A Custom Robotic ARM in CoppeliaSim

Sudip Chakraborty[1] & P. S. Aithal[2]
[1]Post-Doctoral Researcher, College of Computer science and Information science, Srinivas University, Mangalore-575 001, India
OrcidID: 0000-0002-1088-663X; E-mail: sudip.embedded@gmail.com
[2]ViceChancellor, Srinivas University, Mangalore, India
OrcidID: 0000-0002-4691-8736; E-Mail: psaithal@gmail.com

Subject Area: Engineering.
Type of the Paper: Simulation based Research.
Type of Review: Peer Reviewed as per |C|O|P|E| guidance.
Indexed In: OpenAIRE.
DOI: http://doi.org/10.5281/zenodo.4700297
Google Scholar Citation: **IJAEML**

How to Cite this Paper:
Chakraborty, Sudip, & Aithal, P. S., (2021). A Custom Robotic ARM in CoppeliaSim. *International Journal of Applied Engineering and Management Letters (IJAEML)*, *5*(1), 38-50. DOI: http://doi.org/10.5281/zenodo.4700297.

International Journal of Applied Engineering and Management Letters (IJAEML)
A Refereed International Journal of Srinivas University, India.

Crossref DOI : https://doi.org/10.47992/IJAEML.2581.7000.0091

A Custom Robotic ARM in CoppeliaSim

Sudip Chakraborty[1] & P. S. Aithal[2]
[1]Post-Doctoral Researcher, College of Computer science and Information science, Srinivas University, Mangalore-575 001, India
OrcidID: 0000-0002-1088-663X; E-mail: sudip.embedded@gmail.com
[2]ViceChancellor, Srinivas University, Mangalore, India
OrcidID: 0000-0002-4691-8736; E-Mail: psaithal@gmail.com

ABSTRACT

Purpose: For robotics research, we require the robot to test our functions, Logics, algorithms, tasks, etc. Generally, we do not experiment with the practical robot. The primary issue is Practical robots are costly. The individual researcher usually cannot afford it. The second one is, the test with the real robot is risky and can damage property, human life, and itself due to bugs in the program or abnormal activity. So, it is best practice to experiment in Simulator first. When the algorithm is finalized, it can be implemented into a real robot. A researcher who starts the Robotics research, the learning curve is too long to develop a workable robot in Simulator. This paper demonstrates how we can easily create a 7 Degree of Freedom (DOF) custom robot for our research purpose. We will use the CoppeliaSim robot simulator for this purpose. It is free, opensource, and entirely GUI-based. We can create a robot without writing any code using this software.
Design/Methodology/Approach: Here we describe to develop a custom robot. At first, we created a DH parameter for our robot. Then following the step-by-step procedure, the robot is created. After creating, we can attach our code on any object using LUA script language. To control the robot from external world, we can connect through TCP/IP socket communication. Establishing the communication, our robot will move depending on processed algorithm.
Findings/Result: The robotic arm researcher needs robotics arm to test their forward kinematics, Inverse kinematics, statics, dynamics etc. code. Here we design our custom robots for research purpose.
Originality/Value: Using CoppeliaSim, we can design custom robot for our research.
Paper Type: Simulation based Research
Keywords: 7DF ARM, Custom Robot Creation, Robotic ARM, Robotic ARM Simulation

1. INTRODUCTION :

With technological development, computer processing power within a small footprint is becoming popular. The Single-board computer (SBC) for robotics applications is reasonably available now. NVIDIA Jetson Nano is one kind of board. Raspberry Pi, Arduino MEGA is also using on more miniature computing-powered robots. Except for heavy industrial use, most of the robotics application is battery powered mobile robot. When we design this kind of robot, our focus is to achieve maximum efficiency, running long in one charge. Various types of robots are used in multiple fields. From production to household, we are surrounded by the robot. We need in-depth research for robots' flawless operation. It is required to account the various scenario happened in real-world activities. Before going to real-world action, the complete or partly firmware must be simulated safely or in the Simulator. It can be detected significant flaws in the algorithm before damage to anything. So various robot simulators are available from different vendors for robotics research. Some are RoboDK, Gazebo, Webots, Microsoft Robotics studio in the same row. From the list. RoboDK is feature-rich and up to the mark but costly for new robot researchers. The Webots is a free and good simulator for a mobile robot. Microsoft robotics studio was the best fit for our purpose, but the company stops its further development, and no support is available for it. Gazebo simulator is free, but it is Linux-based and requires a lot of computation power. After Lots of brainstorming, we found a simulator that can work best for us. The CoppeliaSim Robot simulator can be used for our robotics research. It is active, and support is good enough. Lots of documents are available around the web. Many researchers work and

