Teleoperation of KUKA youBot Using V-REP and Matlab.
CSCK505: Robotics.
Muhammad Ali

This report details the development of software for teleoperating a KUKA youBot R800 in V-REP and Matlab's app designer. Code implementation, sensor options, performance and results of the robot will be analysed and critiqued.

According to both (Chinello, et al., 2010) and (Abdulghani & Al-Aubidy, 2020), V-REP and MATLAB are excellent simulation tools widely implemented both in commercial setting and academia.

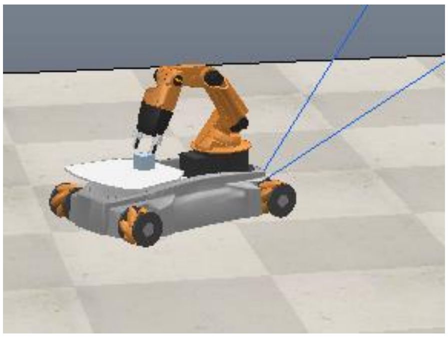


Figure 1: KUKA youBot R800.

In this project, the KUKA youBot is fitted with a visual sensor (visual angle of 90 degrees) as seen in figure 1. The robot is operated via a Graphical User Interface (GUI) developed in Matlab's app designer (figure 2) to efficiently control different types of joints present on the robot.

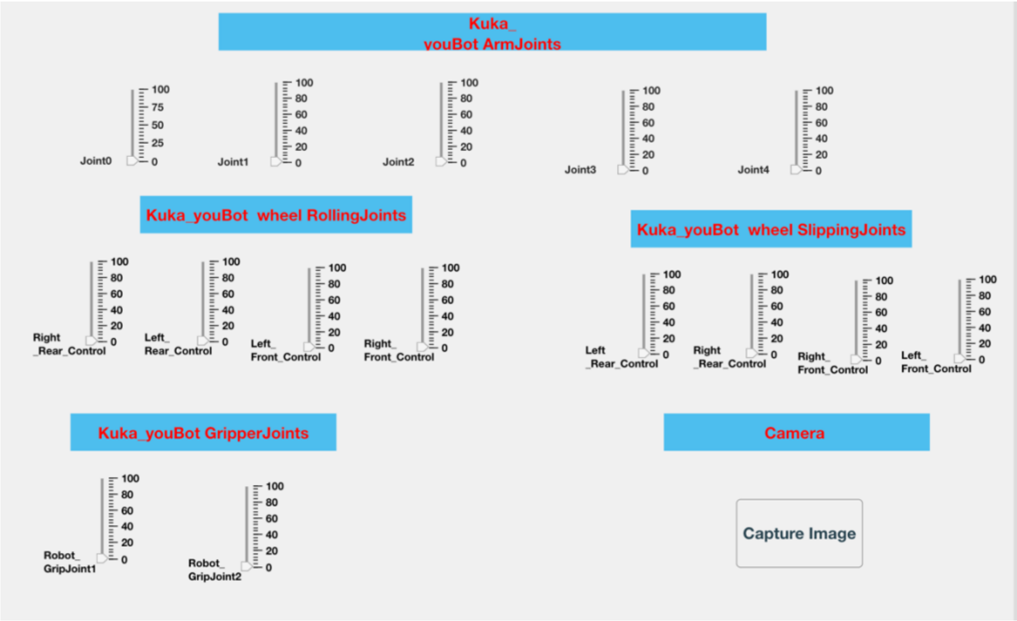


Figure 2: GUI to control youBot via a remote API.

The anatomy of the KUKA youBot is as follows: *4 rolling joints, 4 slipping joints affixed to the base followed by 5 arm joints (seen in figure 2) and 2 gripper joints used for lifting and dropping objects accordingly*.

Coppelia Robotics (formerly V-REP) provides an extensive list of remote API functions to implement varied functionality in different types of robots present in the directory list (CoppeliaRobotics, 2020). The API functions facilitate development of various different robots used both in academic research, as noted by (Chinello, et al., 2010), and industrial setting.

