Design Oriented Project

on

VLM-Based Robotic Grasping System for Medication Management for Visually Impaired Individuals

End Semester Report Submitted By

Akshaya Venugopal (2021AAPS0022P)

Under the guidance of

Dr. Avinash Gautam

(CSIS Dept, BITS Pilani)

Table of Contents

1. Introduction

The ViperX-300 6 DOF Robot Arm, part of the Interbotix X-Series, by Trossen Robotics features advanced DYNAMIXEL X-Series Actuators for enhanced torque and compact design. This project of object segmentation aims to improve on the work done in the previous semester on object detection and pick up.

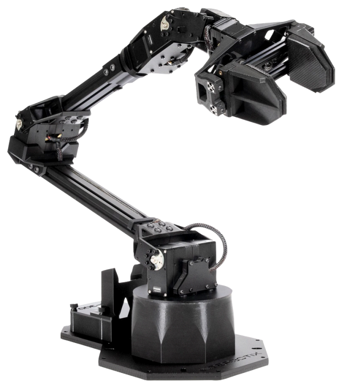

Fig1: ViperX-300 6 DOF Robotic Arm

In the first semester, the ViperX 300 robotic arm was used to detect, pick and place objects in a given workspace. At that time, the object was detected using an overhead camera and OpenCV for colour thresholding to detect the object. But this method had some downsides and wasn't applicable to the purpose of object tracking. In the second semester, I tried to incorporate the use of modern techniques in object tracking and detection to smoothen the process of object tracking in real time.

Now in this semester, using all the learnings from the previous semesters, I have worked on creating a system for disabled individuals to manage complex prescriptions.

The main software used in the previous semesters is Python, OpenCV and ROS. In this semester, additionally we have used YOLO and VLNs. In terms of hardware we additionally have the weight scale, the rotating platform and the Intel Realsense mounted to the arm.

2. Project objectives

This project was intended to focus on developing an integrated VLM-based robotic system tailored to medication management for visually impaired users. The scope includes:

1. Audio Command Interface: Designing an interface to receive and process voice commands to control the robot's actions.
2. Medication Identification: Implementing VLMs and computer vision techniques to accurately identify medication bottles and read labels, including expiry dates.
3. Robotic Grasping and Sorting: Developing grasping algorithms to handle various medication bottles, sort them based on prescription requirements, and manage administration schedules.
4. Delivery Mechanism: Creating a system that delivers the correct medication to the user based on their input, ensuring safe and precise delivery.
5. System Testing and Safety: Ensuring high accuracy in label recognition and medication handling to prevent errors and enhance user safety.

Out of these objectives we have managed to successfully implement the first three objectives with a level of accuracy. In the future, I hope this work is continued and all the objectives are reached.

3. Technical Details

-All of the code for this project was written in Python, and done on an intel NuC system with Ubuntu 20.04 and ROS Noetic.

-To run the whole system follow the following steps:

1. Make sure the ViperX -300 6DOF arm is connected to the NuC and the weight scale and other components are setup
2. Run the following command "roslaunch interbotix_xsarm_control xsarm_control.launch robot_model:=vx300s" on the terminal
3. In a new terminal window run "cd /home/alpha3/Desktop/Arm_project/24-25 sem1/Final Pipeline"
4. Finally run "sh recorder_script.sh"

   Now the system will wait for a new audio recording before it starts executing. All the user has to do is press the button on the microphone to start recording the audio and press the button when the recording needs to end and the system will handle the remaining process.

-All the files related to the project is saved in the following directory of the Intel NuC alpha3 /home/alpha3/Desktop/Arm_project/24-25 sem1/Final Pipeline

-The images folder in this directory contains the captured images from the realsense for the Panorama generation, and images for Aruco detection at different instances

-The old_files folder contains previous transcribed audio files.

-The runs folder contains the YOLO detection results of different runs

-The uploads folder contains the new audio recording every time the user presses the microphone after running the record.py script.

-The best_data.pt file is the trained YOLO model for identifying the medicine bottles

-prompt.txt contains the prompt for the the ChatGPT API call

-panorama.png the panorama generated as a result of the image stitching

-test.mp3 is the audio file generated as a result of google tts to speak the output as audio.

-The README file contains details about the individual scripts.