publishing their papers based on the CoppeliaSim simulator. It is completely free and opensource.
In this paper, we will see how to build a custom seven-degree freedom robot using CoppeliaSim. Without using any code, we can design a robotics arm with little effort. Following step by step carefully, our custom parameter-based robotic arm is made quickly.

2. RELATED WORKS :

Tom P. Huck et al., in their paper "Simulation-based Testing for Early Safety-Validation of Robot Systems" [1], demonstrate a proof of concept (POC) on human safety from robotics movement. Their approach relies on the assumption that the robot system's behaviour is deterministic forgiven human behaviour. This paper to test the safety has taken using prebuild robot. For custom robot design and their parameter variation for protection is still relatively unexplored. M. Kortmann et al. In their research [2], aims to develop a modular robotic arm for multi-purpose and multi-mission use that can be individually configured for the mission at hand module is based on the iBOSS (intelligent Building Blocks for On-orbit Satellite Servicing and Assembly) concept. It would be better if some practical procedure would explain. Javier Pinzon-Arenas et al. presents the development of a virtual environment [3] for testing industrial-type robotic applications, using artificial intelligence techniques through convolutional neural networks. They continue their research on prebuild robot model UR3. We did not find enough integration procedures for interested researchers to test their experiment. Igor Shardyko et al. demonstrate a better approach for applying series elasticity in modern robots focusing on robotic arms [4]. They implement a decentralized control scheme with inverse dynamics-based in the outer loop and torque controllers in joints. Instead of custom robot design, they use Franka Emika Panda prebuild robot for their simulation purpose. It was highly appreciated to carried on research using a custom design robot. To demonstrate the Digital twin for Applying a 6 DoF Robotic Arm and Digital Twin to Automate Fan-Blade Reconditioning for the Aerospace domain [5], John Oyekan et al. use Coppeliasim simulator. They use prebuild robot model from 6DoF ABB. In their paper, CoppeliaSim integration is not noticeable information they provide. Using the Gazebo simulator, an excellent tutorial [6] demonstration we got from Jiawei Hou et al., their demonstration on a mobile robot, their practice approach is applicable. It is basically for mobile manipulation; it would be better to touch or give Robotics arm information. Jan Schneider et al. propose a way [7] to use a combination of reinforcement learning and planning for the construction industry. They introduce robotic arm UR10 for their work. We did not get much information about the simulation procedure. Robotics and A.I. is a better controlling power towards achieving our success in any forms of automation. Tamir Blum and Kazuya Yoshida developed a system [8] where the robot can move through Rough Terrain using Reinforcement Learning.
Human-Robot Collaboration (HRC) is rapidly replacing the traditional application of robotics in the manufacturing industry. Mehrnoosh Askarpour et al. have built a framework to formally model HRC systems and verify human operator's physical safety against ISO 10218-2 [10] standard. They also demonstrate with a 3D simulator potential hazardous situation to the safety engineers in a more transparent way [9]. Omar Gamal et al., in their paper, propose a multimodal deep learning network for autonomous navigation of mobile robots in a static indoor environment. Their trained models showed that the network could control mobile robot motion and safely navigate the target goal. They established a bidirectional communication channel between the simulation and Python environments using CoppeliaSim remote API [10]. Md Nizamuddin Ahmed (M. E), K.Veladri developed 7 degrees of freedom redundant manipulator. In their paper, they import the robot's CAD model, which is produced using external software. It would be better the CAD/ Designed was created in the V-rep environment. [11]. S Samuel Abhishek and K.Veladri developed an Automated guided vehicle used in manufacturing shop floors, assisting older adults at home. It can navigate through the environment where the obstacles are static or dynamic. In this paper, the complete experiment was discussed easily so the new researcher can quickly test their research. It would be more appreciated to developed physical parts like wheels; in the robot simulator itself. Sometimes it is out of scope to use third-party software to developed model [12].

