Large Language Model Based Virtual Robot Pick-Place Manipulation

Group 18:

Shashwat Nayak, Zeel Desai, Rohan Jayachandran

Introduction

Project Concept:

- Integration of Large Language Models (LLMs) with robotic actions.
- Simulated environment for robot and language model interaction.

Importance in Smart Environments:

- Enhancing robot intuitiveness and responsiveness in homes and workplaces.
- Advancing user-friendly robotics for everyday tasks.

Project Aim:

- Employing LLMs for context-aware robotic actions.
- Demonstrate practical LLM applications in household scenarios.

Background and Inspiration

Inspiration

- Title: "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances"
- Key Insight: Potential of LLMs in enhancing robotic functionalities

Bridging Theoretical and Practical Realms

- Challenge: Aligning LLM's theoretical language comprehension with practical robotic tasks
- Innovation: Context-aware and LLM-driven robotic responses for real-world applications
- Extending 'SayCan' Basic Demo Concept

Core Functionality of SayCan

Semantic Knowledge Utilization

Leverages the extensive semantic understanding of LLMs.

Contextual Grounding

Ensures proposed actions are both feasible and appropriate for the context

Robot as the Interface

Acts as the 'hands and eyes' for the language model.

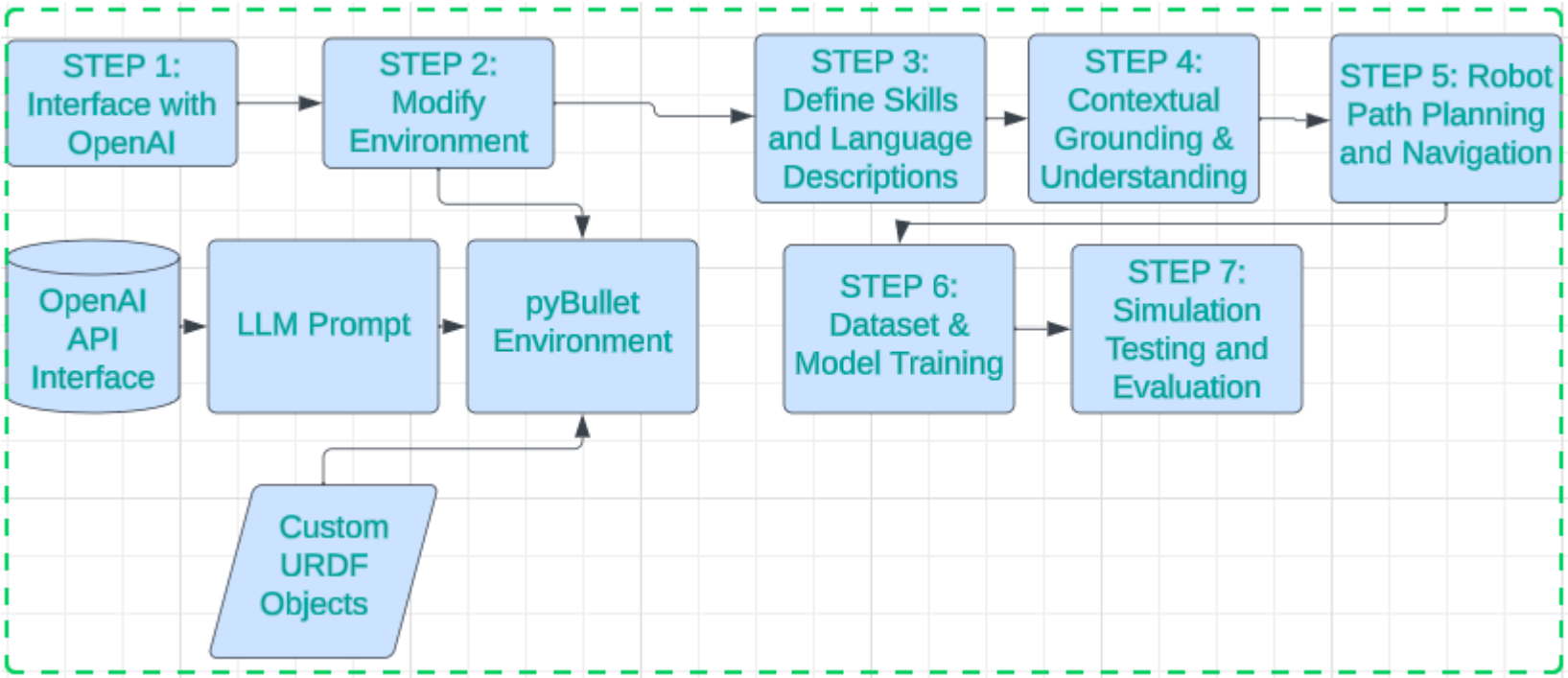
Iterative Task Execution

User instructs, robot interprets and acts, then awaits further instructions.

Skill Probability Scoring

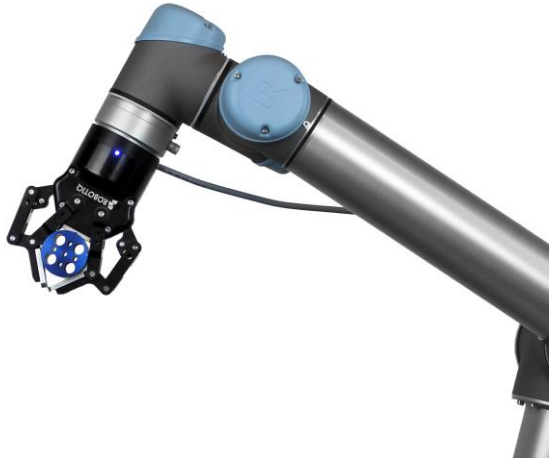
Display of skill selection using combined probability scores.

Approach and Methodology



Robot - UR5e and Gripper 2F85

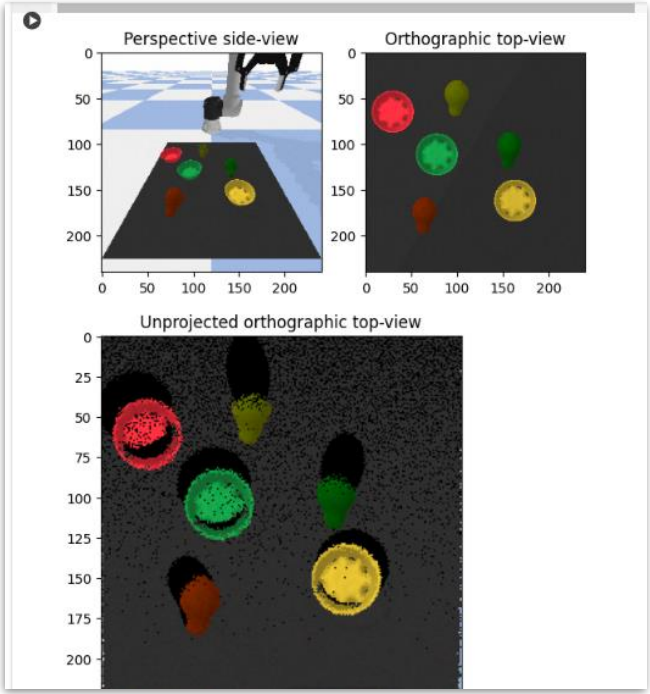
- The UR5e is a lightweight, adaptable collaborative industrial robot designed for medium-duty applications. It offers ultimate flexibility and seamless integration into a wide range of applications
- **Technical Specifications**: **Payload**: Up to 5 kg (11 lbs).**Reach**: 850 mm.**Repeatability**: 0.03 mm.
- Gripper 2F85 is an adaptable gripper that can be attached to universal robots. **Stroke**: 85 mm **Grip Force**: 20 to 235 N (4.5 to 50 lbf) **Form-fit Grip Payload**: 5 kg (11 lbs) **Friction Grip Payload**: 5 kg (11 lbs) **Gripper Weight**: 0.9 kg (2 lbs) **Closing Speed**: 20 to 150 mm/s (0.8 to 5.9 in/s).