Description of individual python scripts:

- ans.py : The final code that is executed on the arm. This script is automatically re-written for every new voice command
- arm_motion.py : This is a reset for the arm, in case it gets stuck in any position, running this script will return it to its sleep pose.
- arm_movement.py : This contains the two functions pick_up() and drop_off() which take x, y positions as inputs to perform the required tasks.
- check_shelf.py : This contains a function check_shelf() which checks the shelf and returns whichever shelf slots are empty
- digits_func.py: This contains a function digit_read() which reads the digits of the 7 segment display weight scale.
- identify_unkown.py: This contains a function identify_new() which can pick up a bottle, put it on the rotating platform, generate a panorama of the label of the bottle and using the VLN find out the name of the medicine.
- name_meds.py: This contains a function name_meds() that uses the VLN to read text from an input image
- origin_finder.py: This contains a function find_aruco_center() which finds the Aruco marker of the weight scale to crop the number on the scale as needed
- origin_realsense.py: This contains a function find_origin() which can calculate the origin in pixels from the overhead frame
- realsense_yolo_func.py: This takes the arm to a base pose from which YOLO will be used to detect and find the positions of different bottles on the table
- recorder.py: This script waits for the mic to be active, records audio and saves it.

- robot_func2: This contains the functions pick up() and get_position() which can be used by the ChatGPT API calls to pick up the bottles
- shelf_aruco.py: This contains a function find_aruco_shelf(image) which takes an input image, finds all the Aruco markers in the image.
- shelve_meds.py: This contains a function shelve_meds(day) which puts a bottle on the Aruco marker corresponding to the specific day
- Speech_pipe.py : This contains two functions, one to transcribe audio to text, and one to send the text to ChatGPT for it to write code based on it.
- tts2.py : This contains the function tts() which converts any text to speech using the google text to speech module
- weight_check_function.py: This contains the function check_weight() which checks the weight of any bottle.

