Experiment-1

Linux command

Experiment-2

Installation of ROS

And

To download the V-REP software, go to:

[http://coppeliarobotics.com/downloads](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbjgxZVcwZVM1bUJyLUVBN0RvVXJmeHIyVThQUXxBQ3Jtc0ttWTd5QkRrT3NPWGVTaUdqcFoxUl9VTkRLQVhFYndZMEtwQ21LcTJtQWNoTFl6S0VnQm5MZ1pUOHpscDhtNC1IeWx3RTVMU3B1cDI3QWZ5ZWZ2V1E4VExLdXBHQmV6S3FiUXRWLWhGRnJHNFB0N1hpTQ&q=http%3A%2F%2Fcoppeliarobotics.com%2Fdownloads)

[CoppeliaSim](http://coppeliarobotics.com/) is robot simulators: you won't find a simulator with more functions, features, or more elaborate APIs:

 Cross platform (Windows, MacOSX, Linux)

 [6 programming approaches](https://www.coppeliarobotics.com/helpFiles/en/writingCode.htm#sixMethods) ([embedded scripts](https://www.coppeliarobotics.com/helpFiles/en/scripts.htm), [plugins](https://www.coppeliarobotics.com/helpFiles/en/plugins.htm), [add-ons](https://www.coppeliarobotics.com/helpFiles/en/addOns.htm), [ROS nodes](https://www.coppeliarobotics.com/helpFiles/en/rosInterfaces.htm), [remote API](https://www.coppeliarobotics.com/helpFiles/en/remoteApiOverview.htm) clients, or [BlueZero nodes](https://www.coppeliarobotics.com/helpFiles/en/blueZeroPlugin.htm))

 6 programming languages ([C/C++](https://www.coppeliarobotics.com/helpFiles/en/apiFunctionListAlphabetical.htm), [Python](https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsPython.htm), [Java](https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsJava.htm), [Lua](https://www.coppeliarobotics.com/helpFiles/en/apiFunctionListAlphabetical.htm), [Matlab](https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsMatlab.htm), and [Octave](https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsOctave.htm))

 more than [400 different API functions](https://www.coppeliarobotics.com/helpFiles/en/apiFunctionListAlphabetical.htm)

 [4 physics engines](https://www.coppeliarobotics.com/helpFiles/en/dynamicsModule.htm) ([ODE](http://www.ode.org/), [Bullet](http://www.bulletphysics.org/), [Vortex](https://www.cm-labs.com/vortex-studio/), [Newton](http://www.newtondynamics.com/))

 Integrated ray-tracer ([POV-Ray](http://www.povray.org/))

 [Full kinematics solver](https://www.coppeliarobotics.com/helpFiles/en/inverseKinematicsModule.htm) (IK and FK for ANY mechanisms)

 [Mesh, OC tree and point cloud interference detection](https://www.coppeliarobotics.com/helpFiles/en/collisionDetection.htm)

 [Mesh, OC tree and point cloud minimum distance calculation](https://www.coppeliarobotics.com/helpFiles/en/distanceCalculation.htm)

 [Path / motion planning](https://www.coppeliarobotics.com/helpFiles/en/pathAndMotionPlanningModules.htm) (holonomic in 2-6 dimensions, non-holonomic for car-like vehicles, and motion planning for kinematic chains)

 [Vision sensors](https://www.coppeliarobotics.com/helpFiles/en/visionSensors.htm) with built-in image processing (fully extendable)

 Realistic [proximity sensors](https://www.coppeliarobotics.com/helpFiles/en/proximitySensors.htm) (minimum distance calculation within a detection volume)

 Built-in [custom user interfaces](https://www.coppeliarobotics.com/helpFiles/en/customUIPlugin.htm)

 Fully integrated [Reflexxes Motion Library type 2](http://www.reflexxes.com/) and [RRS-1 interface specifications](http://www.realistic-robot-simulation.org/)

 [Data recording and visualisation](https://www.coppeliarobotics.com/helpFiles/en/graphs.htm) (time graphs, X/Y graph or 3D curves)

 Integrated [shape edit modes](https://www.coppeliarobotics.com/helpFiles/en/shapeEditModes.htm)

 Dynamic particles for water-/air-jet simulation

 [Model browser](https://www.coppeliarobotics.com/helpFiles/en/userInterface.htm" \l "ModelBrowser) with drag-and-drop functionality (also during simulation)

 Multi-level undo/redo, [movie recorder](https://www.coppeliarobotics.com/helpFiles/en/aviRecorder.htm), simulation of paint dispensing, [exhaustive documentation](https://www.coppeliarobotics.com/helpFiles/index.html), etc.

robot, robotics, simulator, simulation, kinematics, dynamics, path planning, minimum distance calculation, collision detection, vision sensor, image processing, proximity sensor, paint dispensing.

**a console window**: under Windows, when the CoppeliaSim application starts, a console window is created but directly hidden again. This default behavior of hiding the console window can be altered in the [user settings dialog](https://www.coppeliarobotics.com/helpFiles/en/settings.htm). Under Linux, CoppeliaSim needs to be started from the console, which stays visible throughout the whole CoppeliaSim session.

