Lebanese American University

MCE550 – Robotics & Intelligent Systems

Instructor: Dr. Noel Maalouf

Project Report

VENBOT: **V**oice-controlled **E**nvironment-aware **N**LP-based **M**anipulator Ro**BOT**

May 15, 2023

Group Members

- Elise Ghanem (201900223, elise.ghanem@lau.edu)
- Mohamad Said Moudallal (201802015, mohamadsaid.moudallal@lau.edu)
- Rita Mroue (201900775, rita.mroue@lau.edu)

Project Overview

Inspired by the intersectionality and versatility of robotics, VENBOT is the product of our experimentation with ROS and Natural Language Processing (NLP). This project relies on both NLP and computer vision and is thus aware of its surroundings, namely a table with Lego bricks of various sizes and colors, and simulated in Gazebo. VENBOT operates such that the user is prompted to provide vocal input which is processed and interpreted as pick-and-place instructions that a UR5 manipulator executes. This achieves the collection and color sorting of the different Lego bricks in the environment.

A video showcasing the principles and operation of VENBOT can be found **here**. This **repository** contains the source code and simulation files along with the steps to set up the environment to try out VENBOT yourself.

Below are some snapshots of the simulation environment featuring VENBOT, with more images included further below.

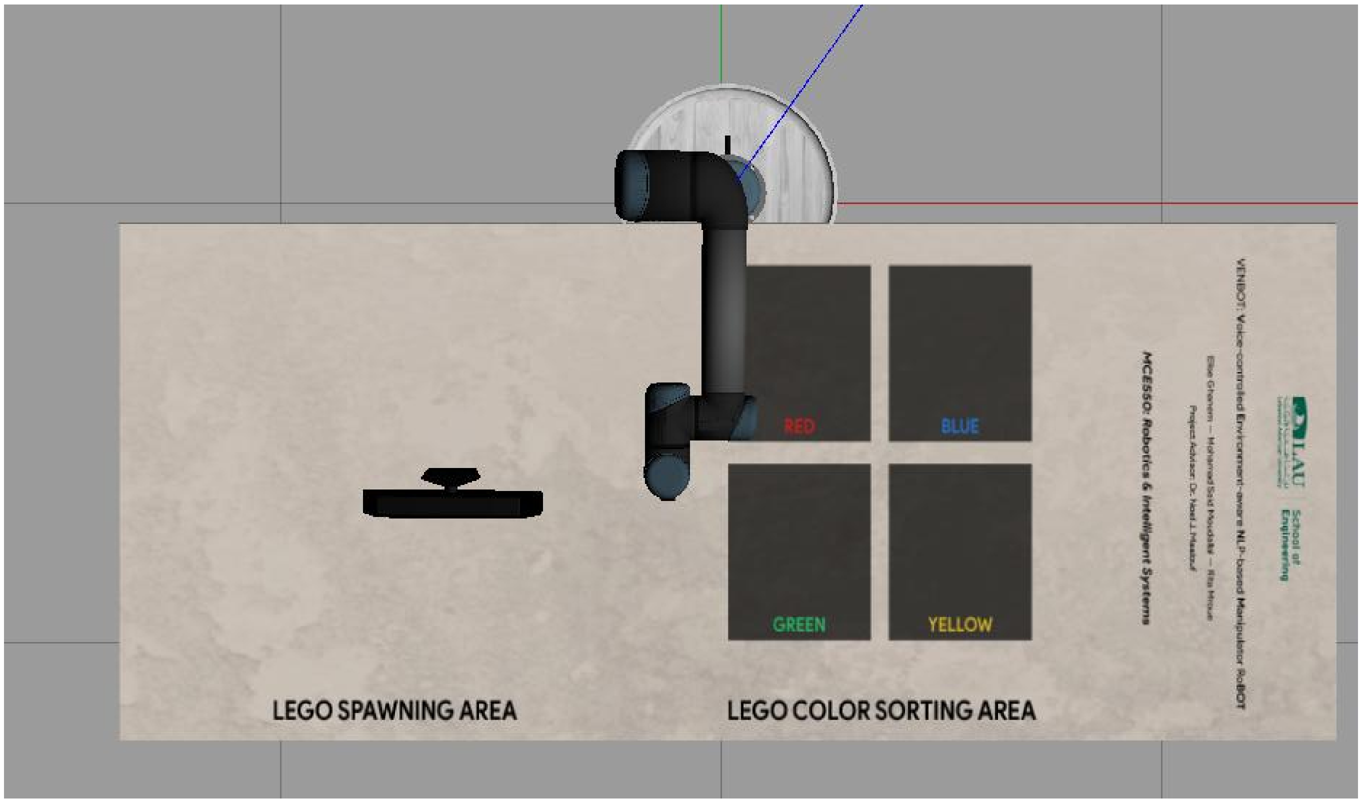

RED
BLUE
GREEN
YELLOW
LEGO SPAWNING AREA
LEGO COLOR SORTING AREA
VENBOT: Voice-controlled Environment-aware NLP-based Manipulator RoBOT
MCE550: Robotics & Intelligent Systems
LAU School of Engineering

Figure 1: VENBOT – UR5 Manipulator & Simulation Environment

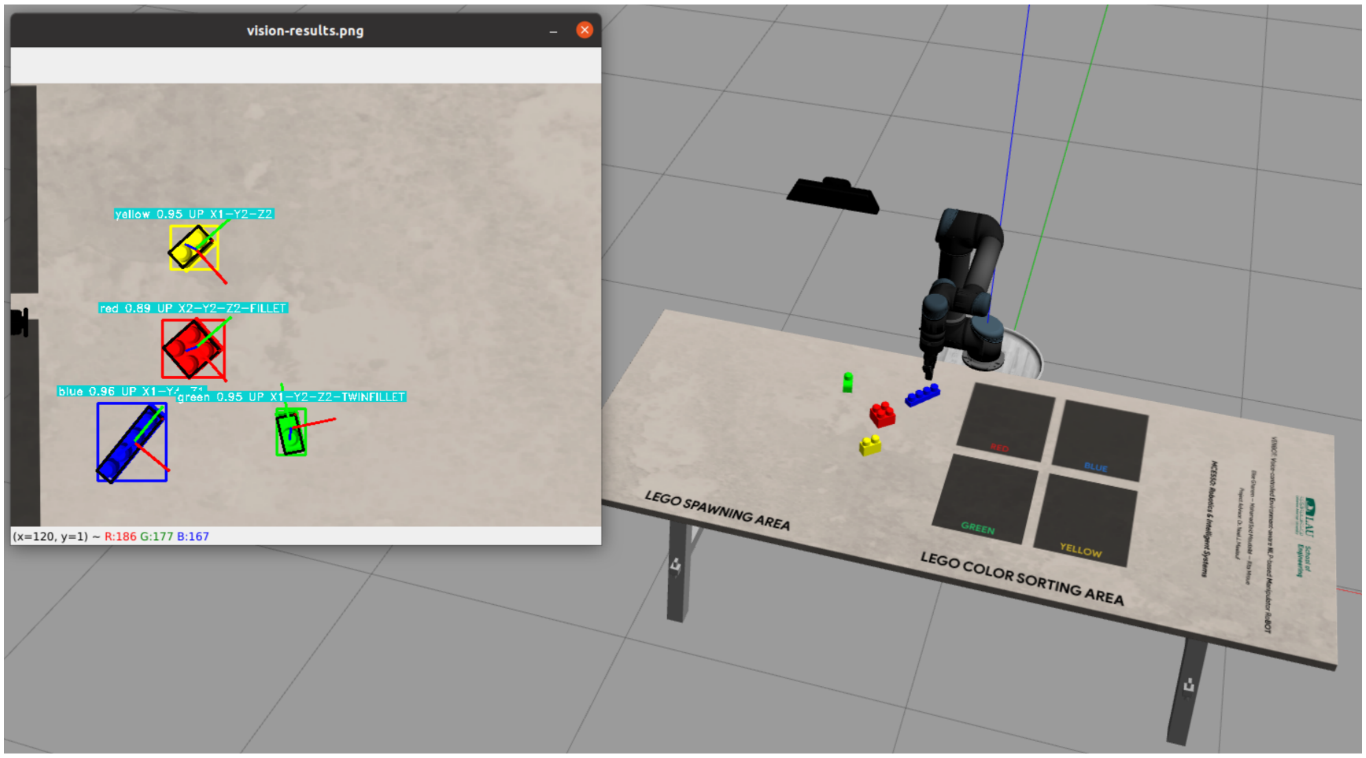


Figure 2: VENBOT – Lego Bricks & Vision Module

Features

The following features are implemented or optimized to fulfill the functionality of VENBOT.

