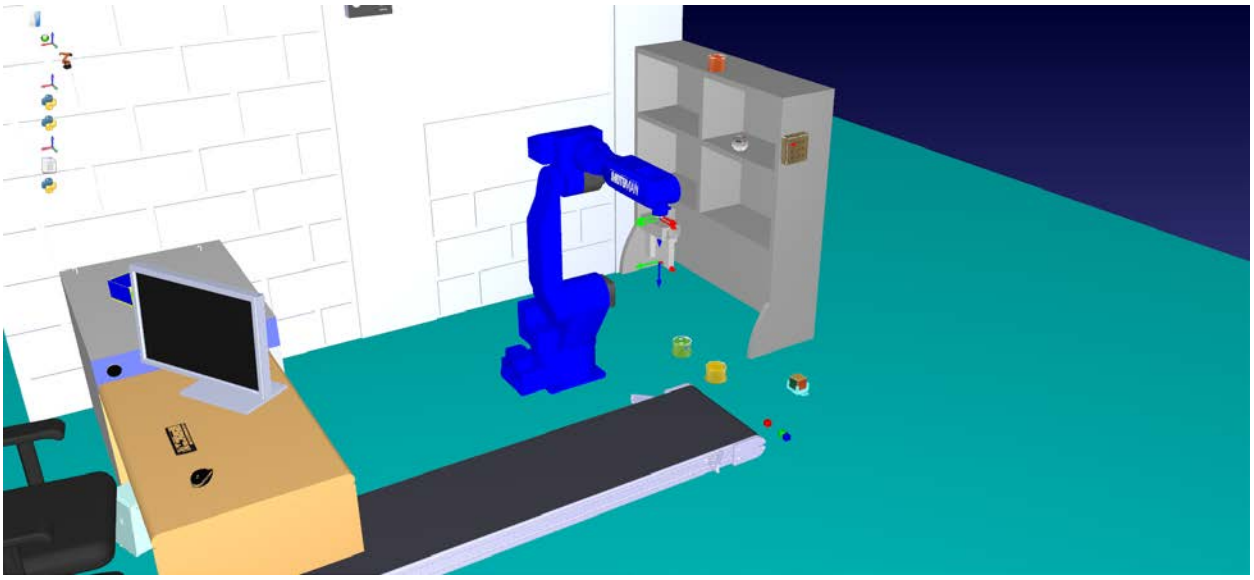
Assignment #3 – Simulation and real-world execution of a simple task

10 points of your final grade

For your third assignment, you are given a RoboDK robot simulator environment. The environment should look like this:



In real life, the environment looks like this:

There are several objects in the environment that the robot can interact with. There are several small tasks you may perform in simulation to gain points in this assignment.

This simulation environment mimics a real robot environment in the Ben-Gurion robotics laboratory. For our lesson on 2.6.2021, I will run your simulations on the **real robot** live on Zoom. Success/failure in the real environment will directly impact your grade, so make sure your simulation is robust.

Submit your work (*.rdk session file) as a single simulation file via the course website by 1.6.2021, 23:59 (China Standard Time).
Important note – Since the assignment will be run on a real robot on 2.6.2021, **there will not be an option to submit late work**. Please keep this in mind.

Four tasks follow. You may perform any tasks you desire, in any order. Your robot's task duration must be **no longer than 4 minutes**. After 4 minutes I will stop the robot.

Note that if you perfectly succeed all of the tasks within the time limit, you can receive up to 24 points. This is, however, quite difficult. I encourage you to focus on only part of the tasks. Remember that the tasks should be performed in simulation as well as real life!

Task #1 - Assembly

There are 4 children's stacking cups in the environment, labeled "Cup G", "Cup H", "Cup I", and "Cup J". Cup G is the smallest, then Cup H, then Cup I, and Cup J is the largest. The cups can fit in one another, when they are "Face up", or stacked on each other when they are "Face down". In the given environment, all four cups are initially "Face down". Note that Cup H is inside Cup I.

Example image:

The task: Stack the cups in a tower.

Reward:

2 cup tower: 2 points

3 cup tower: 3 points

4 cup tower: 4 points

If the tower falls while under construction, you receive the tallest stable structure minus one.

Example: you built a 3 cup tower, and tried to add a 4^{th} cup. When adding the 4^{th} cup, the tower fell. You are therefore awarded 3-1 = 2 points.