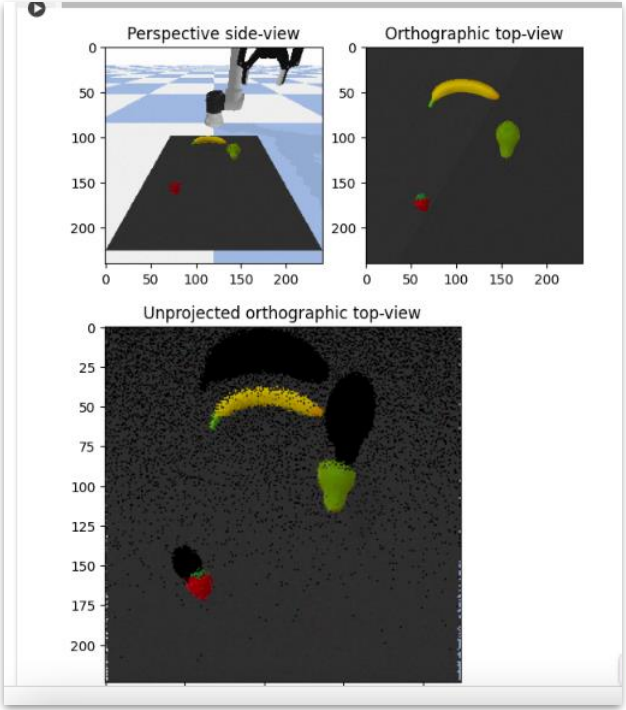


Technical Implementation

1. Implemented a Tabletop pybullet environment with a UR5e and 2-finger gripper, using forward and inverse kinematics.
2. Experimented with loading different objects from open source object collections
3. Successfully loaded household objects from the YCB Dataset.
4. Used a pre-trained ViLD model, based on OpenCV to perform object recognition.
5. Fine-tuned the pick and place primitives for gripping the non-standard objects used in our implementation.
6. Implemented communication with OpenAI API for LLM usage
7. Extended the block & bowl tabletop template on our objects
8. Tried different LLM models such as ‘text-ada-001’, ‘text-curie-002’, and ‘text-davinci-002’
9. Used a pre-trained CLIP model to make combined embeddings for image and text
10. Provided ‘gpt-context’ based on the required configurations for SayCan
11. Tried to execute end to end model for SayCan

Experimental Tabletop Setup





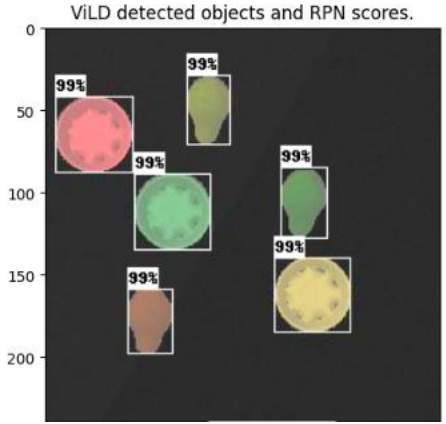
Key Challenges and Solutions

- Introducing new objects with correct orientation and collision properties.
- CLIPort to get new dataset, using new generated dataset, 0 reward error, RL is not implemented properly for new objects.
- Training with new objects/dataset not sufficient (we ran for 300 epochs, 3-4 objects, Required is around 40000 epochs, actual paper uses NVIDIA DGX)
- Open AI API usage issue (Rate limit Error)