- A pick-and-place environment with a UR5 manipulator and suitable gripper to account for the different types and sizes of Lego bricks
- Computer vision component, fixed camera, recognizing objects, estimating their pose, and providing their color
- Voice recognition code converts user vocal input to text that is continuously displayed for convenience and feedback
- Speech segmentation and processing using NLP by associating the recorded text to pick and place commands while accounting for synonyms and invalid requests
- Interface for performing the processed robot commands in real-time

Project Block Diagram

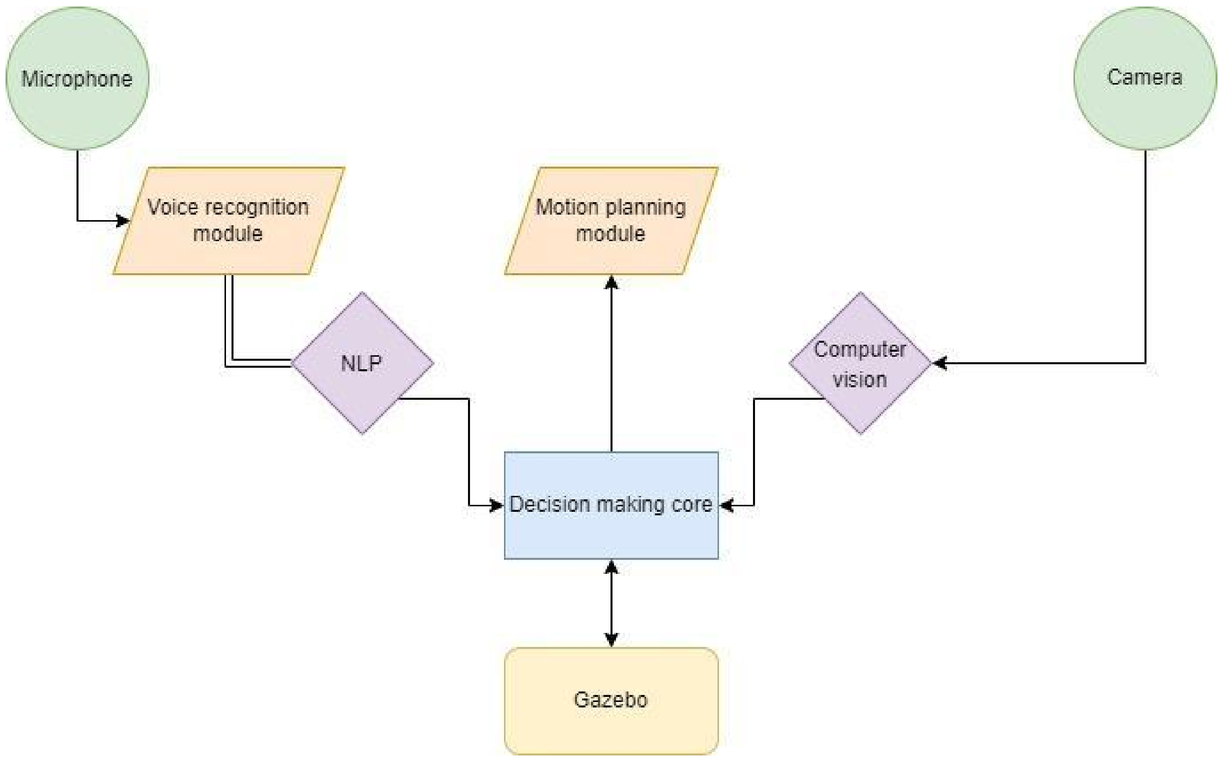


Figure 3: Block Diagram

Approach Followed

1. Review the state-of-the-art technologies, including NLP, computer vision and motion planning in ROS and Gazebo
2. Experiment with off-the-shelf machine learning voice-recognition and NLP techniques
3. Further define the pick-and-place problem and design the corresponding Gazebo simulation world, including a table with defined spawning and sorting areas as well as lego brick models retrieved from an existing project
4. Implement a SOTA robot into Gazebo, the Universal Robot UR5 manipulator
5. Develop the motion planning pipeline for the manipulator robot by customizing pick and place methods
6. Develop a voice recognition module that can accurately detect robot commands
7. Create an NLP interface that processes the chosen commands (pick, place, sort) and responds to invalid requests
8. Develop a computer vision script that recognizes and localizes differently shaped and colors lego bricks, calculates their pose, and returns their grasping point
9. Connect the motion planning pipeline, the NLP pipeline, and the computer vision core using ROS along with user feedback regarding the recorded and processed commands
10. Optimize the motion planning code to respond to invalid trajectories using inverse kinematics and the computer vision module
11. Test VENBOT with different lego configurations through all of its available commands

Development of VENBOT

VENBOT is a convoluted and multifaceted robotic project and an amalgamation of voice recognition, computer vision, natural language processing, motion planning, and actuation in simulation, which need many packages and resources to function properly. Hence, we built VENBOT by following the philosophy of ROS: leveraging open-source software. Apart from open-source software, we had to develop a handful of original packages that ensure the unique functionality of our project. The following lists expand further on the aforementioned packages along with the descriptions of the modifications we made.

Original Packages

- voice_commands: This is a package that we developed from scratch that allows the user to send commands through speech. The package then recognizes and converts the speech into text. The package then performs segmentation on the received texts and publishes the resulting command on a topic, if it was valid. That topic is then subscribed to in the motion planning package.