3. OBJECTIVES :

a) Through this paper, we will introduce the free and open-source CoppeliaSim Robot Simulator
b) The articulated robotic arm can be made quickly.
c) It minimizes the new robot researcher learning curve.
d) Straight forward experiment procedure for the experiment replication.

4. APPROACH AND METHODOLOGY :

We will develop our custom robot in the CoppeliaSim robot simulator. Before that, we need to familiar with the IDE (Integrated Development Environment). When we open CoppeliaSim, the below screen appears.

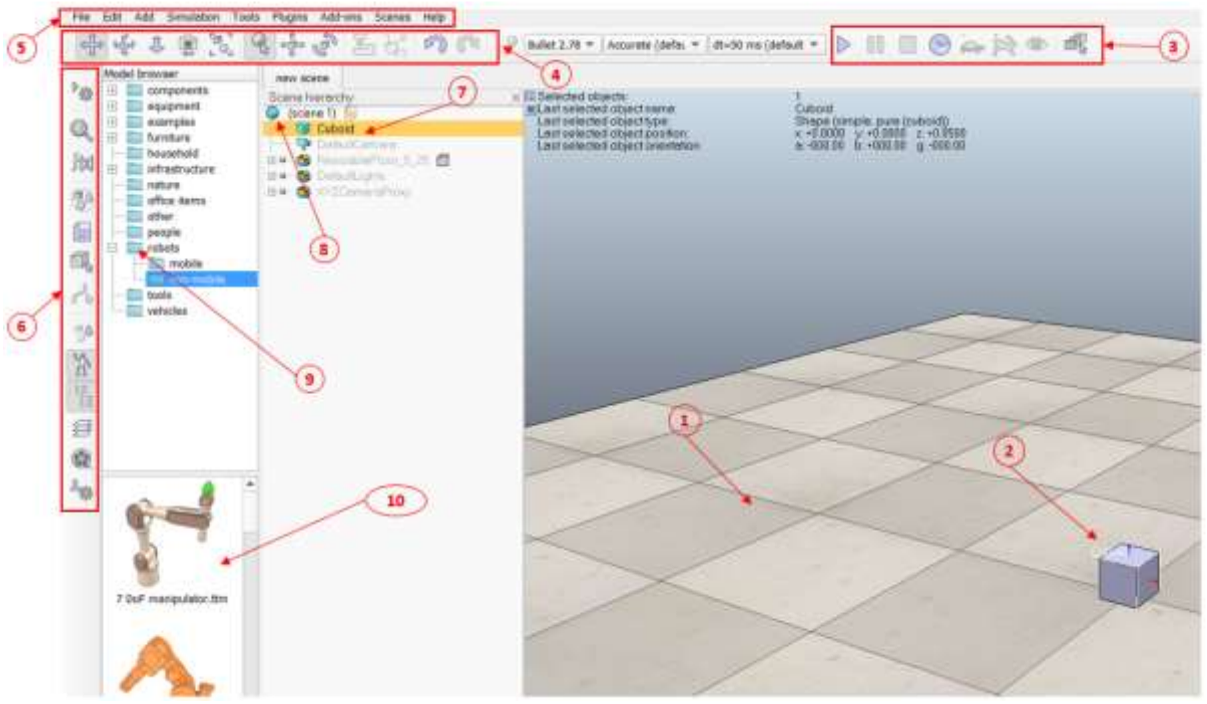

Fig. 1 : CoppeliaSim robot simulator with IDE

(1) This is the workspace. At first, it appears with a cleaned floor without any object. When we load the scene using the "File" menu or drag the model from the model browser, it appears here.
(2) This is one Cuboid object. Once the object is placed on the floor or workspace, it appears with the XYZ axis. Three different colours define the axis. X is red, Y is green, and Z is blue. The object can be move or rotate by axis handle.
(3) This is a simulation Control toolset. From the left first triangle, the button is the "Start/resume " Simulation Button. The next double vertical thick line is the "Suspend simulation" button to pause the simulation at runtime. A square shape button is used to stop the simulation, and the rest button set is not so important, so we skipped the description.
(4) This toolbar is handy. From the left, the first crossed arrow button is the "Camera Pan" button. It is used to move workspace vertically or horizontally. Click the button first and then move the mouse as we need (clicking the mouse on the workspace). The entire workspace will move left-right or top-bottom. The curved crossed arrow button is the "camera rotate" button. It is used to rotate the whole workspace towards the X, Y, or Z-axis. The next top-bottom arrow button is the "Camera shift button." Using this button, our visible area moves near or far location. The next top-bottom, the left-right arrow, is the "Object/item shift" button. It is a valuable and essential button. This button is used to shift objects along with the X, Y, or Z buttons. Another important button is the "object/item rotate" button. It is used for rotation around.
(5) This is a menu bar. It is functionality like other standard software except for Add menu. It is an essential menu for our experiment. We will see later how to use Add menu.

(6) This Left Side toolbar has a vital role in our research. From the top, the second lens icon button name is "Scene object properties." It is a frequently used tool. We can change physical object parameters like width, height, etc. The f(x) signed button is the "Calculation module properties button." Using this, we can change the kinematics and dynamic parameter of a selected object. Another paper icon button is the scripts button. This button is required when we want to add a script associated with an object. From the bottom, the fifth button is the toggle button. To show or hide the model browser windows, we simultaneously press this button. The fourth button (From the bottom) is used to show or hide scene hierarchy windows.