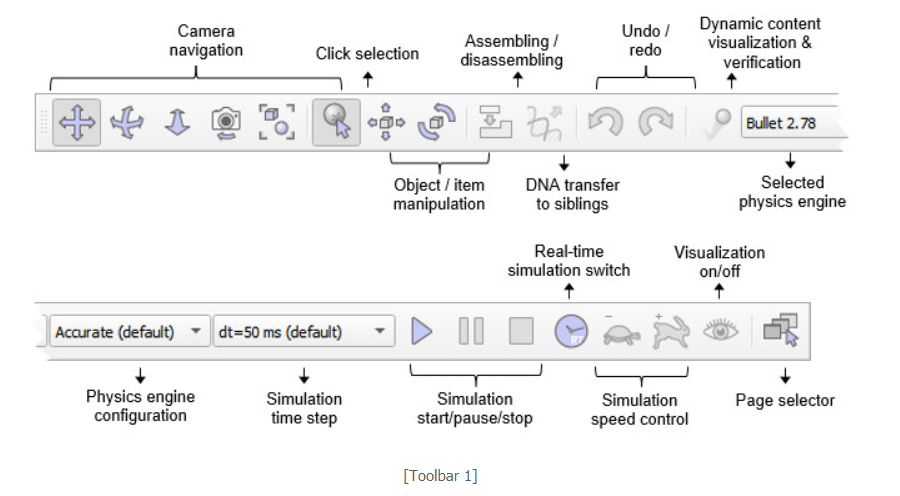
The application bar, as well as any surface within the application window, can also be used to drag-and-drop CoppeliaSim related files into the scene.

 Supported files include "\*.ttt"-files (CoppeliaSim scene files) and "\*.ttm"-files (CoppeliaSim model files).

**menu bar**: the menu bar allows accessing almost all functionalities of the simulator. Most functions in the menu bar can also alternatively be accessed through a popup menu, a double-click on an icon in the scene hierarchy view, or through a click of a toolbar button.

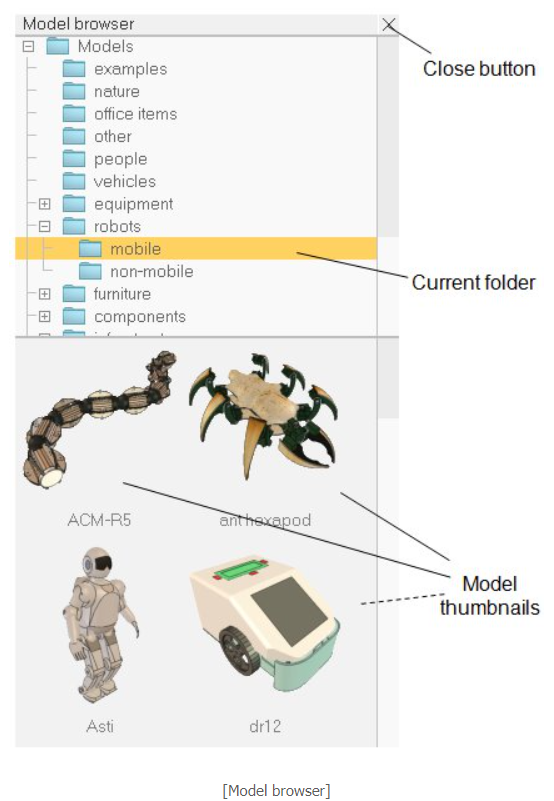
**toolbars**: the toolbars present functions that are often accessed (e.g. changing the navigation mode, selecting another page, etc.). Some functions in toolbar 1, and all functions in toolbar 2 can also be accessed through the menu bar or popup menu.

Following figure explains each toolbar button's function:

****

****

**model browser**: the model browser is visible by default, but can be toggled with its corresponding toolbar button. It displays in its upper part a CoppeliaSim model folder structure, and in its lower part, thumbnails of [models](https://www.coppeliarobotics.com/helpFiles/en/models.htm) contained in the selected folder.

****

Experiment-3

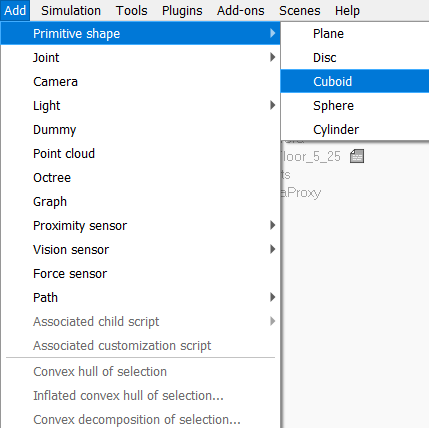
Turtle sim simulation using ROS

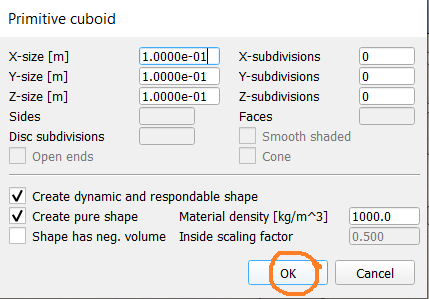
**Experiment-4**

Aim Of The Experiment:

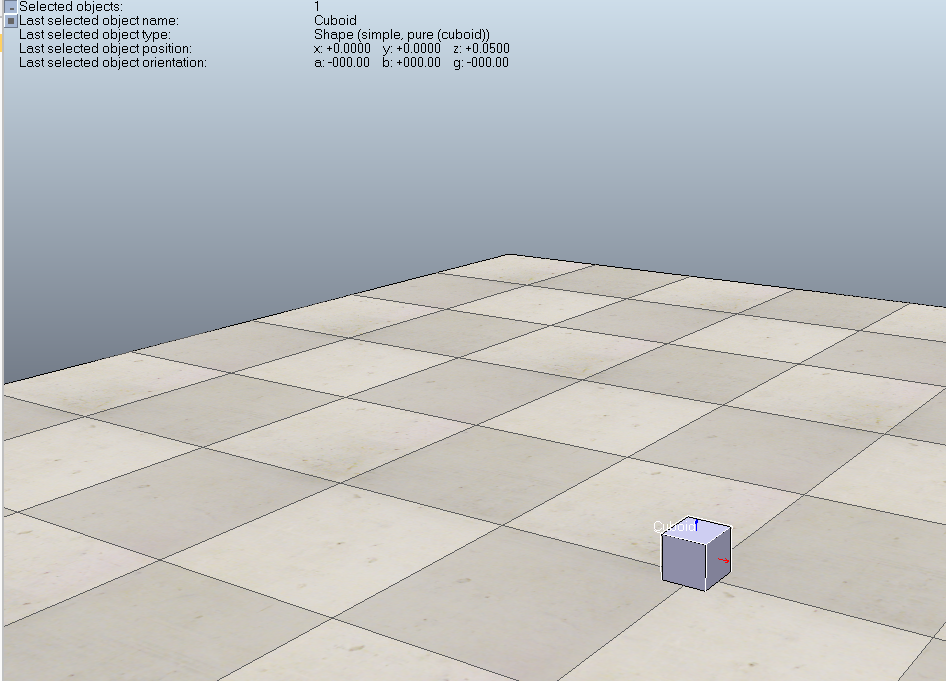
Introduction to CoppeliaSim

Step-1:

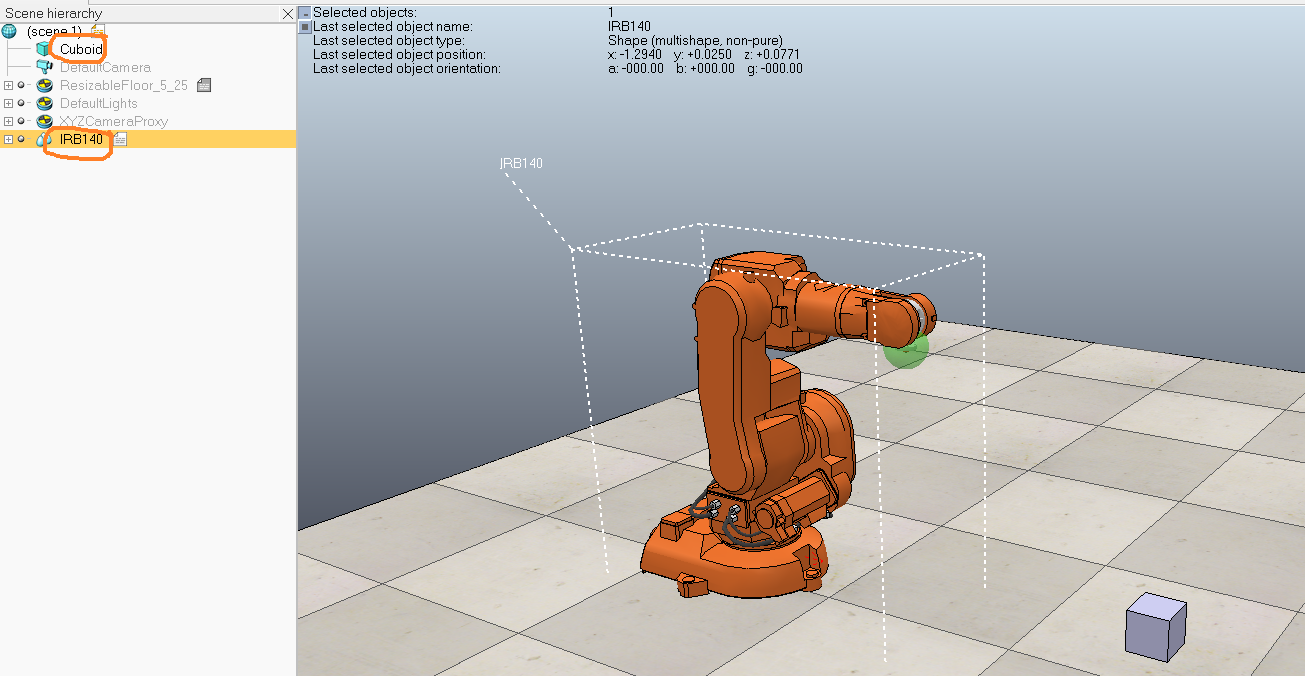




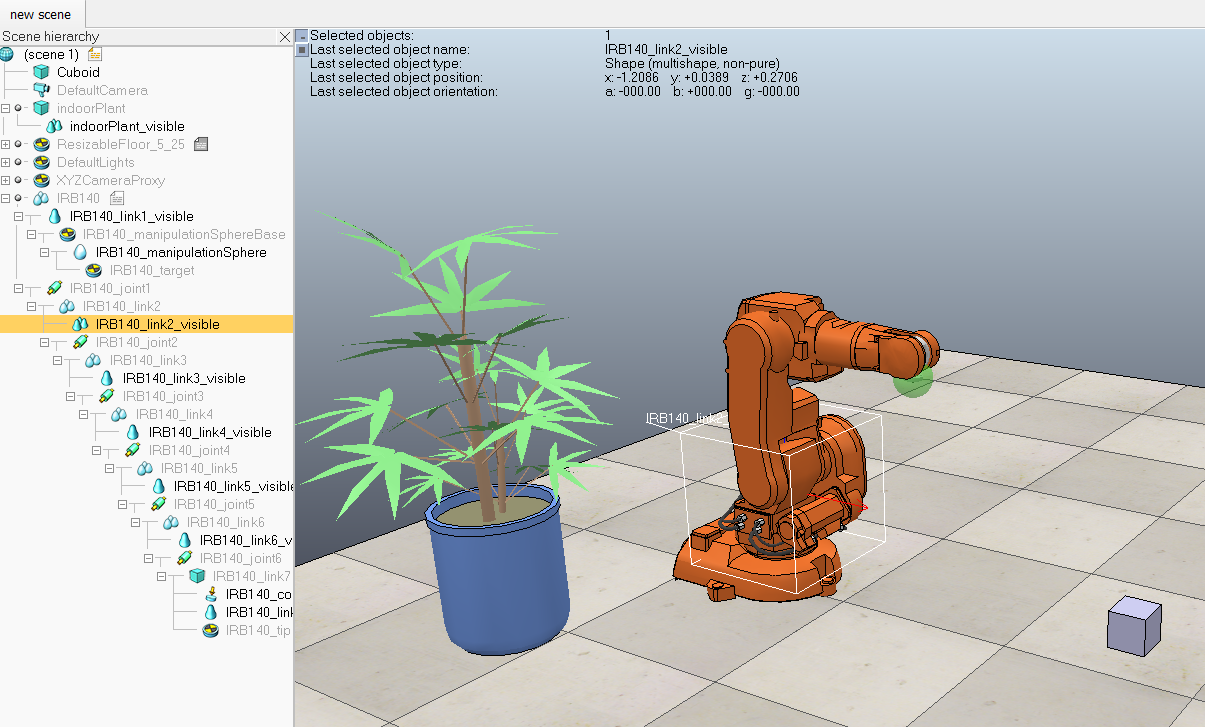
Defoutl Cuboid

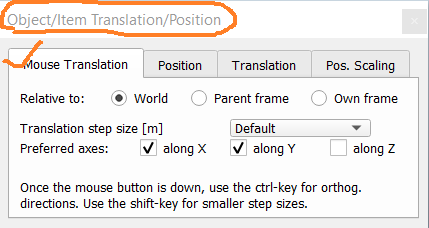


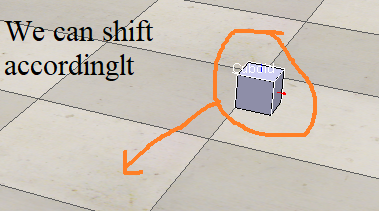
Step-2

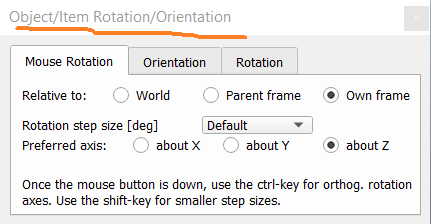


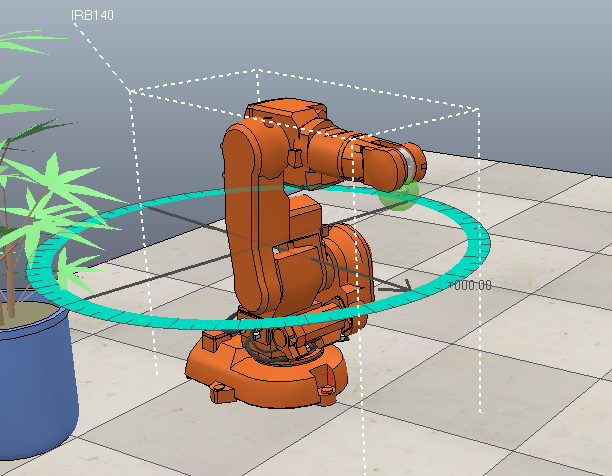
Step 3:





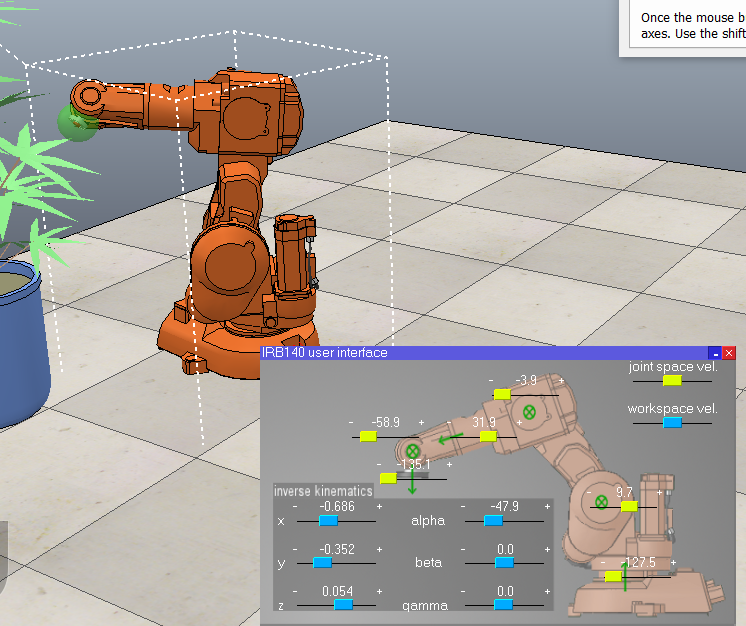
 



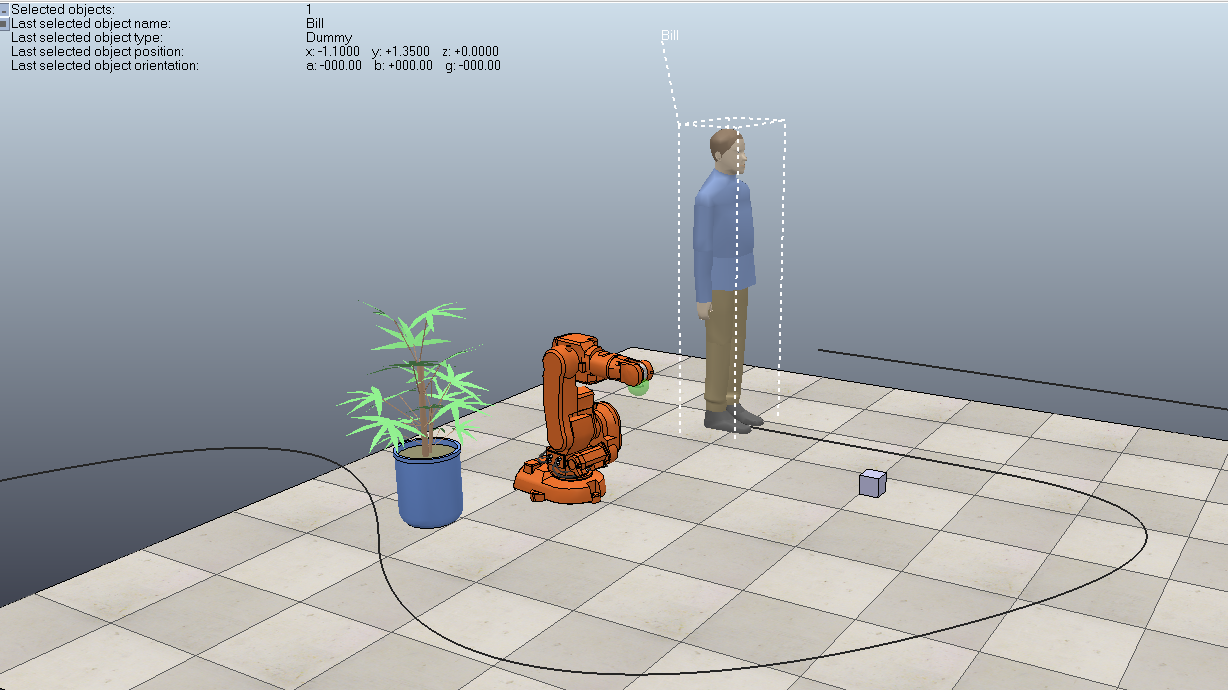
Step-4

Simulation



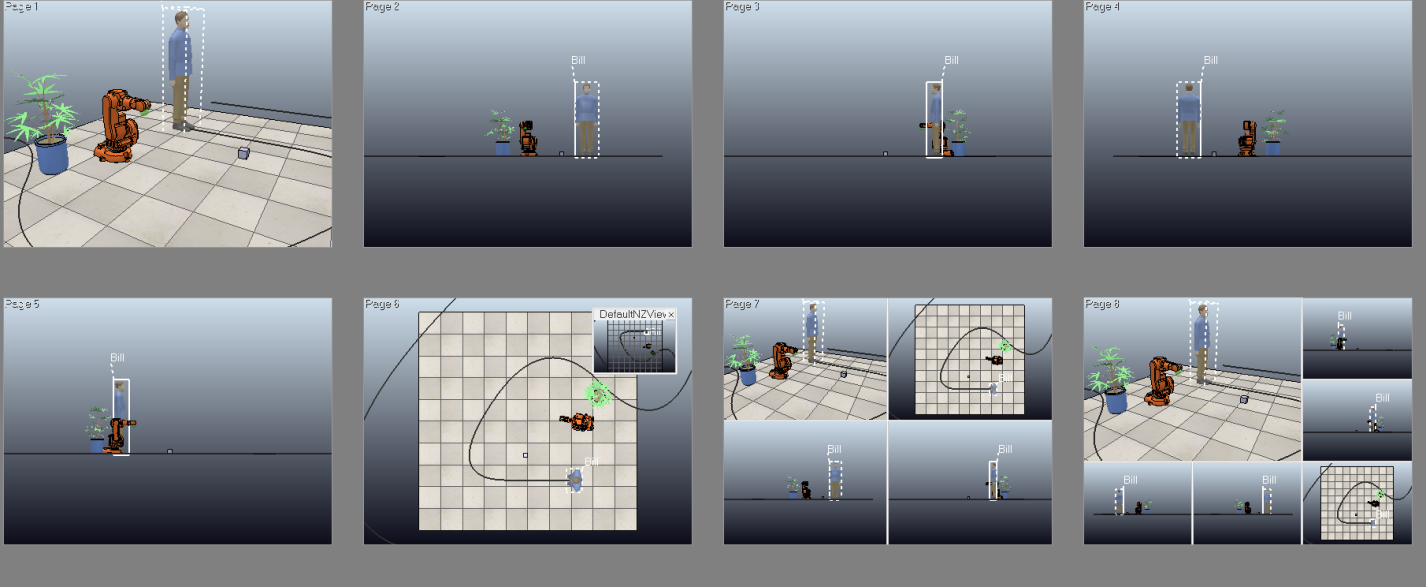






Step 5:

Finally we can view the scene



**Conclusion:**In This experiment we learn how to use CoppeliaSim(V Reb) and some of the capabilities.

Experiment-5

Aim of the experiment:

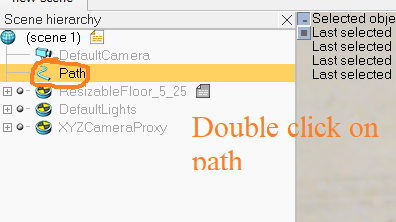
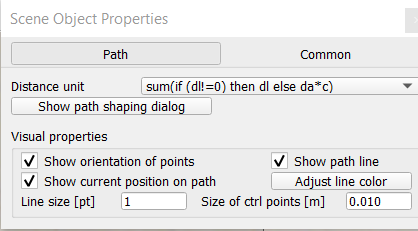
# Creating paths on the floor using CoppeliaSim(V-Reb)

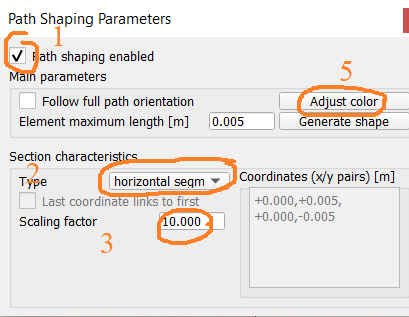
# Step-1:

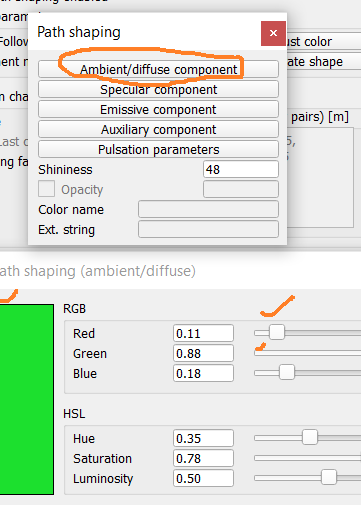
# 

# 

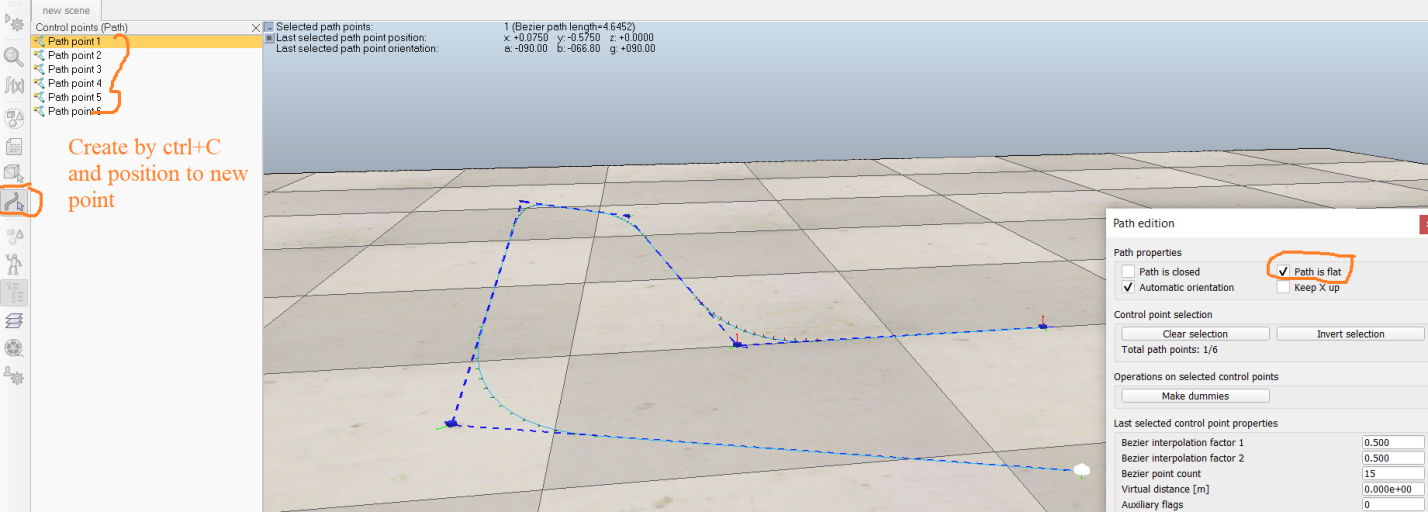
Step-2

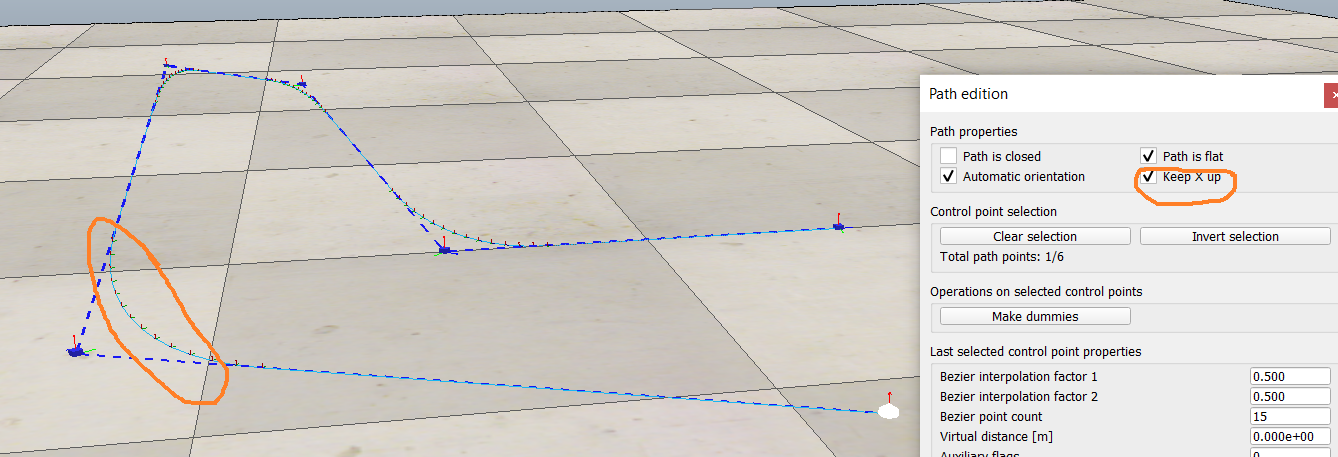
 

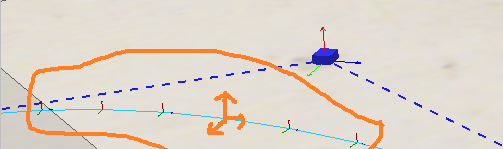




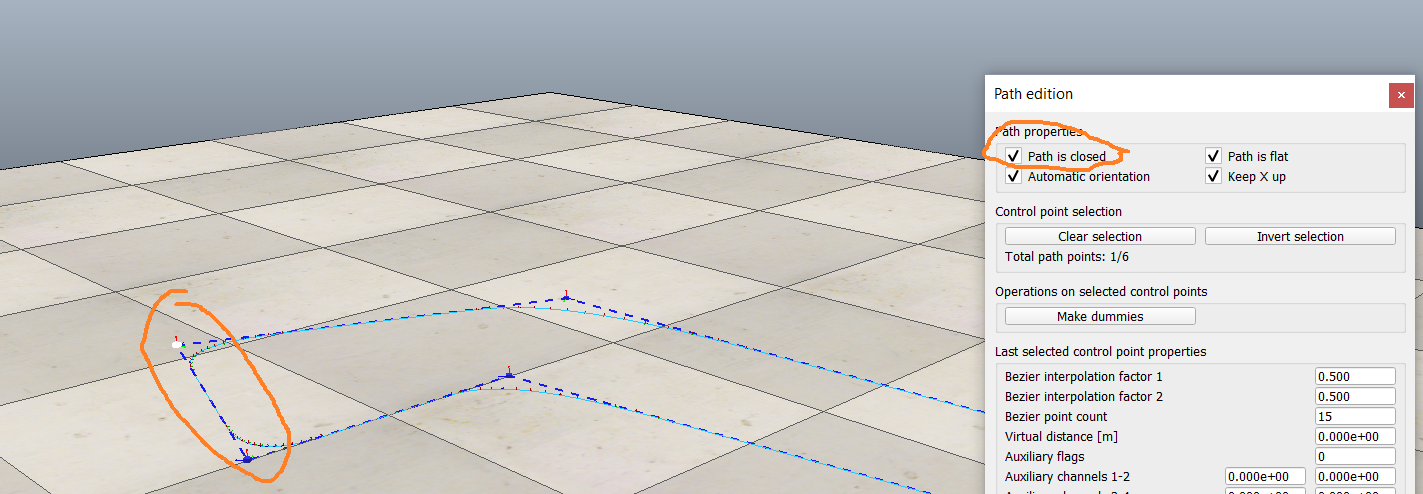
Step-3



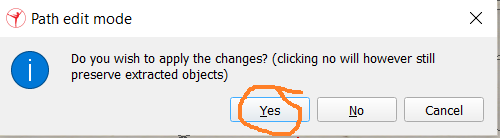


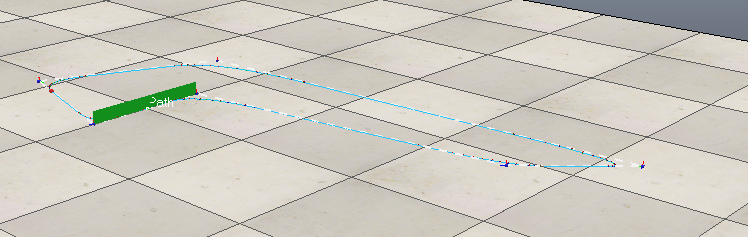


Step-4



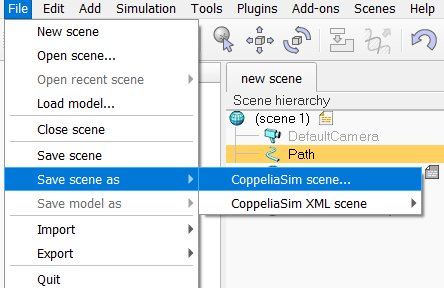
Step-5

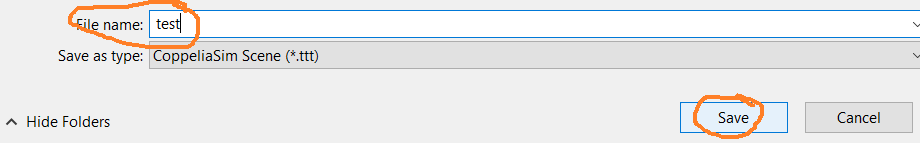




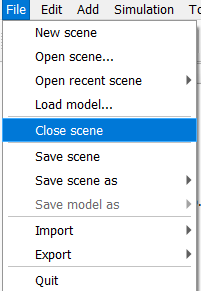
Path is not properly visible

Step-6

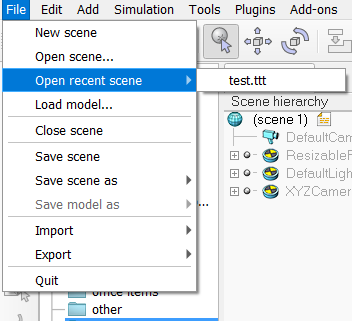




