PARTICIPANT INFORMATION STATEMENT

The aim of this study is to evaluate GenLfD, a software developed to intuitively teach robots industrial tasks. The study is being conducted by Shirine El Zaatari at Coventry University.

You have been selected to take part in this questionnaire survey because you have learned and attempted to use GenLfD to program a robot in Simulation. Your participation in the survey is entirely voluntary, and you can opt out at any stage by closing and exiting the software. If you are happy to take part, please answer the following questions relating to your experience with using GenLfD. Your answers will help us to identify weak points in the software and improve it.

The survey should take approximately 30 minutes to complete. Your answers will be treated confidentially and the information you provide will be kept anonymous in any research outputs/publications. Your data will be held securely. All data will be deleted by December 2021.

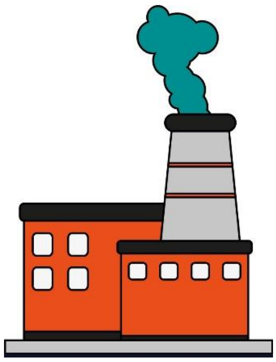
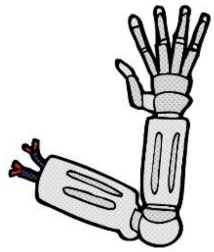
The project has been reviewed and approved through the formal Research Ethics procedure at Coventry University. For further information, or if you have any queries, please contact the lead researcher Shirine El Zaatari at elzaatas@uni.coventry.ac.uk. If you have any concerns that cannot be resolved through the lead researcher, please contact Professor Weidong Li at aa3719@coventry.ac.uk.

Thank you for taking the time to participate in this survey. Your help is very much appreciated.

☐ I have read and understood the above information. I understand that, because my answers will be fully anonymised, it will not be possible to withdraw them from the study once I have completed the survey. I agree to take part in this questionnaire survey. I confirm that I am aged 18 or over.

☐ I agree to evaluate GenLfD, a software developed to intuitively teach robots industrial tasks.

INTRODUCTION

You are a technician in a factory. You are required to program a robot. You have no programming experience.

You are presented with a tool called **GenLfD**, which allows you to program robots for Generic tasks using Learning from Demonstrations.

In this experiment, you are required to:

1. **Use** GenLfD to program a pick-and-place task in Simulation, following this tutorial document. (15-20 minutes)

2. **Assess** GenLfD by answering a few questions.(5-10 minutes)

To be able to run the program, you need to have installed:

1. MATLAB R2020a

2. CoppeliaSim EDU (free download https://www.coppeliarobotics.com/downloads)

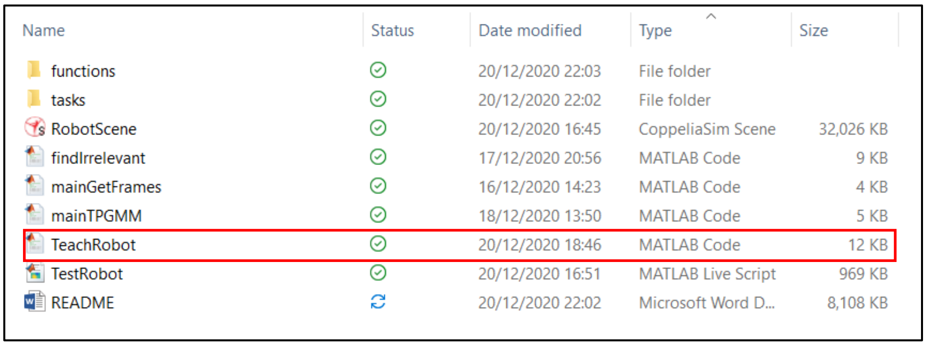
INSTRUCTIONS

Step 1

The tutorial folder includes different files and folder including code functions, simulation files and saved examples. As a user, you will only be using:

- TeachRobot.m, a MATLAB script used to record task demonstrations
- TestRobot.mlx, a MATLAB live script used to test the learnt task model

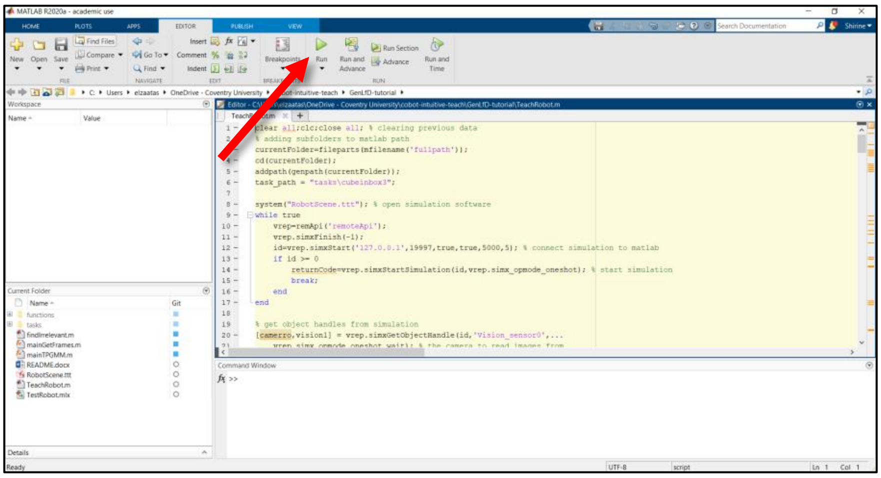
Firstly, open TeachRobot.m with MATLAB R2020a.