Clarification: If you build a 4 cup tower, you receive only 4 points (even though there is a 2 and 3 cup tower before).

Task 2 – Demolition

Assuming you have completed task #1, you may complete the following task.

There is a “Baseball” object on one of the shelves in the environment. The baseball is nearly rigid, and weighs approximately 145g.

The task: Knock down your tower by tossing the Baseball.

Reward:

1 point for every cup knocked over

Clarifications:

- The robot may not touch the tower during this task
- The tower must be knocked over using the baseball
- The robot may not be touching the baseball at the time of impact
- A cup is “knocked over” if it falls from the original tower. The first cup is knocked over if it is no longer “face down”

Task 3 – Assembly

The task: Place the Baseball inside one of the cups.

Reward:

2 points

Clarifications:

- The cup must be "face up" when the baseball is placed inside it
- The cup must remain "face up" after the baseball is placed within it.

Task 4 – Manufacturing Line

There are three rigid cubes in the environment. One is red, one is green and one is blue.

There is also a conveyor belt. The conveyor belt will be moving at a constant speed of 20 mm/s.

At the end of the conveyor belt are 3 boxes, meant to collect red, green and blue cubes, respectively.

When a cube is placed on the conveyor belt in the correct position, it will fall into the collection box.

The task: Place the cubes on the conveyor belt, so that they fall into their respective collection boxes.

Reward:

Two cubes fell into correct collection boxes: 1 point

Three cubes fell into correct collection boxes: 2 points

Three cubes fell into correct collection boxes within 5 seconds of the first: 3 point

Three cubes fell into correct collection boxes within 2 seconds of the first: 5 points

Three cubes fell into correct collection boxes within 0.5 seconds of the first: 7 points

Clarifications:

- Cubes can only be dropped into the collection boxes by the conveyor belt, not the robot
- The robot can place the cubes at any position on the conveyor belt
- If you manage to time the cube placement so that the cubes fall into their collection boxes together, you will receive bonus points. The maximum number of points is 7 (all three cubes fell into the correct collection box within 0.5 seconds)

Task 5 – Dexterity

There is a “9 Hole Peg” object rigidly attached to the shelf. The object has 9 cylindrical holes. The top-left hole has a peg inside of it (see simulation). The vertical/horizontal distance between holes is 33mm.

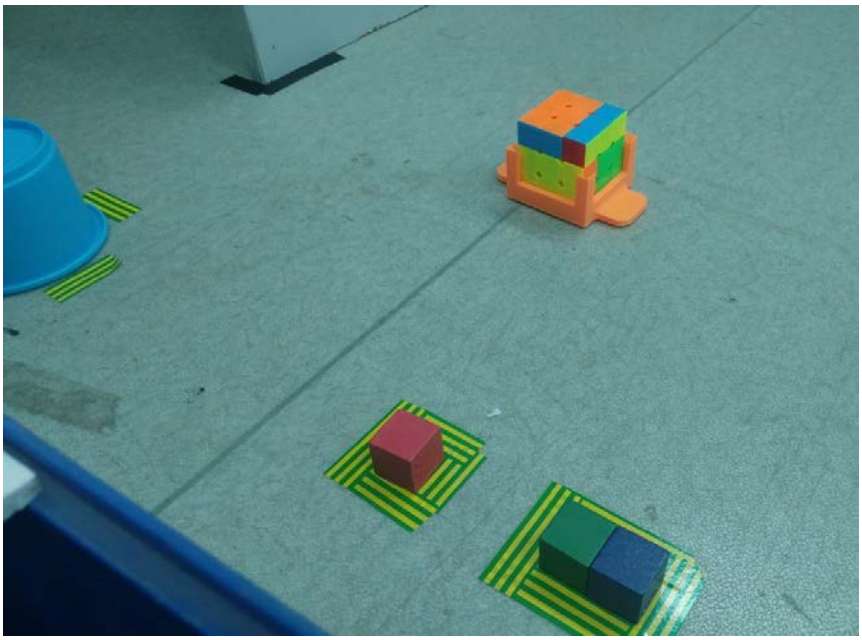
The peg is 40mm long, with a diameter of 6mm. The holes are approximately 22.5mm deep.

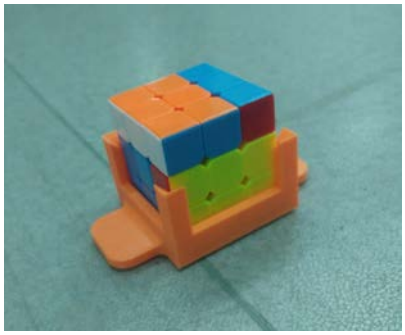
The task: Move the peg to the central hole.