(7) This is the name of the object. To change the object's name, we need to click on the object. The "I" shape cursor will blink. Then we should change the name and press enter.
(8) This part is the hierarchical model display. Whatever we placed the objects, it is always part of hierarchical structure.
(9) This is the model browser button. We can select from the categorized folder and drag the mouse to the workspace. The object will place on the floor.
(10) When we select from the model browser folder, all objects in the folder are displayed in this area.

5. EXPERIMENT :

(1) We can download CoppeliaSim from https://www.coppeliarobotics.com/downloads

(2) Open CoppeliaSim.

(3) Menu bar> Add> Primitive Shape>Cylinder> "OK".

(4) In the Scene hierarchy (see above 8) > double click on name (here "Cylinder") > Cursor will Blink> delete name by Back space> change name: "**Base**".

(5) Select "Base" > from left Toolbar, click on "scene object properties" (lensed Icon)> "View/modify geometry"> uncheck "Keep proportions"> set the values: **X[m]=0.0800**, **Y[m]= 0.0800**, **Z[m]=0.0700** > "Apply" > close two windows.

(6) Select "Base" > From Top toolbar, press "Object/item shift" (Four-sided arrow Icon) > Position Tab > Relative to "World"> Set the values: **X-cord. [m]=0.0000**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0350**. close the window. Top toolbar> "Object/item rotate" (rotate arrow icon) > Orientation tab> Relative to world and then set the values : **Alpha[deg.] =0.0000**, **Beta[deg.] =0.0000**, **Gamma[deg.] =0.0000**. close the window.

(7) Menu bar> Add> Joint> Revolute. Change name to" **J1**". Pressing control button, select J1 and then Base> right click on workspace>Edit> "Make last selected object parent" > select J1> "Scene object property" button> set Visual properties: Length[m]= 0.040, Diameter[m]=0.040> close the window. From top tool bar click on "Object/item shift" button> select Position tab>select Relative to "Parent frame"> set the values: **X-cord. [m]=0.0000**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.040**. close the window. from Top toolbar click on "Object/item rotate" button> Orientation tab> select Relative to "Parent frame" and set the value: **Alpha[deg.] =0.0**, **Beta[deg.] =0.000**, **Gamma[deg.] =0.00.** close the window. Select J1> Top toolbar click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame"> select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(8) Menu bar>Add> Primitive Shape>Cuboid> OK. Rename "**L1**". Pressing control button, select L1, then J1> menu bar> Edit> select " Make Last selected object parent" Select L1> click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.050**, **Y[m]=0.030**, **Z[m]=0.040.** close two windows. Select L1> top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.00**, **Y-cord. [m]=0.00**, **Z-cord. [m]=0.00>** close the window. Top tool bar> "object/item rotate" button>

orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =90.0**, **Beta[deg.] =0.000**, **Gamma[deg.] =0.000**. close the window.