Results - ViLD Object Detection

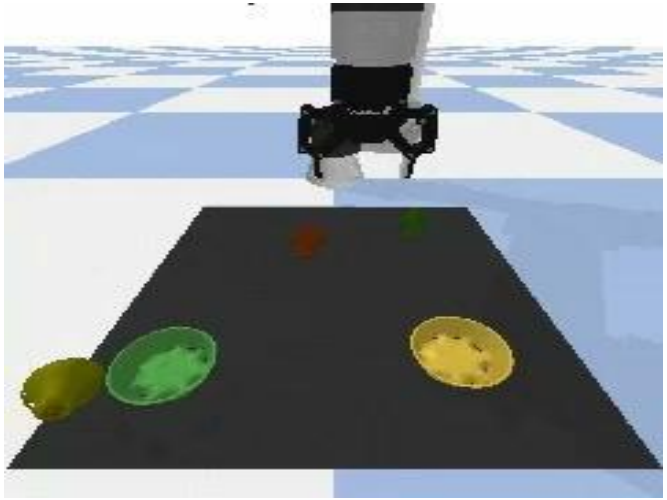
```
Building text embeddings...
100%|██████████| 8/8 [00:00<00:00, 43.97it/s]
Found a yellow circular bowl with score: 0.33460823
Found a green circular bowl with score: 0.32902828
Found a red circular bowl with score: 0.31244054
Found a green fruit with score: 0.3059148
Found a green fruit with score: 0.28679696
Found a red fruit with score: 0.26144752
WARNING:py.warnings:<ipython-input-172-59421901b57c>:83: DeprecationWarning: getsize
  display_str_heights = [font.getsize(ds)[1] for ds in display_str_list]

WARNING:py.warnings:<ipython-input-172-59421901b57c>:94: DeprecationWarning: getsize
  text_width, text_height = font.getsize(display_str)
```



Results - Prompt based Direct Manipulation

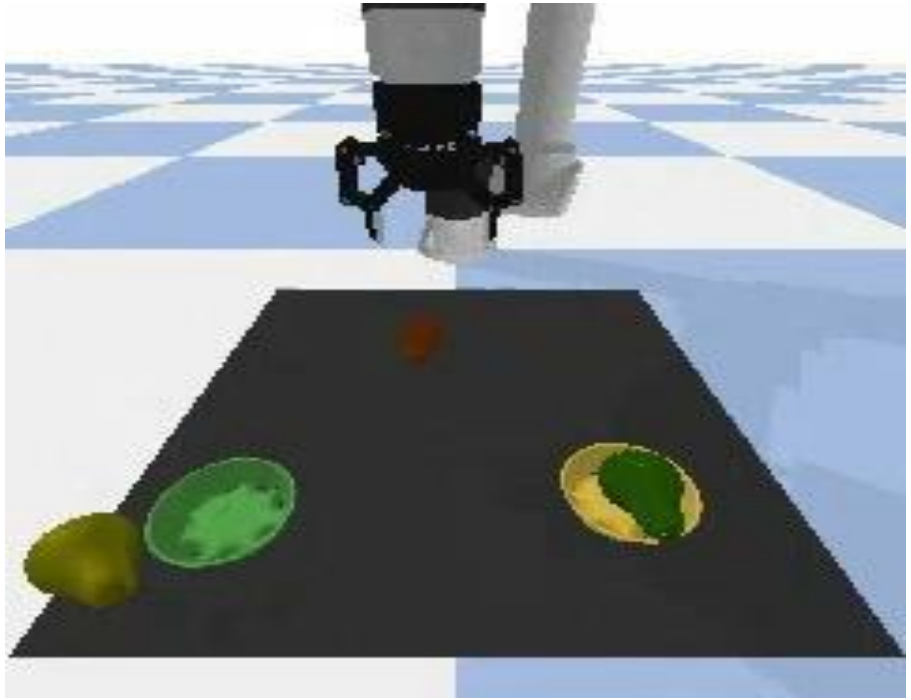
Input: Pick the green fruit and place it on the yellow bowl.