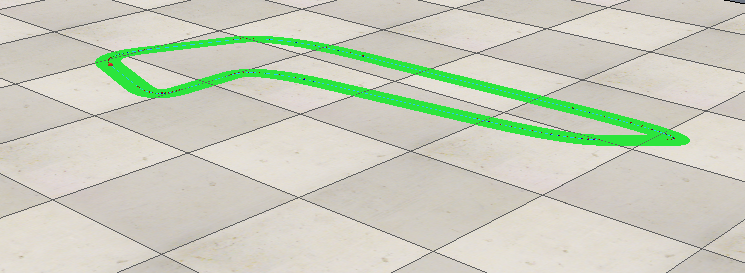
Step-7



Step-8



Step-9



Conclusion:In this experiment we learn how to create path in CoppliarSim(V-Reb)

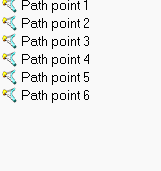
Experiment-6

Aim of the experiment:To design and simulate line follower robot using CoppeliaSim(Vrep)

Component Required:

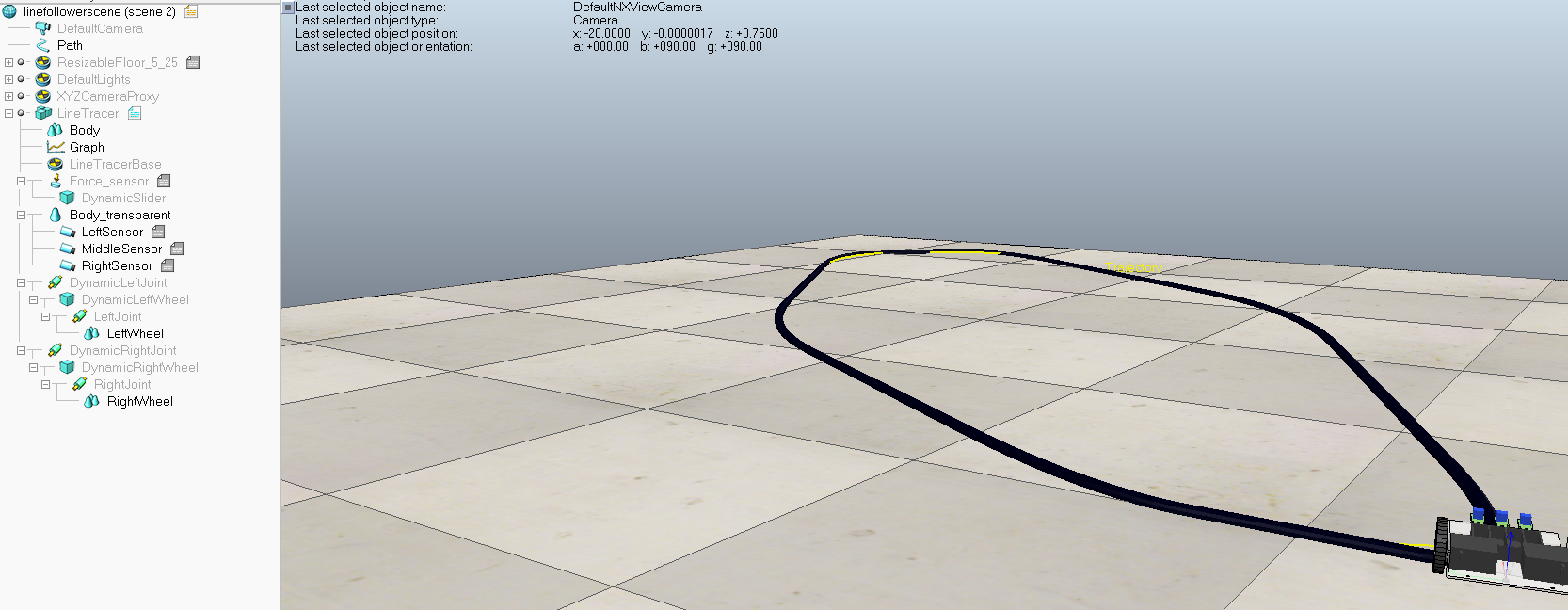
Step-1:

Create path by add path



Step-2:

From model browser select linefollower.ttt plugin and drag in the path scene and adjust the robot in proper position so that robot will run on the path .



Conclusion