Name	Status	Date modified	Type	Size
functions		20/12/2020 22:03	File folder	
tasks		20/12/2020 22:02	File folder	
RobotScene		20/12/2020 16:45	CoppeliaSim Scene	32,026 KB
findIrrelevant		17/12/2020 20:56	MATLAB Code	9 KB
mainGetFrames		16/12/2020 14:23	MATLAB Code	4 KB
mainTPGMM		18/12/2020 13:50	MATLAB Code	5 KB
TeachRobot		20/12/2020 18:46	MATLAB Code	12 KB
TestRobot		20/12/2020 16:51	MATLAB Live Script	969 KB
README		20/12/2020 22:02	Microsoft Word D...	8,108 KB

Step 2

When MATLAB launches, run the TeachRobot script by clicking Run, in the Editor Toolbar.

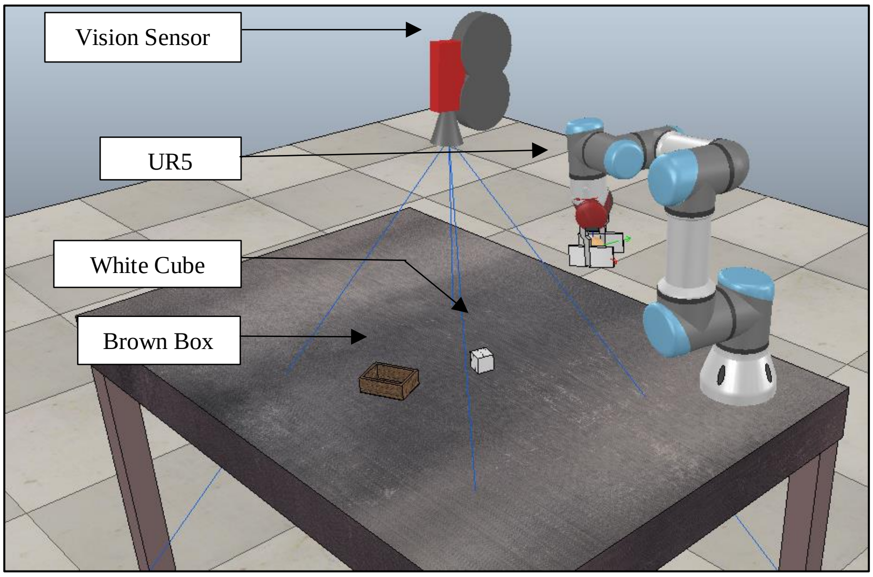


Step 3

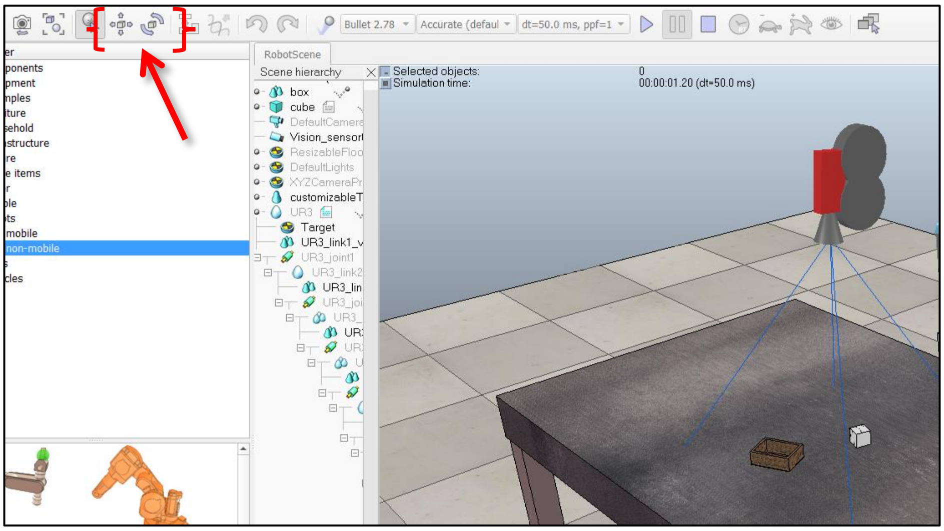
CoppeliaSim EDU simulation software will automatically open to a scene showing, a UR5 robot, a white cube, a brown box and a vision sensor.

The simulation will be used to record 5 demonstrations of the robot picking the cube and placing it in the box. The vision sensor will record a 2D image of the initial table setup of each demonstration.

In each demonstration, the position and orientation of the cube and box should vary, so the robot learns a task model.

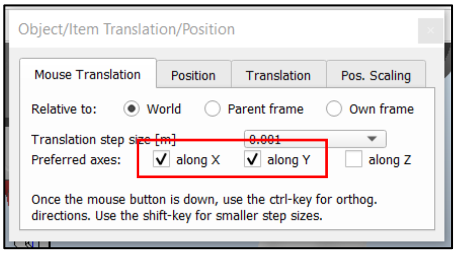


Use the object shift and rotate tools to vary the positions of **only** the cube and box on the table.