Open-Source Packages

- Yolov5: YOLO is an object recognition and segmentation open-source library that allows easy and robust detection of trained models.
- OpenCV: This package allows us to use a certain set of tools for image processing, contour drawing, and other functionalities in our vision package.
- spaCy: This library allows us to analyze and process data in the field of NLP. It is used to tokenize the user input and extract the verbal command and color from it.
- Merriam-Webster: Merriam-Webster offers an API that helped us go through and check synonyms for the commands sent through speech to the robot. For example, the verbs

"pick" and "take" should both have the same functionality. This goes on to all the available commands that we offer in VENBOT so far.

- Speech Recognition: This package creates the speech-to-text interface by continuously recording and displaying the user input.
- UR5 Pick and Place: This is an open-source package that was developed by a team at the University of Trento in Italy. It is a pick-and-place simulation that uses Yolov5 and OpenCV to detect different Lego bricks and then place them in their correct positions according to their types using a UR5 manipulator robot and a Robotiq two-finger gripper. The modifications that we made to this package were extensive, to the point where we had to rewrite and refactor hundreds of lines of code. The modifications we made are described thoroughly in the following list.
    - Spawning Lego bricks was modified to spawn Legos belonging to the colors that we need to target in our project.
    - The gazebo world was modified to suit the functionality of VENBOT. The spawning area was changed and new placing areas were created for placing Lego bricks according to their color.
    - The vision package was modified in order to perform color segmentation in addition to object detection and object segmentation. The detected model colors were then published along with the detected model names and poses using a custom message that is built on Gazebo's ModelStates message.
    - The motion planning package was modified and updated extensively. It transforms the main functionality of the original package from automatic sorting to waiting for pick-and-place user commands through speech.

- The sorting functionality was also refactored inside the motion planning package, making it activated through speech.
- Another major contribution that was added to VENBOT is the planned trajectory validity checking algorithm. The algorithm computes the inverse kinematics of the manipulator arm robot and then checks if the calculated solution falls in the range of all of the joint limits that we defined previously inside the URDF of the UR5 robot.