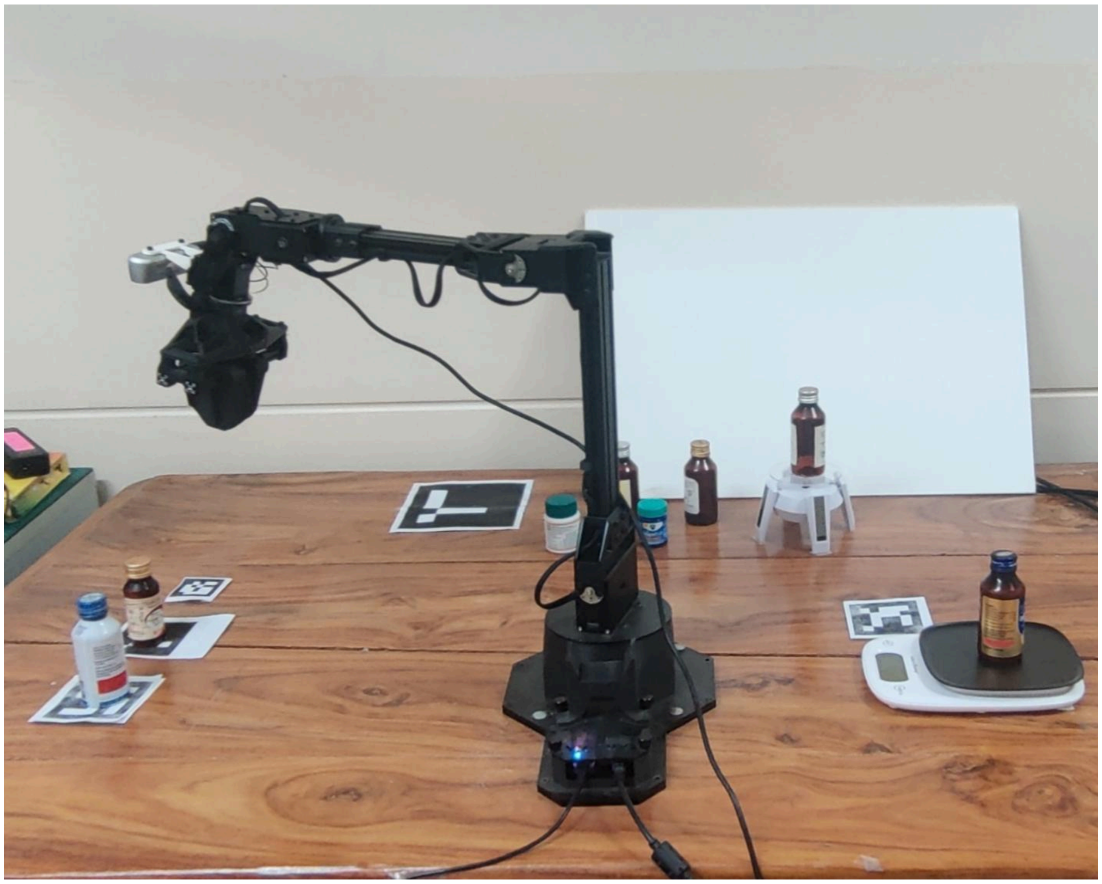

Fig 2: Photo of the system setup

4. Problems Faced

- For the highest streaming quality of the RealSense, the USB Cable used must be a USB 3.0 cable. Other cables are not able to give this high resolution, and can throw an error “Failed to resolve requests” in any of the codes that invoke the RealSense.
- The Aruco Markers are at times mistaken for the medicine bottles but this was later rectified with the better trained YOLO Models.
- The panorama generation can fail if the alignment of the bottle is slightly off at times (1 in 15 runs)
- The VLN sometimes can’t read the image if the panorama is blurry, but this is usually an issue with the panorama generation itself.
- Sometimes the Aruco markers are not fully detected in the shelving code. That was mostly an issue with the print of the Aruco Markers.

5. Scope for future work

While the system we implemented has certain features, there is still scope for improvement in various aspects:

- The integration of the YOLO with the arm for picking up the unknown bottles still needs some refinement. The interpolation is not always accurate.
- The system of sorting medicines on the table needs to be linked to the database, so that a caretaker can keep track of the medicines that are available.
- The expiry date reading system needs to be implemented. I attempted to implement it, but there were issues with the quality of the images that were being captured by the RealSense which was making it difficult to find and read the expiry date accurately.
- The app and the robotic system are yet to be linked. They work independently, but the data from the app needs to be modified by the local system and the app should be able to control the arm remotely. We tried to implement this, but were not able to reach completion.
- The field of view that the Arm uses to check for empty bottles is very narrow at the moment, ideally we would want to increase it to such that the arm can detect a new bottle placed anywhere on the table and not just in a fixed location.

6. Acknowledgements

I would like to thank Dr. Avinash Gautam sir for his continued support on this project. I have had a wonderful experience working on this project the past three semesters and I have gained a lot of knowledge in the process.

I would also like to thank the PhD research scholars of the INSPIRE lab once again, for helping me with any issues I had and providing me with possible solutions for any troubleshooting.

I thank Manas Kalra, for the work on the YOLO training and helping on the bottle recognition system and the app development.

I am extremely thankful for the opportunity that I have had to work in the lab and learn so much. I am proud of the work that I have managed to accomplish over these semesters, and that is thanks to the support from the lab and sir. I look forward to seeing someone else take this work forward in the future and I hope to see the fully implemented system one day.

7. List of References

- Rosebrock, A. (2023, March 22). *Recognizing digits with OpenCV and Python - PyImageSearch*. PyImageSearch. https://pyimagesearch.com/2017/02/13/recognizing-digits-with-opencv-and-python/
- Chakraborty, D. (2021, September 29). *OpenCV Contour Approximation - PyImageSearch*. PyImageSearch. https://pyimagesearch.com/2021/10/06/opencv-contour-approximation/
- GitHub repository for the microphone recording code reference https://github.com/spatialaudio/python-sounddevice/blob/master/examples/rec_unlimited.py
- *pyrealsense2 — pyrealsense2 2.56.0 documentation*. (n.d.). https://intelrealsense.github.io/librealsense/python_docs/_generated/pyrealsense2.html#module-pyrealsense2

A personal Notion Page I have created for all links related to this project : https://highfalutin-dirigible-13a.notion.site/Robotic-Small-Object-Pick-and-Place-4a598fd4e8ed4865accec02705514757?pvs=4

The github containing all the code I have written and used in the past semesters:

https://github.com/Aksh248/ViperX300-Arm-Project-INSPIRE-Lab/tree/main