GRASP RECOGNIZED OBJECTS IN PYBULLET ENVIRONMENT

Kotur Guna Pragna
S987P265
gxkotur@shockers.wichita.edu

Abstract: In this paper, the robot arm simulation is performed for the grasping of the object and placing the grasped object in the tray. The inverse kinematics function is used for the object loaded and grasping.

I. INTRODUCTION:

Background:

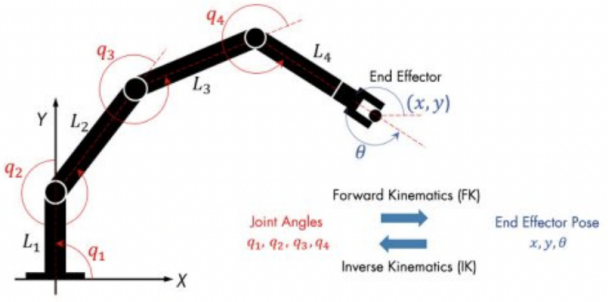
The robot simulation is performed for various automation purposes by the robot manufacturer, engineer for optimizing the throughout the production and improves the workflow of the robot. The simulation is also performed for the robotic tasks which impact the activities that are inward or downward. Simulation is an important technique for testing offline industrial robotic arm programming and reducing difficulties during cell construction. The 3D robotic simulation is performed for designing the cell of the robot to determine the space it is reached without any collison. This is very important for designing the optimal tasks, size and space for the robotic cell.

Understanding Inverse kinematics

Inverse kinematics is a technique for performing the robotic tasks by necessary mathematical programming. This process calculates the orientation and the position of the robotics end effector. The inverse kinematics is quite different from the forward kinematics as follows, If the joints are positioned to a given position, forward kinematics specifies where the end effector will be. The forward kinematic equation can only have one solution. The end effector will always end up in the same place if the joints are configured to a fixed position.

Majority of industrial robots are made up of multiple joints that may be controlled separately. Each joint is linked to one or more other joints in a variety of ways, some of which are complex. The end effector is connected to the kinematic chain at the very end. The robot's end effector travels to a specific location in its workspace, it's only natural to specify your intended position it moves.

The joint positions of a robot arm for configuring forward and inverse kinematics are shown in the below figure[3].



The inverse kinematics is performed in two main approaches which are analytical and numerical techniques. The joint variables are solved analytically based on the configuration data in

the first technique. Joint variables are obtained using computational approaches in the latter approach to solution.

PROCEDURE OF THE PROJECT:

This project is based on the kinematic movement of the robotic arm which is shown using the 3D robotic Simulation. The end effector of the robot arm is moved using the inverse kinematics with the help of the joint arms in the arm. We perform the installation, a suitable environment like python for the robotic arm to grasp the object. The Hand Reading, orientation and palm position are used as the key factors for grasping the object.

The project is performed in the following steps:

1) **Environmental Setup:**

   Installing the Anaconda and the pybullet environment. Pybullet is the open-source implementation for the reinforcement learning in robotics. This also allows for the simulation of it.

2) Install the python3 as the programming environment depending on the system such as Mac, Ubuntu and windows.
3) For the processing of the various robot functions, the pybullet environment is set up using the gym and various commands in the terminal like
   - pip3 install pybullet --upgrade --user
   - python3 -m pybullet_envs.examples.enjoy_TF_AntBulletEnv_v0_2017may
   - python3 -m pybullet_envs.examples.enjoy_TF_HumanoidFlagrunHarderBulletEnv_v1_2017jul
   - python3 -m pybullet_envs.deep_mimic.testrl --arg_file run_humanoid3d_backflip_args.txt
4) Record the Tray location by loading the assigned object in the environment.
5) By adjusting the hand position and grasp position and records the hand, palm position and orientation
6) **Inverse kinematics Function** is used for the hand to approach the object
7) For grasping the object by **Hand direct control function** and placing it in the tray we have the successful trail.

BRIEF SUMMARY OF THE PROJECT:

This project depicts a guide for a robot gripping and recognizing things in a pybullet environment. It goes through the process of producing and labeling RGB photos for object detection model training and testing, as well as applying trained neural networks and PGM for object recognition.