(9) Menu bar>Add> Joint>Revolute. Rename "**J2**". Pressing control button, select J2 and then L1> Edit> select " Make Last selected object parent". Select J2> from left tool bar, select "Scene Object Properties" button> Length[m]: 0.050, Diameter[m]: 0.040. close the window. top toolbar>"Object/item shift" button > position tab> select "Parent frame"> set the values: **X-cord. [m]=0.0000**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000>**close the window. top toolbar> click on "object/item rotate" button>Orientation tab> select "Parent frame"> set the values: **Alpha[deg.] =0.0**, **Beta[deg.] =0.000**, **Gamma[deg.] =0.000**. close the window. Select J2> Top toolbar> click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame" > select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(10) Menu bar>Add> Primitive Shape>Cuboid> OK. Rename "**L2**". Pressing control button, select L2, then J2> menu bar> Edit> select " Make Last selected object parent" Select L2> click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.110**, **Y[m]=0.030**, **Z[m]=0.030.** close two windows. Select L2> top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.055**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000** > close the window. Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =0.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0**. close the window.

(11) Menu bar>Add> Joint>Revolute. Rename "**J3**". Pressing control button, select J3 and then L2> Edit> select "Make Last selected object parent". Select J3> from left tool bar, select "Scene Object Properties" button> Length[m]: 0.030, Diameter[m]: 0.040. close the window. top toolbar>"Object/item shift" button > position tab> select "Parent frame"> set the values: **X-cord. [m]=0.055**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000** >close the window. top toolbar> click on "object/item rotate" button>Orientation tab> select "Parent frame"> set the values: **Alpha[deg.] =0.0**, **Beta[deg.] =0.000**, **Gamma[deg.] =0.000**. close the window. Select J3> Top toolbar> click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame"> select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(12) Menu bar>Add> Primitive Shape>Cuboid> OK. Rename "**L3**". Pressing control button, select L3, then J3> menu bar> Edit> select " Make Last selected object parent" Select L3> click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.090**, **Y[m]=0.030**, **Z[m]=0.030.** close two windows. Select L3> top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.045**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000** > close the window. Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =0.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0**. close the window.

(13) Menu bar>Add> Joint>Revolute. Rename "**J4**". Pressing control button, select J4 and then L3> Edit> select "Make Last selected object parent". Select J4> from left tool bar, select "Scene Object Properties" button> Length[m]: 0.030, Diameter[m]: 0.040. close the window. top toolbar> "Object/item shift" button > position tab> select "Parent frame"> set the values: **X-cord. [m]=0.045**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000** >close the window. top toolbar> click on "object/item rotate" button>Orientation tab> select "Parent frame"> set the values: **Alpha[deg.] =0.0**, **Beta[deg.] =0.000**, **Gamma[deg.] =0.00**. close the window. Select J4> Top toolbar> click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame"> select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(14) Menu bar>Add> Primitive Shape>Cuboid> OK. Rename "**L4**". Pressing control button, select L4, then J4> menu bar> Edit> select " Make Last selected object parent" Select L4> click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.080**, **Y[m]=0.030**, **Z[m]=0.020**close two windows. Select L4> top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.040**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000** > close the window. Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =0.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0**. close the window.

(15) Menu bar>Add> Joint>Revolute. Rename "**J5**". Pressing control button, select J5 and then L4> Edit> select "Make Last selected object parent". Select J5> from left tool bar, select "Scene Object Properties" button> Length[m]: 0.030, Diameter[m]: 0.040. close the window. top toolbar> "Object/item shift" button > position tab> select "Parent frame"> set the values: **X-cord. [m]=0.040**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.0000** >close the window. top toolbar> click on "object/item rotate" button>Orientation tab> select "Parent frame"> set the values: **Alpha[deg.] =0.0**, **Beta[deg.] =90.00**, **Gamma[deg.] =0.000**. close the window. Select J5> Top toolbar> click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame" > select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(16) Menu bar>Add> Primitive Shape>Cylinder> OK. Rename "**EF_Spindle**". Pressing control button, select EF_Spindle, then J5 > menu bar> Edit> select " Make Last selected object parent" Select EF_Spindle > click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.030**, **Y[m]=0.030**, **Z[m]=0.035.** close two windows. Select L4> top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.000**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.035** > close the window. Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =0.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0**. close the window.