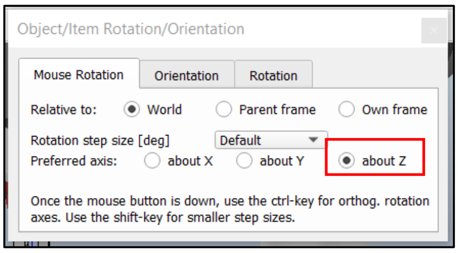


Make sure:

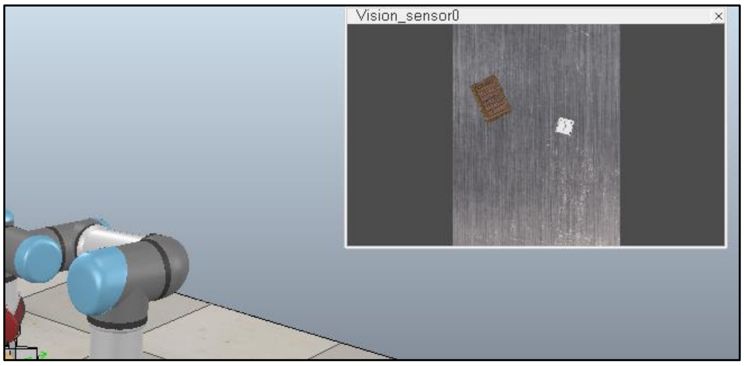
- You don't move the robot, the vision sensor or the table.
- When using the shift tool, enable shifting along the *x* and *y* dimensions only, relative to World.



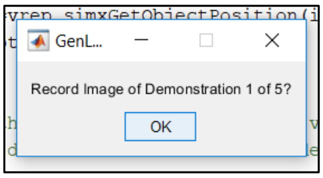
- When using the rotate tool, enable rotating along the *z* direction only, relative to World.



- The objects remain within the field of vision of the vision sensor. You can confirm this in the next step, or in the vision sensor's pop-up output.



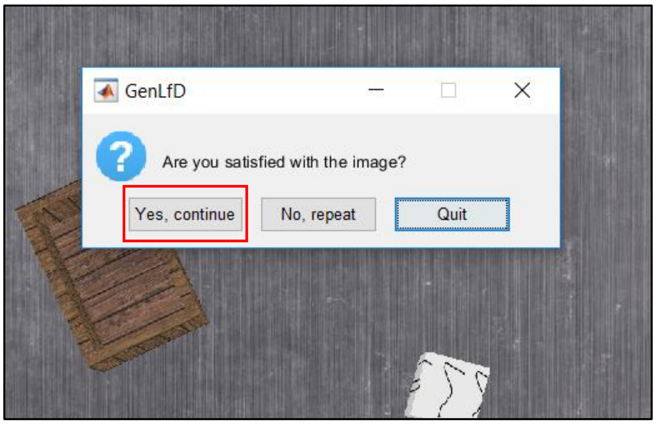
Once the cube and the box are moved and rotated to your choice, click *OK* on the GenLfD dialogue box that opens within MATLAB.



Step 4

The captured image will be shown as well as a GenLfD dialogue box asking you if you are satisfied with the image. Click *Yes*, continue if you are satisfied with the image.

However, if any of the objects are outside the camera's field of vision, move them again to your choice and then click *No, repeat*. If you would like to terminate the software, click *Quit*.

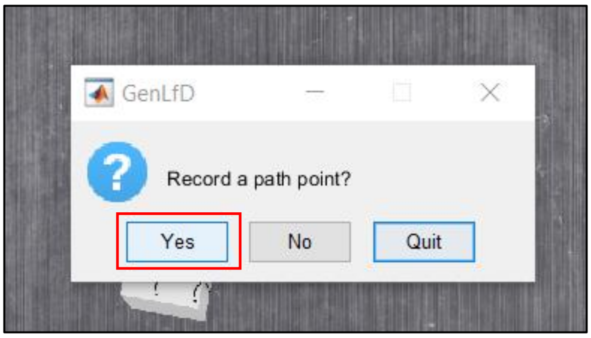


Step 5

If you clicked *Yes, continue*, now is time to record the demonstration path. In this task, each path is made from 4 path points:

1. Pre-grasp point, above the cube with open gripper
2. Grasp point, on the cube with closed gripper
3. Midway point, between the cube and box, above the table with closed gripper
4. Drop point, above the box with open gripper

In the next dialogue box, you will be asked if you want to record a path point. Click *Yes*, to begin recording.

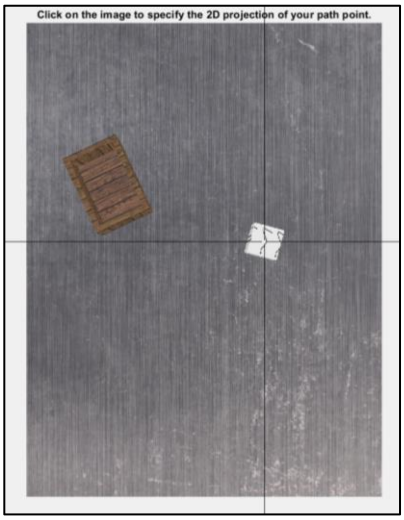


Step 6

The first path point in a pick-and-place task is typically above the object to be picked. In this case it is above the cube.