Validation

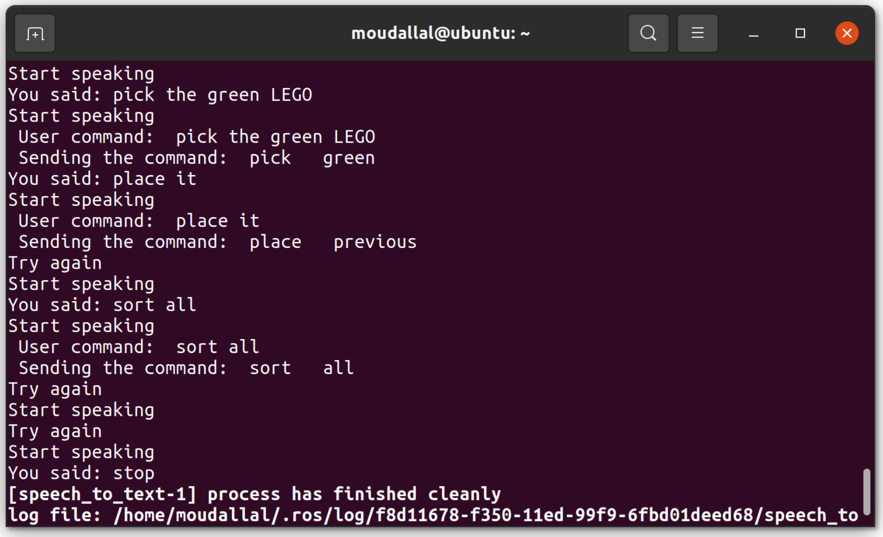
The performance of the robot is assessed after having tested its individual modules separately. These modules are then combined together to create a complete operating, autonomous, and synchronized environment.

First, and after having replicated the simulation environment of the referenced repository along with the installation of the requirements and dependencies, we tested the original motion planning code where the pick and place process is automatic rather than according to the user input.

Then, we tested the created voice recognition module with all of our voices. The user is able to visualize the recorded and processed commands which are then sent to the motion planning module. There, we confirmed the response of the manipulator and the gripper according to the user input.

Despite the slow simulation time, we were able to validate the envisioned performance of VENBOT, creating a voice-controlled and NLP-powered SOTA manipulator capable of color sorting and in Gazebo.

The following are snapshots from our validation process, starting with the voice recognition and command processing.



```
Start speaking
You said: pick the green LEGO
Start speaking
 User command:  pick the green LEGO
 Sending the command:  pick   green
You said: place it
Start speaking
 User command:  place it
 Sending the command:  place   previous
Try again
Start speaking
You said: sort all
Start speaking
 User command:  sort all
 Sending the command:  sort   all
Try again
Start speaking
Try again
Start speaking
You said: stop
[speech_to_text-1] process has finished cleanly
log file: /home/moudallal/.ros/log/f8d11678-f350-11ed-99f9-6fbd01deed68/speech_to
```

Figure 4: Voice recognition and command publishing

The final result following the color sorting and placing of all the picked LEGO bricks is shown below.

