \in \mathbb{R}^6} = \underbrace{\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \\ y'_1 \\ y'_2 \\ y'_3 \end{bmatrix}}_{\mathbf{y} \in \mathbb{R}^6}$$

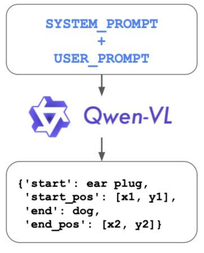
Solve with least square:

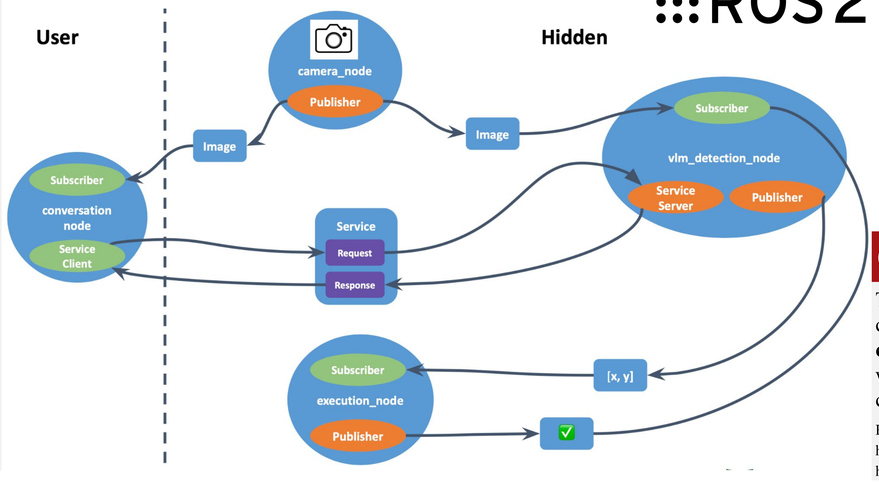
$$\mathbf{u} = \text{argmin}_{\mathbf{u}} \|\mathbf{M}\mathbf{u} - \mathbf{y}\|_2^2$$



Pick-and-place object via Vision-Language Understanding

- A Vision-Language Model (Qwen2.5-VL) serves as an **instruction-driven object detection module**, allowing users to extract bounding boxes for target objects **directly through natural language prompts**.
- Given an instruction like *"Place the ear plug on the dog,"* the VLM returns structured JSON containing the bounding boxes of the source and destination objects.
- Detected bounding boxes are converted to grasp points using 2D hand-eye calibration and mapped to the robot's workspace.
- The robot executes top-down pick-and-place motions based on the center of each bounding box.
- A feedback loop computes the placement error and reattempts execution if needed, improving final accuracy.

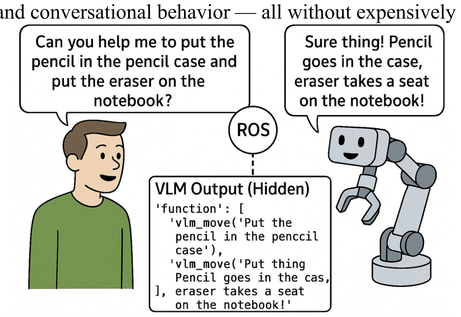




Empower Natural Conversation with Robot

A large language model (LLM) acts as a conversational interface for the robot, responding to user instructions with both structured commands and natural dialogue.

- **System Prompt Design**: The system uses **few-shot prompting** and **in-context learning** to enable flexible command interpretation and conversational behavior — all without expensively fine-tuning the model.
- **Instruction-to-Action**: Guides the LLM to output a JSON with:
  - function: robot actions (e.g., vlm_move(...)), visual question answering (e.g., vlm_vqa(...)).
  - response: friendly user message
- **ROS 2 Integration**: JSON outputs are parsed and dispatched to control nodes for movement, perception, and interaction.



Predict 2D Grasp Frames for Alignment

- Apply **EfficientSAM** to segment objects from VLM-predicted bounding boxes.
- Apply **PCA** on the mask to extract a 2D grasp frame (center + X/Y axes).
- Converted frame output into robot gripper orientation using kinematic transformations.

The grasp pose is defined by the frame:

- Origin: $\boldsymbol{\mu}$ (center of mass)
- X-axis: $\vec{X}$ (grasp alignment direction)
- Y-axis: $\vec{Y}$ (orthogonal axis)

The 3D homogeneous transformation matrix for planar grasping is:

$$T_{\text{gripper}} = \begin{bmatrix} X_x & Y_x & 0 & \mu_x \\ X_y & Y_y & 0 & \mu_y \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where h is the height of the gripper above the object surface.

User Input	VLM Response	Action Taken
"Hi there!"	Friendly reply	No motion executed
"What is linear algebra?"	Educational answer	No motion executed
"Put the red block on the dog"	JSON + dialogue	Pick-and-place executed; 2D frame labeling activated if object size > 100 px (≈ 1.89 cm)
"Place the pen in the case and the eraser on the notebook"	Multi-action JSON + reply	Two pick-and-place actions executed; frame labeling triggered on larger object(s)

Conclusion & Future Work

The system achieves robust 2D manipulation through vision-language reasoning and real-time control. Across trials, first-attempt placement errors ranged from **0.69–1.14 cm**, improving to **0.17 cm** with feedback-based correction—demonstrating consistent and precise execution. Future work will extend the system to 3D by integrating an RGB-D camera and performing 3D hand-eye calibration for full 6-DOF grasping.

Ref: 1. Cartoons generated by ChatGPT 4o 2.*Camera Module Image.* Adafruit Industries. *Adafruit Ultra Tiny 640x480 Camera Module.* https://www.adafruit.com/product/5733. 3. *Robot Arm Image.* Elephant Robotics. *myCobot 280 Pi Product Introduction.* https://docs.elephantrobotics.com/docs/mycobot_280_pi_cn/1-ProductInformation/1.ProductIntroduction/1-ProductIntroduction.html.