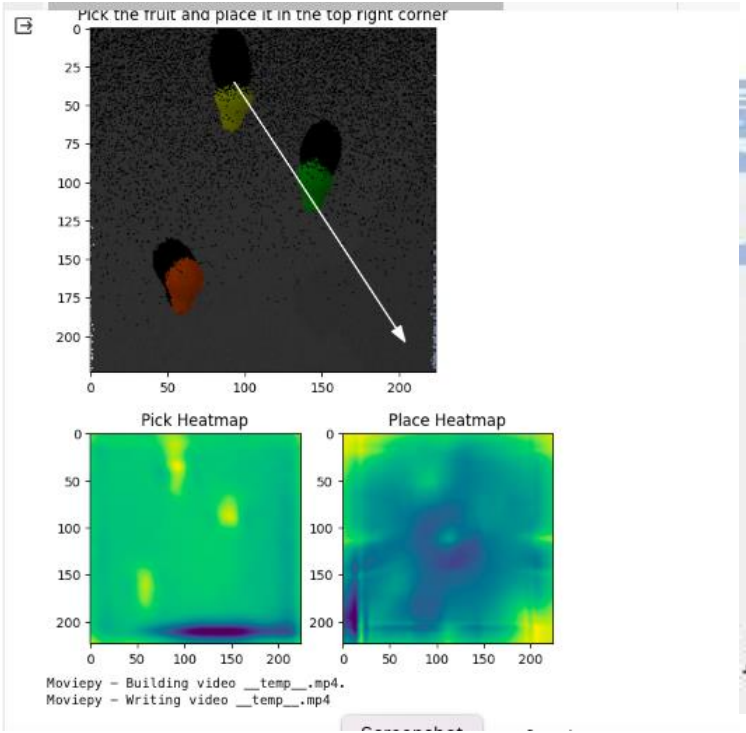
Input: Pick the green fruit and place it on the bottom left corner.

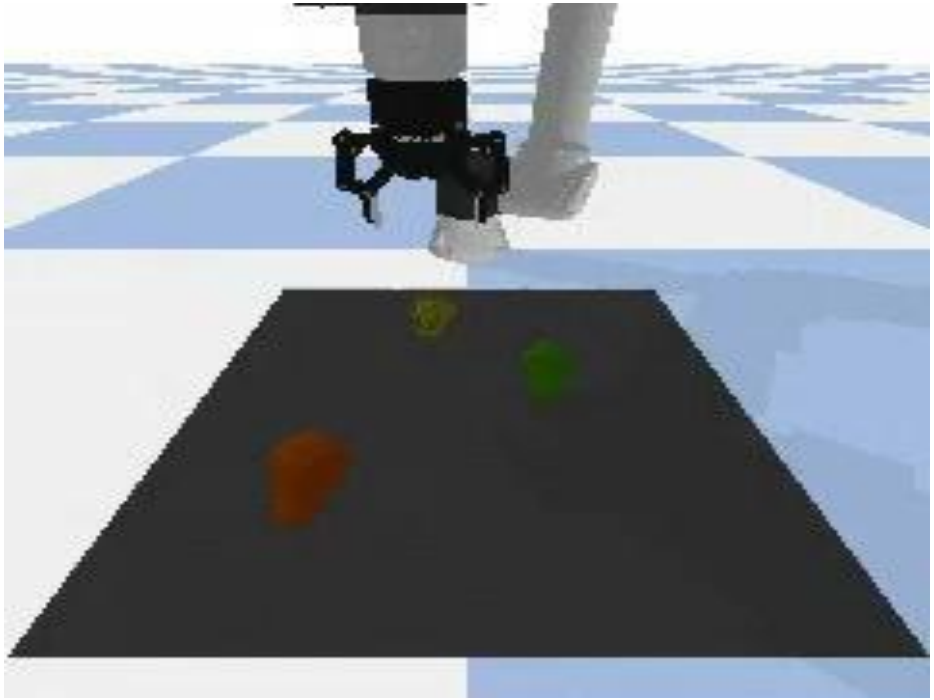
Results - Prompt based Direct Manipulation

Input: Pick the red fruit and place it on the green bowl.