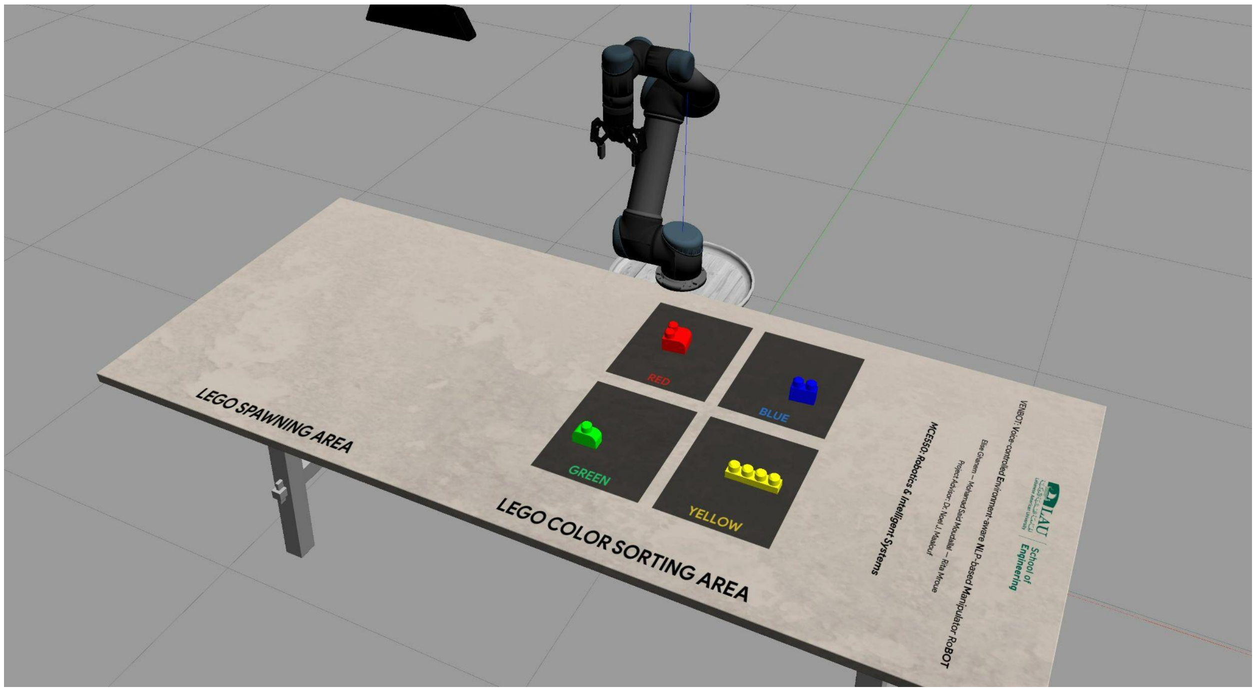

LEGO SPAWNING AREA
RED
BLUE
GREEN
YELLOW
LEGO COLOR SORTING AREA
VENBOT: Voice-controlled Environment-aware NLP-based Manipulator RoBOT
Elise Ghanem – Mohamad Said Moudallal – Rita Mroue
Project Advisor: Dr. Noel J. Maalouf
MCE550: Robotics & Intelligent Systems
LAU
Lebanese American University
School of Engineering

Figure 5: Final Pick-and-Place Sorted Environment

Limitations

- Working in Virtual Machines resulted in slow, and sometimes lagging, performance despite code optimization. For instance, the real time factor in Gazebo was very low.
- The chosen voice recognition library is less problematic than others in terms of integration and is the most accurate among them. However, when processing the command, its response becomes slow and the user has to wait for its module before speaking.
- The available APIs and libraries in the literature have limitations regarding the synonyms they understand, with one library excluding some of the synonyms used in another one, and so on.
- There is no existing functional manipulator arm at LAU, which did not let us test our developed and working-in-simulation code on real hardware.

Future Work

Given that VENBOT combines different technologies (ROS, Gazebo, NLP, Computer Vision, and Voice Recognition), it provides us with room for massive improvement and innovation in these different fields.

- Train the vision models to detect different objects rather than just Lego bricks.
- Extend the range of adjectives that can be given to the robot. e.g.: “pick the closest Lego brick”, “pick the smallest and farthest red Lego brick”, and so on and so forth.
- Placing a picked brick back in its original position is a functionality that can be added to VENBOT.
- This project also opens up future advanced topics such as getting two manipulator arms to work together in order to perform different tasks while being activated by speech.

Conclusion

This project was an opportunity for us to enhance our knowledge and practical experience in manipulating the tools and applying the concepts we encountered in this course. These include ROS scripting, simulation in Gazebo, forward and inverse kinematics among others. We also further experimented with and incorporated Computer Vision and NLP to our Gazebo simulation featuring the SOTA UR5 manipulator. Besides the technical knowledge acquired, another takeaway lies in our improved way of approaching robotics problems, writing more efficient code, and better problem solving and troubleshooting.

References

Simulation

https://github.com/pietrolechthaler/UR5-Pick-and-Place-Simulation/tree/main

https://github.com/utecrobotics/ur5

Voice recognition

https://pypi.org/project/SpeechRecognition/

NLP Core – Tokenization & Synonym extraction

https://spacy.io/usage

https://www.nltk.org/

https://github.com/johnbumgarner/synonyms_discovery_aggregation