(17) Menu bar>Add> Joint> Prismatic. Rename "**EF_Left_Motor**". Pressing control button, select EF_Left_Motor and then EF_Spindle > Edit> select "Make Last selected object parent". Select EF_Left_Motor > from left tool bar, select "Scene Object Properties" button> Length[m]: 0.050, Diameter[m]: 0.020. close the window. top toolbar> "Object/item shift" button > position tab> select "Parent frame"> set the values: **X-cord. [m]=0.030**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.027** >close the window. top toolbar> click on "object/item rotate" button>Orientation tab> select "Parent frame"> set the values: **Alpha[deg.] =0.0**, **Beta[deg.] =-90.0**, **Gamma[deg.] =0.000**. close the window. Select EF_Left_Motor > Top toolbar> click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame"> select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(18) Menu bar>Add> Primitive Shape>Cuboid> OK. Rename "**EF_Left_Support**". Pressing control button, select EF_Left_Support, then EF_Left_Motor > menu bar> Edit> select " Make Last selected object parent" Select EF_Left_Support > click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.100**, **Y[m]=0.010**, **Z[m]=0.040.** close two windows. Select EF_Left_Support > top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.030**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=-0.020** > close the window. Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =-90.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0**. close the window.

(19) Menu bar>Add> Joint> Prismatic. Rename "**EF_Right_Motor**". Pressing control button, select EF_Right_Motor and then EF_Spindle > Edit> select "Make Last selected object parent". Select EF_Right_Motor > from left tool bar, select "Scene Object Properties" button> Length[m]: 0.050, Diameter[m]: 0.020. close the window. top toolbar>"Object/item shift" button > position tab>

select "Parent frame"> set the values: **X-cord. [m]=-0.030**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.027** >close the window. top toolbar> click on "object/item rotate" button>Orientation tab> select "Parent frame"> set the values: **Alpha[deg.] =0.0**, **Beta[deg.] =-90.0**, **Gamma[deg.] =0.000** close the window. Select EF_Right_Motor > Top toolbar> click on "Object/item rotate" button> "Mouse Rotation" tab> select "Own frame"> select "about Z"> click and drag the mouse pointer left to right, our joint will rotate round Z axis. It means we successfully done the job.

(20) Menu bar>Add> Primitive Shape>Cuboid> OK. Rename "**EF_Right_Support**". Pressing control button, select EF_Right_Support, then EF_Right_Motor > menu bar> Edit> select " Make Last selected object parent" Select EF_Right_Support > click "Scene object property" button >click "View/modify geometry" button>uncheck "keep proportions" and set the value: **X[m]=0.100**, **Y[m]=0.010**, **Z[m]=0.040.** close two windows. Select EF_Right_Support > top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.030**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.020** > close the window. Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value> **Alpha[deg.] =-90.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0.** close the window.

(21) Menu bar>Add > dummy. Rename "**tip**". Pressing control button, select "tip" and then EF_Spindle> Edit menu> select "Make Last selected object parent"> select "tip"> click "Scene object property"> set the value Object size[m]: 0.020> close the window. top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=0.000**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.100.** Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value>: **Alpha[deg.] =0.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0.**

(22) Menu bar>Add > dummy. Rename "**target**". select" target"> click "Scene object property"> set the value Object size[m]: 0.020> close the window. top tool bar> "Object/item shift" button > position tab> select "Parent frame" and set the value: **X-cord. [m]=-0.125**, **Y-cord. [m]=0.0000**, **Z-cord. [m]=0.075.** Top tool bar> "object/item rotate" button> orientation tab> select Relative to "Parent frame" and set the value>: **Alpha[deg.] =0.0**, **Beta[deg.] =0.0**, **Gamma[deg.] =0.0.**

(23) Pressing control key, select "tip" and "target"> right-click on workspace>Edit> Link selected dummies> Edit menu> Link selected dummies> select IK,tip-Target. A red-colored both side arrow line be visible in the scene hierarchy.