Each path point consists of 3 dimensions: *x, y* and *z*. Firstly, you record the *x* and *y* position by clicking on the cube in the image.

When clicking, observe how the robot end effector moves above the cube in CoppeliaSim.



A dialogue box will pop-up but **DO NOT** click it yet.

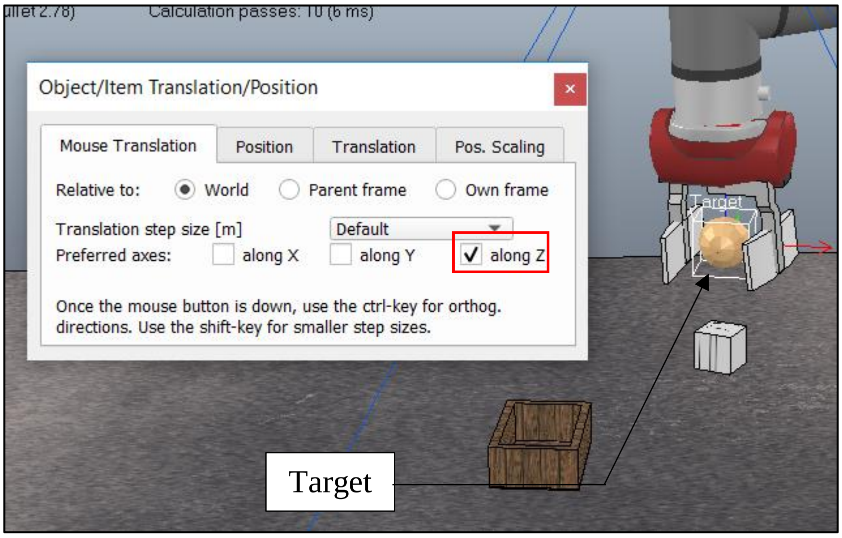
Step 7

In CoppeliaSim, you will find that the robot's gripper is now is above the *x-y* point that you chose in Step 6. You will see that there is a golden sphere within the robots grippers. This golden sphere will be referred to as the **Target.**

Now, you should adjust the z position of the path point, i.e. the height of the gripper above the object.

The **Target** is too high above the cube in the z direction.

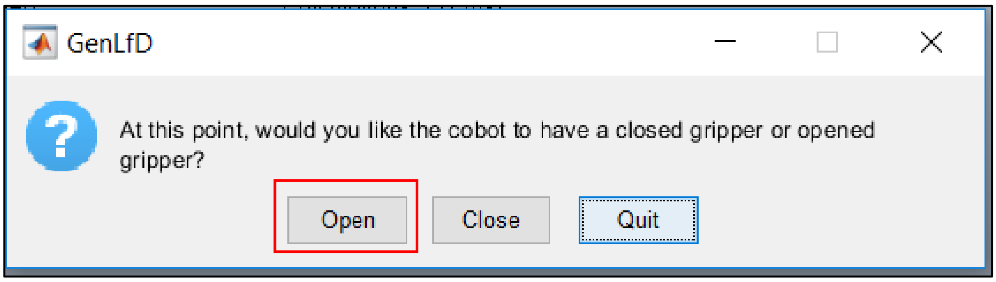
To bring the **Target** down to a more appropriate height, as in Step 3, open the shift tool then click on the **Target** and move it along the **z direction** down to a few centimetres above the white cube.



Return to the open dialogue box in MATLAB and now click *OK*.

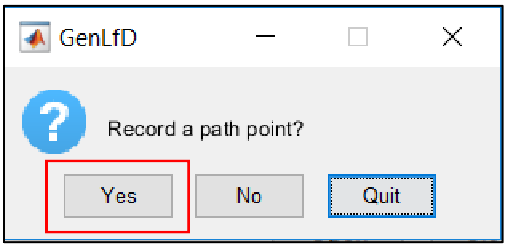
Step 8

After recording the coordinates of a path point, you are asked whether the gripper is to be open or closed at that point. For the 1st path point, click *Open*.

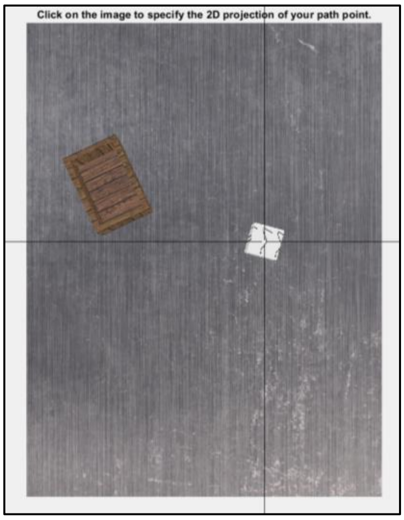


Step 9

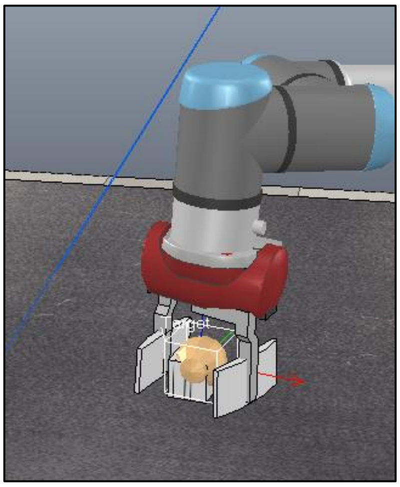
Next, you need to record the 2nd path point. Click *Yes* when asked to record a path point.