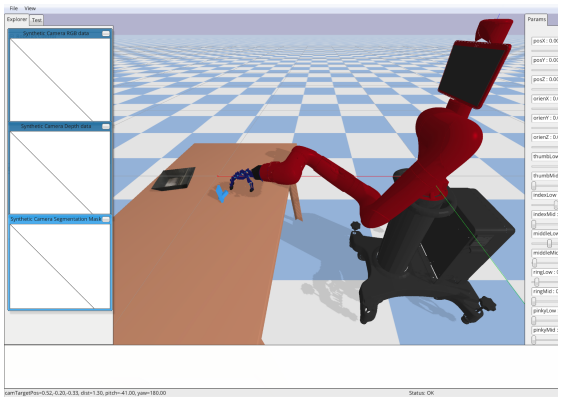
We use the inverse kinematics approach for handling the robotics arm with position and orientation. The objective is to use continual developments in deep learning technologies to construct object hypotheses to verify any hypothesis generated by an object detector and pose estimator. The object positions are then shown to be sturdy enough for a mobile manipulator to approach and grip the thing.

II.PROJECT DESCRIPTION AND RESULTS

Procedure Description

1)Grasp type Detection

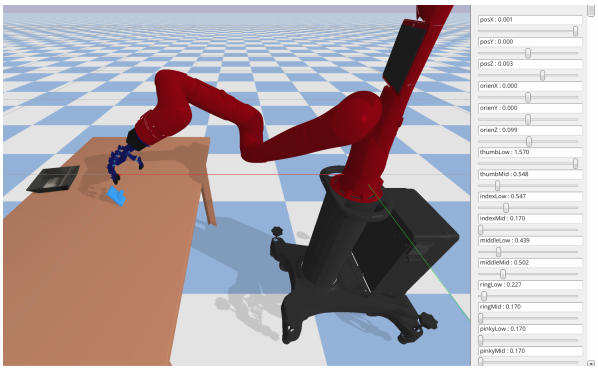
- The grasp topology is decided for our object for the grasping of the object. Using the shape of the object we decide how to control the fingers. The fingers of the robot arm have five functions for the five fingers present. The five fingers include humb, indexF, midF, ringF and pinkyF. The functions of the fingers are given as follows: Thumb is the thumb finger, ringF is the ring finger, indexF is the index finger, midF is the middle finger, and pinkyF is the little finger.



- Each function's range control parameter is specified in the remark. By guiding each finger and grabbing the object, you can develop the type of grasp you want to utilize. Every function of the finger has a control function such as Lower is the control parameter for a finger's lower joint, while Middle is the control parameter for a finger's lower joint.
- Develop the grasping of the object we are using for the grasping by the robot arm by guiding each finger and moving them with the sliders.

2) Controlling the Robot arm towards the object

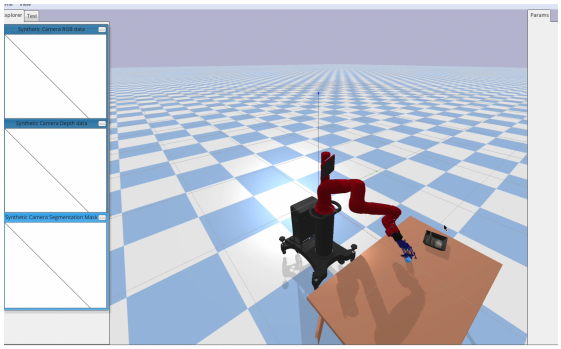
- palmP() is a function used for transferring the palm of the robot arm to its location of the object that is the target location.



- To position the side we want the arm to be changed we use the function TargetP.
- The palm is placed in the xyz plane with the to change the orientation and the angle of the robot arm.
- The vector [x, y, z], where x, y, and z are the target point's coordinates. We build the grip type that we have picked to grasp the object using these functions and setting up the position. The positing of the object helps to create the plane for the movement of the arm towards the object.

3) Assemble the object(picking up)

- Move the palm up after grasping the object with the arm i.e after setting up the orientation and the position of the xyz plane along with the fingers movement.
- If the object grasped is not dropping then the grasp is considered as successful.