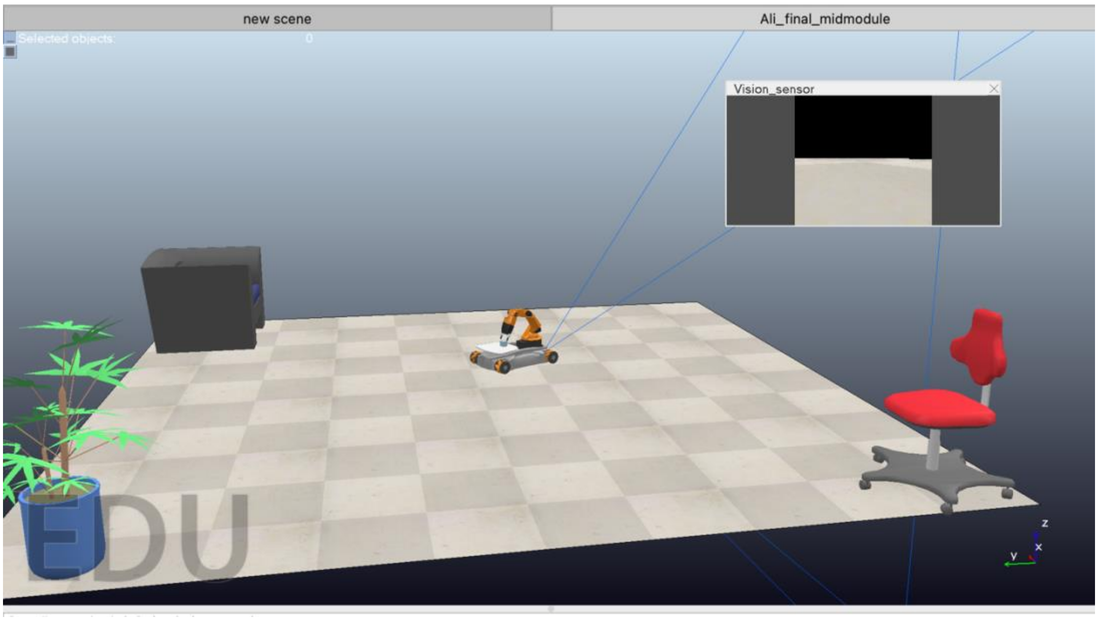
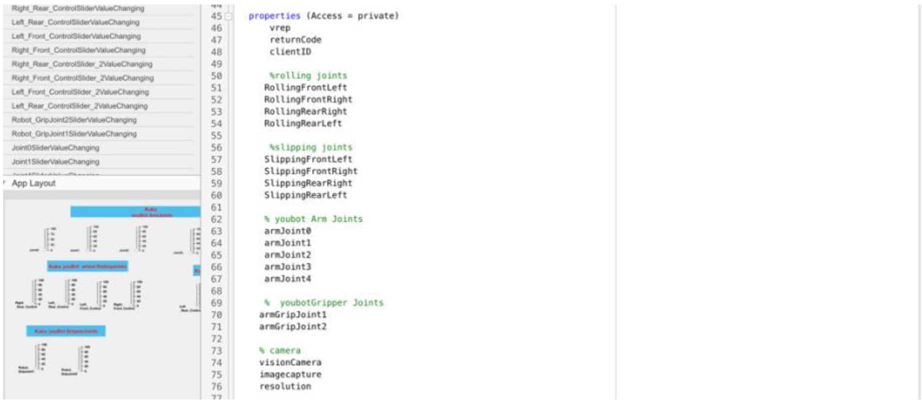


Figure 3: Main scene developed in Coppelia.

Coppelia provides various objects along with different types of robots to create the scene as seen in figure 3. The software also allows various modification to the generic robots such as addition of vision sensor to youBot (implemented in this project).

MATLAB is a powerful commercial software used for numerical computation, statistics analysis and graphical presentation. For this project, MATLAB's app designer is employed to create control buttons for manipulating movement of youBot in V-REP environment. A simple design methodology is preferred (function over aesthetics) which involves different joints controlled by a slider button and images captured of objects present with a simple click (capture image button in figure 2). The simulation is initiated by accessing the Ali_week5.mlapp which connects the GUI created in MATLAB with youBot in Coppelia via remAPi.

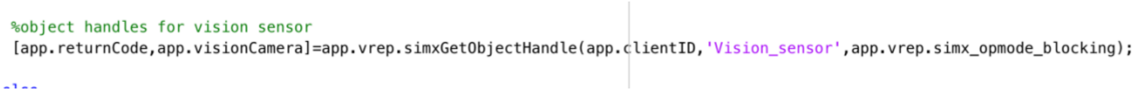


Right_Rear_ControlSliderValueChanging
Left_Rear_ControlSliderValueChanging
Left_Front_ControlSliderValueChanging
Right_Front_ControlSliderValueChanging
Right_Rear_ControlSlider_2ValueChanging
Right_Front_ControlSlider_2ValueChanging
Left_Front_ControlSlider_2ValueChanging
Left_Rear_ControlSlider_2ValueChanging
Robot_GripJoint2SliderValueChanging
Robot_GripJoint1SliderValueChanging
Joint0SliderValueChanging
Joint1SliderValueChanging
App Layout

```
properties (Access = private)
    vrep
    returnCode
    clientID

    %rolling joints
   RollingFrontLeft
   RollingFrontRight
   RollingRearRight
   RollingRearLeft

    %slipping joints
   SlippingFrontLeft
   SlippingFrontRight
   SlippingRearRight
   SlippingRearLeft

   % youbot Arm Joints
   armJoint0
   armJoint1
   armJoint2
   armJoint3
   armJoint4

   %  youbotGripper Joints
  armGripJoint1
  armGripJoint2

  % camera
  visionCamera
  imagecapture
  resolution
```

Figure 4: Properties of the robot.

Object handles, with names of different components shown in figure 4 and 5, are employed to represent joints and objects of KUKA youBot. The remote API functions provided by Coppelia for manipulation are extensively used, vrep.simxGetObjectHandle in figure 5 used for obtaining vision sensor values.



```
%object handles for vision sensor
[app.returnCode,app.visionCamera]=app.vrep.simxGetObjectHandle(app.clientID,'Vision_sensor',app.vrep.simx_opmode_blocking);
```

Figure 5: Vision sensor handle.

The objective of this assignment involved moving different joints of the youBot and capture images to confirm mobility and proper functionality which was achieved with several simulations and changes to settings via the GUI. In the video attached with the report, it can be seen that both wheels and arms joints are moving as intended. In terms if possible improvements, time delay in communication and transmission of higher resolution images should be further researched.

References:

Chinello, F., Scheggi, S., Morbidi, F. & Prattichizzo, D., 2010. The KUKA Control Toolbox: motion control of KUKA robot manipulators with MATLAB. *IEEE ROBOTICS AND AUTOMATION.*

Abdulghani, M. M. & Al-Aubidy, K. A.-A., 2020. *Design and Evaluation of a MIMO ANFIS using MATLAB and V-REP.* Chennai, 10th International Conference on Recent Trends in Communication and Computer Networks.

CoppeliaRobotics, 2020. *Remote API functions (Matlab).* [Online]
Available at:
https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsMatlab.htm
[Accessed 29 November 2021].