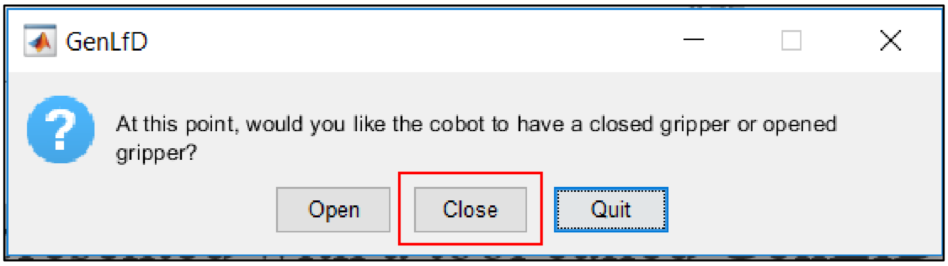
Secondly, click on the cube to record the *x-y* coordinates of the 2nd path point.



Then, adjust the height of the **Target** so that the cube is in between the gripper fingers.



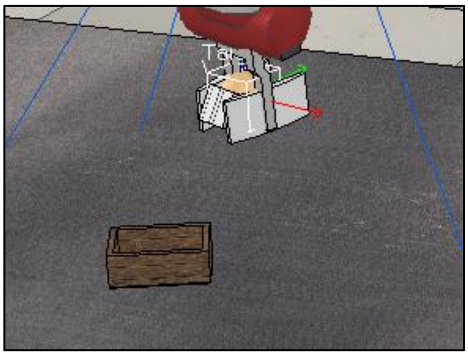
When asked if the gripper is closed or opened, click *Close*. You will see the gripper will close in the simulation.

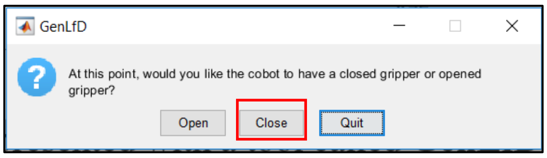


Step 10

Repeat the above step for the 3rd point; however, this time

- The *x-y* coordinates are between the cube and the box
- Raise the height a few centimetres above the table
- Keep gripper closed



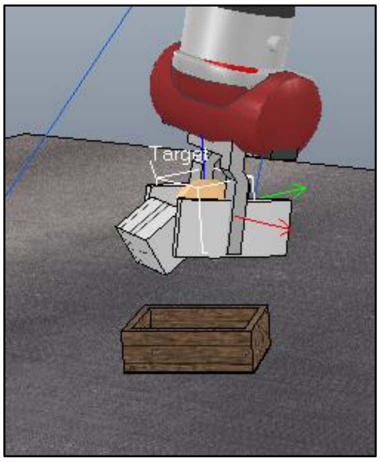


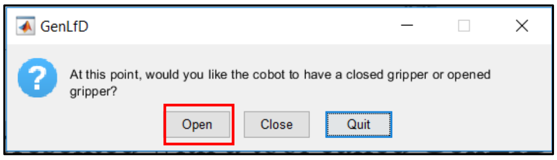
(Note: when the robot moves to the specified point, the cube might fall out of the gripper. This is just a simulation fault and will not affect the end learnt model. Therefore, continue the steps and ignore this fault. Refer to Potential Problems #1)

Step 11

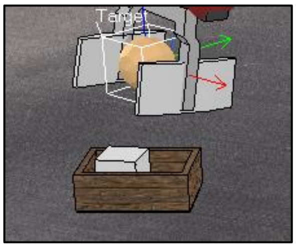
Repeat the above step for the 4th point which should represent the drop position.

- The *x*-*y* coordinates are on the box
- Adjust the height to be slightly above the box
- Set the gripper to open



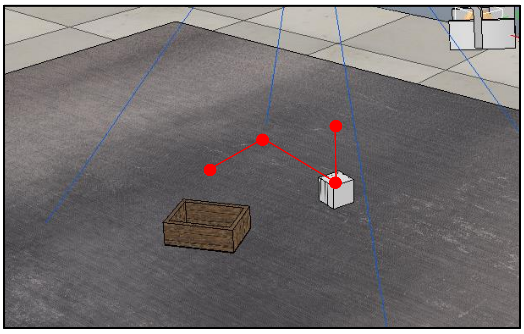


You will see that the object falls into the box and the task is completed.

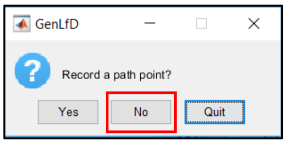


Step 12

You have thus recorded the necessary path points to accomplish the task.



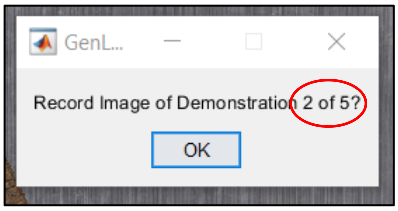
When prompted again to record a path point, click *No*, since the task is completed.