(24) Select join J1> from left side toolbar select "scene object properties" button > "Mode" Listbox> select "Inverse Kinematics mode">checked Hybrid operation. Close the window. We must repeat this procedure for all joints J1-J5, EF_Left_Motor, and EF_Right_Motor

(25) From Left tool bar > press Calculation module property button(f(x) icon)> Kinematics tab> Add new IK group> Press "Edit IK elements"> from right side list box, select "tip"> click "Add new IK element with tip" button> close both windows.

(26) Select "Base"> Scene Object Properties button> click on "Show dynamic properties dialog button> uncheck "Body is dynamic"> close window. This procedure will follow for Base, L1, L2, L3, L4, EF_Spindle, EF_Left_Support, and EF_Right_Support.

(27) Select J1. From left tool bar select scene object property button. Uncheck "position is cyclic" and set the parameter below. Pos. min.[deg.] =0.00, Pos. range[deg.] =180.00, Position[deg.] =0.00.

(28) As step 27, click joint J2-J5, EF_Left_Motor, EF_Right_Motor one by one and set the joints parameter value as below:

Table 1: Joint Value Parameter

S.L. No.	Name	Value		
		Min	**Max**	**Default**
1	J1	0.0000	180.0000	0.0000
2	J2	45.0000	90.0000	95.0000
3	J3	0.0000	180.0000	30.0000
4	J4	0.0000	120.0000	0.0000
5	J5	0.0000	360.0000	0.0000
6	EF_Left_Motor	-1.0000	1.0000	0.0000
7	EF_Left_Motor	-0.5000	1.0000	0.0000

(29) We can set the colour to each object we created for beautification. Here we will see how L2 objects can be applied colour. Select L2> Click on "scene object properties" button> press “Adjust colour" Button> press "Ambient/diffuse component"> select color using Red, Green, and Blue slider. We can apply the colour for J1-J5, L1-L4, EF_Spindle, EF_Left_Motor, EF_Right_Motor, EF_Left_Support, EF_Right_Support.

(30) Run the simulation. Select "Target." Click on "Object/item shift" button> select Mouse Translation Tab> select Relative to "World"> select only "along Z">drag the mouse up and down. We can see our End Effector is following our target object. Finally, our robots will look like as in figure 2.