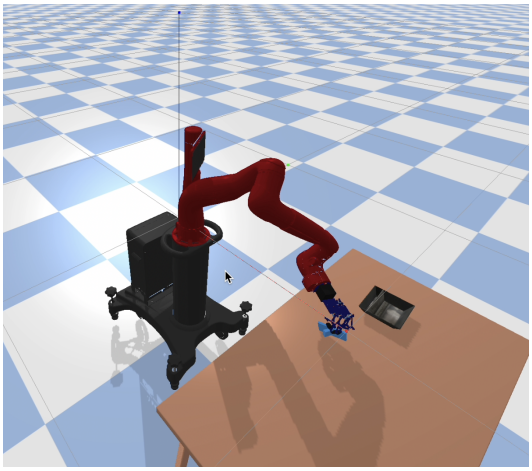
Applied Algorithm of the project.

- The object which the robot arm picks needs to be loaded to the environment.
- We need to record the tray location in the environment. The sawyerEnv.py ensures this is a present tray and object location is recorded.
- The tunegrasping.py is run to adjust and record the hand position, grasp. This gives the output handIReading, palmPosition and Orientation
- Next load the object and tray into the main.py file.
- Update the outputs such as handIReading, palmPosition and Orientation in tunegrasping.py in both main.py and sawyerEnv.py.
- Apply the **Inverse Kinematics Function** in main.py which uses palm position and orientation as inputs
- Use the **Hand Direct Control Function** in main.py for grasping the object with its palmID and FingertipID.
- Finally after the object is successfully grasped move the object towards the tray location by using the tray location and orientation of inputs.
- The successful trail of object grasping is completed.

III. DISCUSSION OF RESULTS:

- After running the sawyerEnv.py and tunegrasping and inserting the values of the object.
- Insert the object position and the orientation and the hand positions of the robot arm

```
reached initial position
reached the target
grasping the object
grasped the object
picking up the object
picking up the object
first_step
Traceback (most recent call last):
  File "main.py", line 355, in <module>
    currentP = palmP([initial_palmPosition[0], initial_palmPosition[1], initial_palmPosition[2]],
  File "main.py", line 93, in palmP
    jointPoses = p.calculateInverseKinematics(sawyerId, 19, targetP, targetOrientation=orientation, jointDamping=jd)
pybullet.error: Error in calculateInverseKinematics
numActiveThreads = 0
stopping threads
destroy semaphore
semaphore destroyed
Thread with taskId 0 exiting
Thread TERMINATED
destroy main semaphore
main semaphore destroyed
(base) gunapragna@Gunas-MacBook-Air midterm_project_cs898ca %
```



IV.CONCLUSION:

Object detection involves identifying instances of a certain type, such as within a certain image or picture. This can be achieved by learning the special features each object has. The Tensorflow Object Detection APIallows you to easily create

or use an object detection models and transferring learning.The main objective of this project -grasping objects in pybullet environment.Prior to this we created images of objects and labelled them.And trained the data to get a accurate output. The output of the project is attached along with the report,which is-the robot grasped the object which was placed on the table successfully and picked it up.

V. REFERENCES:

[1] K. Li and K. Wang, "Robot arm simulation based on model-free reinforcement learning," 2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), 2021, pp. 1104-1107, doi: 10.1109/ICAICA52286.2021.9498063.

[2] M. Farag, A. N. Abd Ghafar and M. H. Alsibai, "Grasping and Positioning Tasks for Selective Compliant Articulated Robotic Arm Using Object Detection and Localization: Preliminary Results," 2019 6th International Conference on Electrical and Electronics Engineering (ICEEE), 2019, pp. 284-288, doi: 10.1109/ICEEE2019.2019.00061.

[3] S. Zhang, Z. Guo, J. Huang, W. Ren and L. Xia, "Robotic Grasping Position of Irregular Object Based Yolo Algorithm," 2020 5th International Conference on Automation, Control and Robotics Engineering (CACRE), 2020, pp. 642-646, doi: 10.1109/CACRE50138.2020.9229933.

[4] Xie Z., Berseth G., Clary P., Hurst J., van de Panne M., " Feedback Control For Cassie With Deep Reinforcement Learning," arXiv1803.05580 (2018).

[5] Y. Ren, H. Sun, Y. Tang and S. Wang, "Vision Based Object Grasping of Robotic Manipulator," 2018 24th International Conference on Automation and Computing (ICAC), 2018, pp. 1-5, doi: 10.23919/IConAC.2018.8749001.