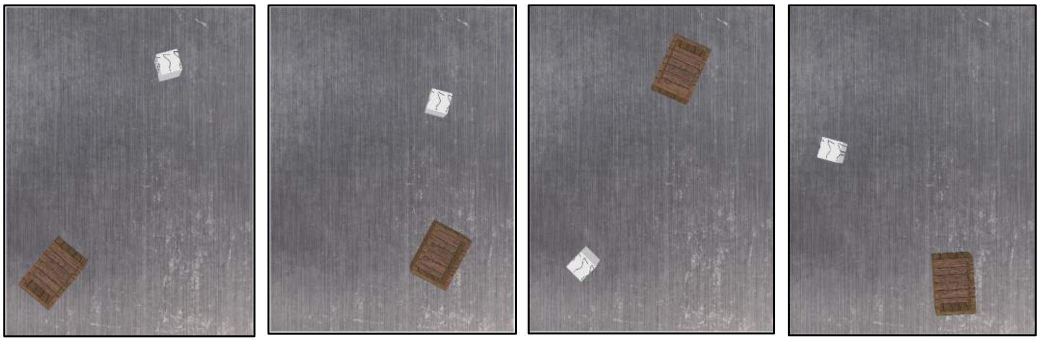
Step 13

For the robot to learn the task, you need to provide multiple demonstrations with varied positions of cube and box. That way, the robot will understand that the path points are dependent on the positions of the cube and the box.

Now you will repeat steps 3 to 12, 4 more times until you have recorded 5 demonstrations in total.

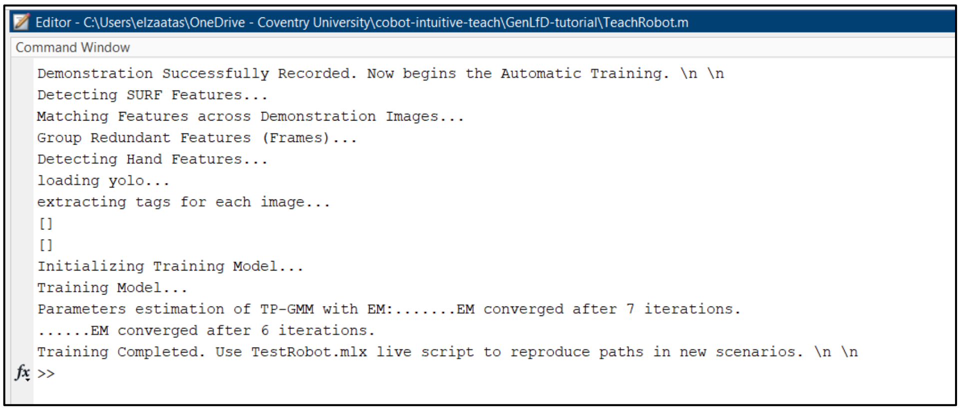


Make sure to vary the positions and orientations of the cube and the box well between demonstrations to create a good variety. Below are a few examples of how you can vary the positions of the cube and box.



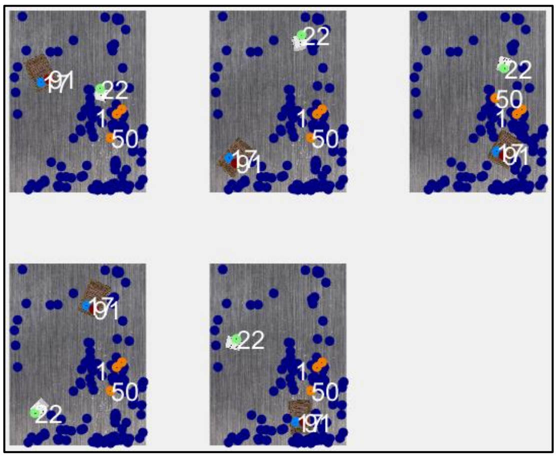
Step 14

Once all the demonstrations are recorded, the training process starts automatically so you only need to wait. At the end, you will see the result of the training process.

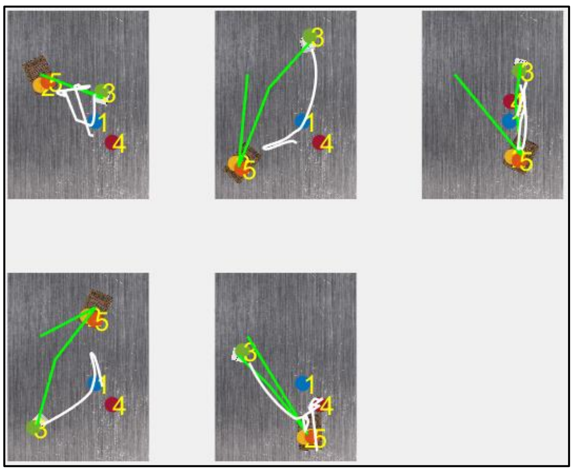


```
Editor - C:\Users\elzaatas\OneDrive - Coventry University\cobot-intuitive-teach\GenLfD-tutorial\TeachRobot.m
Command Window
Demonstration Successfully Recorded. Now begins the Automatic Training. \n \n
Detecting SURF Features...
Matching Features across Demonstration Images...
Group Redundant Features (Frames)...
Detecting Hand Features...
loading yolo...
extracting tags for each image...
[]
[]
Initializing Training Model...
Training Model...
Parameters estimation of TP-GMM with EM:.......EM converged after 7 iterations.
......EM converged after 6 iterations.
Training Completed. Use TestRobot.mlx live script to reproduce paths in new scenarios. \n \n
fx >>
```