CLIPort Based Demo





SayCan - LLM Scoring Example - Curie

```
[ ]  query = "To pick all fruits and place it on their same colored bowls, I should:\n"
     options = make_options(PICK_TARGETS, PLACE_TARGETS)
     scores, response = gpt3_scoring(query, options, engine='text-curie-001', limit_num_options=24, option_start='\n', verbose=True)

     Considering 24 options
     Scoring 24 options
     cache hit, returning
     -67.574663860178          robot.pick_and_place(green fruit, green bowl)
     -67.85551619967801        robot.pick_and_place(red fruit, red bowl)
     -68.23578072867801        robot.pick_and_place(yellow fruit, yellow bowl)
     -68.49527550967801        robot.pick_and_place(yellow fruit, green bowl)
     -70.265747439678          robot.pick_and_place(yellow fruit, red bowl)
     -70.501145216678          robot.pick_and_place(green fruit, red bowl)
     -71.07650843667801        robot.pick_and_place(green fruit, yellow bowl)
     -72.733631177678          robot.pick_and_place(red fruit, green bowl)
     -73.528250517678          robot.pick_and_place(red fruit, yellow bowl)
     -88.90312651767802        robot.pick_and_place(red fruit, middle)
     -89.51515781767802        robot.pick_and_place(red fruit, top left corner)
```

SayCan - LLM Scoring Example - Ada

```
query = "To pick all fruits and place it on their same colored bowls, I should:\n"
options = make_options(PICK_TARGETS, PLACE_TARGETS)
scores, response = gpt3_scoring(query, options, engine='text-ada-001', limit_num_options=24, option_start='\n', verbose=True)
```

```
Considering 24 options
Scoring 24 options
-97.85414741999999      robot.pick_and_place(green fruit, middle)
-98.58011295159999      robot.pick_and_place(yellow fruit, red bowl)
-100.52327231999999     robot.pick_and_place(green fruit, red bowl)
-102.74691601999999     robot.pick_and_place(green fruit, green bowl)
-103.0503400766         robot.pick_and_place(yellow fruit, green bowl)
-103.2662361516         robot.pick_and_place(yellow fruit, yellow bowl)
-104.23028945159999     robot.pick_and_place(yellow fruit, middle)
-104.49452375159998     robot.pick_and_place(yellow fruit, bottom right corner)
-104.60010391999998     robot.pick_and_place(green fruit, bottom right corner)
-105.3210558996         robot.pick_and_place(red fruit, green bowl)
-105.77874771999998     robot.pick_and_place(green fruit, yellow bowl)
```

SayCan - LLM Scoring Example - davinci

```
query = "To pick all fruits and place it on their same colored bowls, I should:\n"
options = make_options(PICK_TARGETS, PLACE_TARGETS)
scores, response = gpt3_scoring(query, options, engine='text-davinci-002', limit_num_options=24, option_start='\n', verbose=True)
```

```
Considering 24 options
Scoring 24 options
-53.761287392353        robot.pick_and_place(red fruit, red bowl)
-58.472346540852996     robot.pick_and_place(yellow fruit, yellow bowl)
-60.404859791853        robot.pick_and_place(green fruit, green bowl)
-65.187291882353        robot.pick_and_place(red fruit, yellow bowl)
-65.534864932353        robot.pick_and_place(red fruit, green bowl)
-69.70216333785301      robot.pick_and_place(green fruit, yellow bowl)
-70.187875517853        robot.pick_and_place(green fruit, red bowl)
-70.937570751853        robot.pick_and_place(yellow fruit, green bowl)
-71.510847011853        robot.pick_and_place(yellow fruit, red bowl)
-74.93217164205299      robot.pick_and_place(red fruit, bottom left corner)
-75.99270968235301      robot.pick_and_place(red fruit, top left corner)
```

Future Work and Improvements

1. Implement SayCan on new objects.
2. Train the CLIP/CLIPort models for accurate affordances
3. Look into RL based policy for learning in the environment.
4. Create an end to end working system utilising all the components (ViLD, CLIP, SayCan).

Conclusion

We extended the concepts of integrating LLMs with robotic actions in a simple tabletop environment. We have experimented on different object setups and different LLM configurations. The integration of the semantic knowledge and the real world ability of robots will be a useful tool in realizing smarter machines and systems both at home and in commercial applications.

References

- https://say-can.github.io/
- https://cliport.github.io/
- https://arxiv.org/abs/2104.13921(Vild)
- https://www.ycbbenchmarks.com/