Reward: 3 points

Task 6 – Dexterity

There is a “Rubiks Cube” object inside a 3D-printed “Rubiks Cube Holder” object. The “Rubiks Cube Holder” object is rigidly attached to the floor. The Rubiks cube is not as it appears in the simulation, but rather like such:





Note that the Rubik’s cube is very close to being solved.

When the Rubik’s cube is fully within the Rubik’s cube holder, only the top layer is free to rotate.

The task: Solve the Rubik’s cube.

Reward: 4 points

Clarifications:

- The Rubik’s cube is solved when all six sides have one color each
- The final location/orientation of the Rubik’s cube is unimportant, as long as it is solved

Technical notes and clarifications

1. **Before submitting your work, change the Open/Close Gripper scripts from "Simulation Mode" to "Run on Robot Mode"**

There are two python scripts in the simulation that control the pneumatic parallel-jaw gripper. These are "Open_Gripper.py" and "Close_Gripper.py".

You do not need to understand the Python programming language to complete this assignment. However, you do need to understand the following.

These scripts were written to control the pneumatic gripper though an external microcontroller. The standard way to control a gripper is through the robot's controller, but I decided to go a different way for this project for technical reasons.

By default, the python scripts move the gripper in simulation but not in the physical world. To open/close the gripper in the physical world, you must uncomment line 6 in both scripts. To do this, right-click the python script and press "Edit Script". Then delete the "#" sign at the beginning of line 6. Do this for both scripts before submitting your work!

Note that if you do this without a microcontroller attached to your PC, uncommenting this line will result in an error. This is because the scripts attempt to create a serial connection to a non-existing device. Therefore, there is no reason to uncomment until you submit your work.

2. **The gripper will always fully open or close, regardless of the object being grasped**

This can be changed with quite a bit of work, but there is no need for you to do this. Since RoboDK is not a physics simulator, it's fine if the object collides with the gripper when being gripped. You will have to imagine the correct result, or simulate it manually if you need the precision.

3. **I have included two "Reset" programs in the simulator. They are different.**

It is common practice in RoboDK to include a program that resets the positions of all of the objects. Normally, objects are moved around during the writing of the program, and it makes sense to have a program that automatically resets all of the objects. I have written such a program, called "Reset Objects". When this program is run, all of the objects move back to their original positions

Due to a bug, when RoboDK imports textured .obj files for the first time, they appear with color. After saving and re-opening, the color disappears. Therefore, when you open the simulation for the first time, some of the objects will appear grey. This makes no real difference, but it might be aesthetically unpleasing.

To overcome this, I have written a python script "Initialize Objects.py". This script essentially erases the objects, and re-loads the same objects into the simulation, at the same places. Therefore, when script runs, "Reset Objects" no longer works—because the objects are essentially entirely new objects.

Depending on how you implement your simulation, this might disrupt some of your program. For instance, if you attach a target to an object's reference frame, the target will no longer be linked (because the object has been deleted).

To run "Initialize Objects.py", you will need to extract the folder "YCB Objects" to the same directory that that your simulation is in.

4. **Opening and closing the gripper takes time, and the simulation is not 100% accurate**

When the Open/Close Gripper scripts are run, a command is sent to the microcontroller after roughly 0.5s. Once the command is sent, a solenoid is triggered, and the gripper opens/closes. This is a physical operation and takes time. Keep this into account when designing your tasks. This is especially important for time-dependent tasks such as 2 or 4.

5. **When the robot moved from one target to another, there is some latency due to imperfect control**

In simulation, the robot reaches a target and immediately moves to the next. In reality, there is a pause between the two states because the robot must reach the exact position before continuing. In more advanced programming, there is a solution for this called **rounding**, where the paths are merged into one. We will not be applying this in this assignment, so assume a slight pause (~0.5s) between robot movements.

6. **Many things are slightly different in simulation and reality.**

You are encouraged to request clarifications and even demonstrations in the course website. For instance, you can ask for the weight of the Rubik's cube, or request the robot to grasp a cube and then release it, to see how it behaves in real life. I will perform the task and upload a video. Keep in mind that the earlier you make a request, the more likely it is that I will be able to fulfill it prior to the assignment due date.

Good luck!