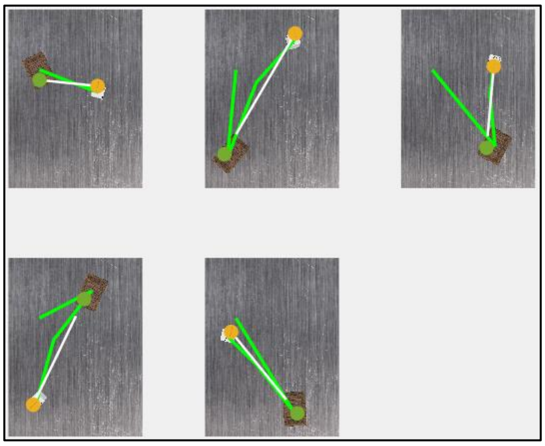
Firstly, the task parameters will be detected. They are the different dots in the image below. It is important that at least one dot belongs to both the cube and the box.



Secondly, the model will be trained to obtain probabilistic distributions of the paths with respect to the task parameters. Here we notice that the generated white path, is not satisfactory since it is very different than the ground truth path in green. The next step will solve this problem.



Thirdly, a reinforcement learning algorithm will identify optimal task parameters that generate a better path. You can see that the new path in white is much better than the originally generated path in red.

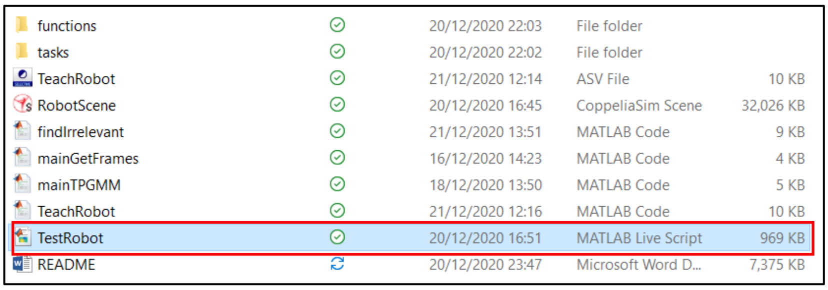


(Note: In the training process, two errors might occur. Refer to Potential Problems #2 and #3.)

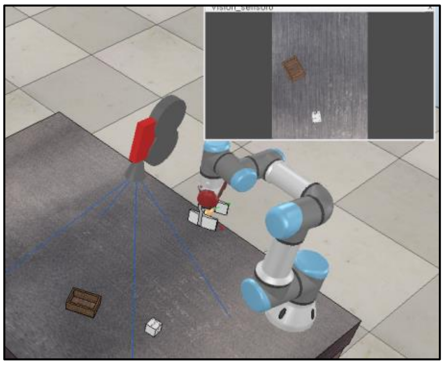
Step 14

Once the training is completed, you can test your learnt model.

In the folder, click on TestRobot.mlx.



In CoppeliaSim, vary the positions of cube and box such that they are in new positions the cobot has never seen.

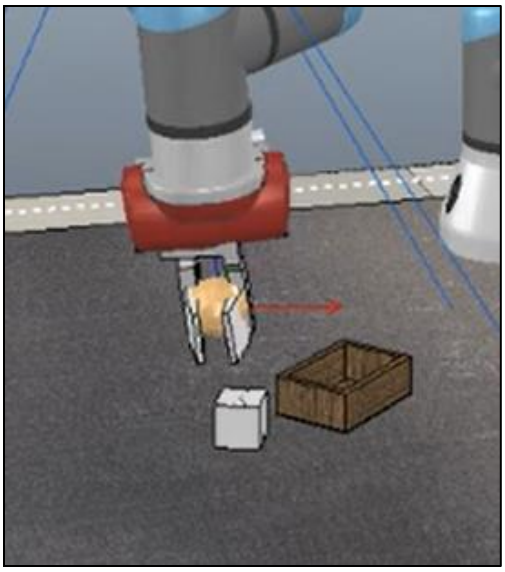


Run the TestRobot MATLAB live script. You will obtain the reproduced path in this new scenario.

(Note: If you receive an error saying no subtask was matched, that means the algorithm failed to detect the

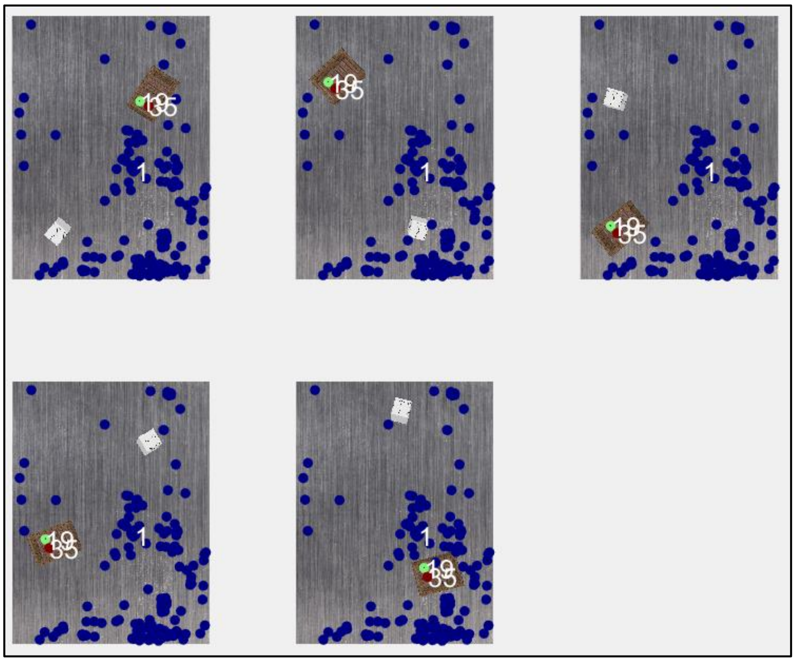
Potential problems

1. Failed Demonstration



When recording the demonstration, the cube might fall out of the gripper while moving. This is a simulation dynamics error and won't actually affect the algorithm performance.

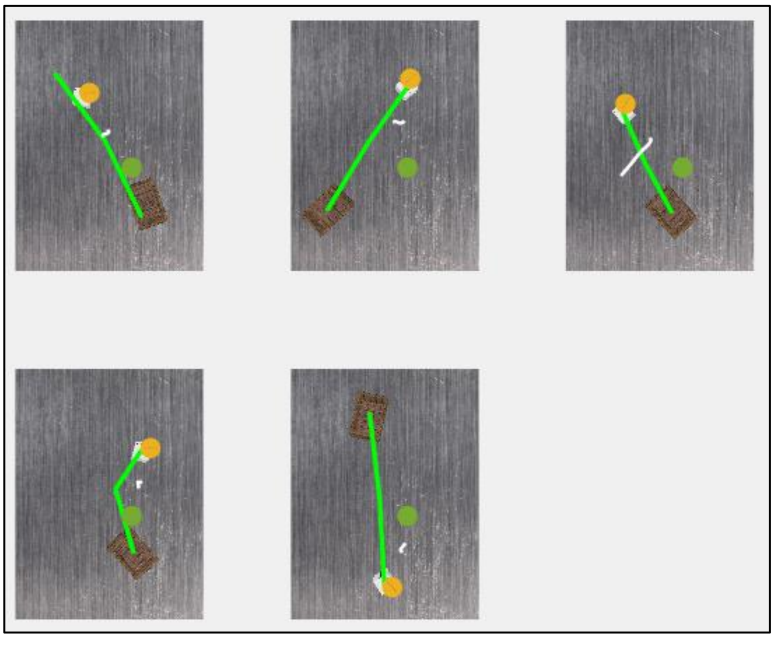
2. Missed objects



When looking at the matched frames of references (the dots), we can notice that none of them belong to the cube. That means that the cube was not detected by the algorithm. Therefore, the reproduced paths will be ineffective.

This could be because the object doesn't have prominent features, or it has a reflective surface, or there are shadows. Try to overcome the above problems and rerun the recording algorithm. If problem persists, seek help from the programmer.

3. Incorrect convergence



The reproduced path in white is unsatisfactory and not meaningful, that is because the green frame belongs to the table, not the box. When this happens, run the function of findirrelevant.m until satisfactory results are obtained.

4. No Features Detected

In TestRobot live script, the algorithm attempts to detect the task parameters in the new image. In rare occasions, they might fail to detect and in such a case, the path cannot be reproduced.

SURVEY

Please complete this survey thoroughly to evaluate the GenLfD software, its user-friendliness, performance and clarity.

Gender:	☐ Male ☐ Female
Age:	____
Level of Education:	☐ Doctorate ☐ Masters ☐ Bachelors ☐ College ☐ Secondary school ☐ Professional Certification
Programming experience:	☐ I have no computer knowledge ☐ No programming knowledge ☐ I have done visual programming (LabVIEW…) ☐ I have experience with programming ☐ I am a professional programmer
What the programming successful?	☐ Yes

	☐ No, error 1 ☐ No, error 2 ☐ No, error 3 ☐ Other error:
How many times did you run the program?	______ times
Insert time taken to program the task.	______ minutes
Which one of these tasks do you think the GenLfD is capable of programming?	☐ picking up tool and handing it to human ☐ tight tolerance peg-in-hole ☐ going from A to B ☐ going from A to B with obstacles ☐ going from A to B with moving obstacles ☐ sorting objects into two different containers ☐ marking objects

On a scale of **1 to 5**, 1 being **Not At All** and 5 being **Very Much**, please answer the following questions:

Questions	**1**	**2**	**3**	**4**	**5**
How intuitive is GenLfD?					
How confident are you that you are capable to use GenLfD to program robots for other tasks?					

How confident are you that GenLfD can be used to program other tasks?					
Did you get bored while using GenLfD?					
Did you understand the steps occurring when the code was running autonomously?					
If you managed a factory, how likely are you to introduce GenLfD for the factory users?					

Any further comments about GenLfD?

Thank you for your participation.