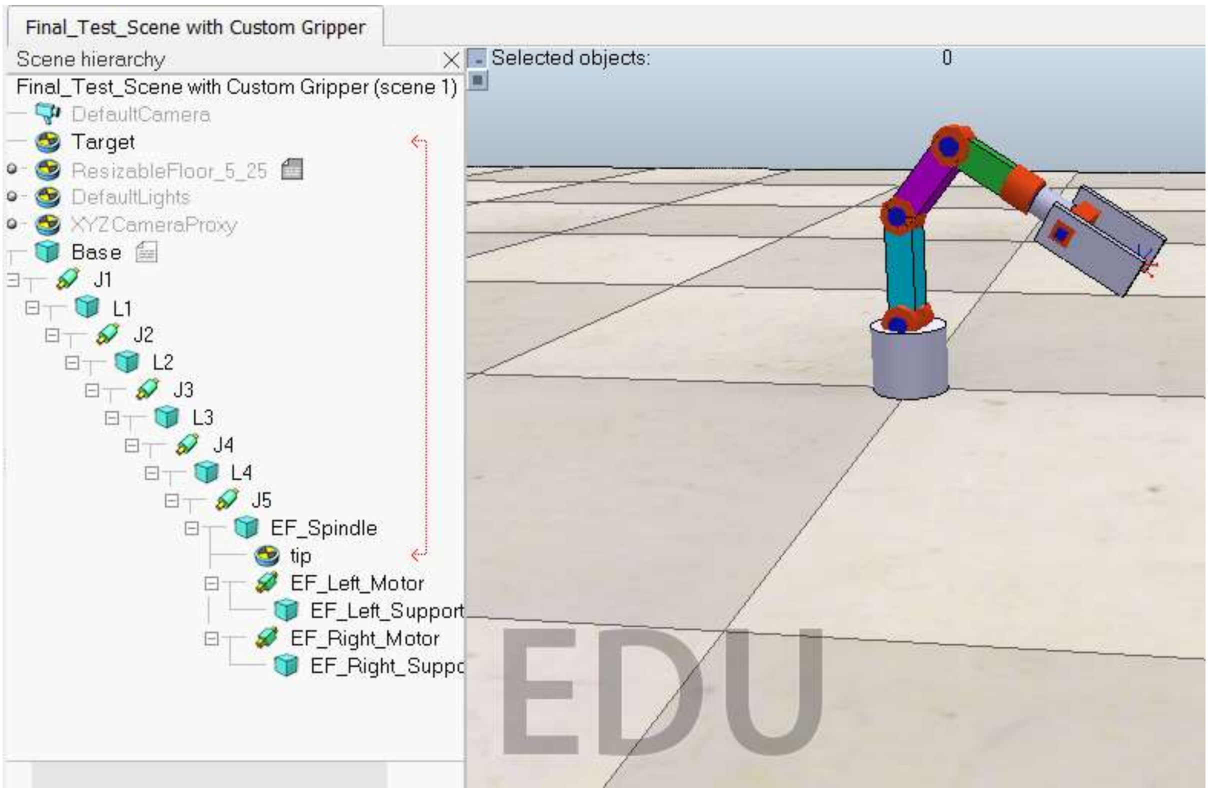

Fig. 2: Complete view of Robotics ARM

Table 1: List of geometrical parameters of used object

S.L. No.	Item	Name	Geometry				
			X[m]	Y[m]	Z[m]	Length[m]	Diameter[m]
1	Primitive	Base	0.080	0.080	0.070		

	shape>Cylinder		0	0	0		
2	Joint>Revolute	J1				0.0400	0.0400
3	Primitive shape>Cuboid	L1	0.050	0.030 0	0.040 0		
4	Joint>Revolute	J2				0.0500	0.0400
5	Primitive shape>Cuboid	L2	0.110 0	0.030 0	0.030 0		
6	Joint>Revolute	J3				0.0300	0.0400
7	Primitive shape>Cuboid	L3	0.090 0	0.030 0	0.030 0		
8	Joint>Revolute	J4				0.0300	0.0400
9	Primitive shape>Cuboid	L4	0.080 0	0.030 0	0.020 0		
10	Joint>Revolute	J5				0.0300	0.0400
11	Primitive shape>Cylinder	EF_Spindle	0.030 0	0.030 0	0.035 0		
12	Joint>Prismatic	EF_Left_Motor				0.0500	0.0200
13	Primitive shape>Cuboid	EF_Left_Suppo rt	0.100 0	0.010 0	0.040 0		
14	Joint>Prismatic	EF_Right_Mot or				0.0500	0.0200
15	Primitive shape>Cuboid	EF_Right_Sup port	0.100 0	0.010 0	0.040 0		
16	Dummy	tip	0.0200				
17	Dummy	Target	0.0200				

Table 2: List of position and orientation parameter of used object

S.L. No.	Name	Position			Orientation			Relative to
		X-cord. [m] (a)	Y-cord. [m]	Z-cord. [m] (d)	Alpha[deg.] (a)	Beta [deg.]	Gamma [deg.] (q)	
1	Base	0.000 0	0.000 0	0.035 0	0.0000	0.0000	0.0000	World
2	J1	0.000 0	0.000 0	0.040 0	0.0000	0.0000	0.0000	Base
3	L1	0.000 0	0.000 0	0.000 0	90.0000	0.0000	0.0000	J1
4	J2	0.000 0	0.000 0	0.000 0	0.0000	0.0000	0.0000	L1
5	L2	0.055 0	0.000 0	0.000 0	0.0000	0.0000	0.0000	J2
6	J3	0.055 0	0.000 0	0.000 0	0.0000	0.0000	0.0000	L2
7	L3	0.045 0	0.000 0	0.000 0	0.0000	0.0000	0.0000	J3
8	J4	0.045 0	0.000 0	0.000 0	0.0000	0.0000	0.0000	L3
9	L4	0.040 0	0.000 0	0.000 0	0.0000	0.0000	0.0000	J4
10	J5	0.040 0	0.000 0	0.000 0	0.0000	90.000 0	0.0000	L4

11	EF_Spindle	0.0000	0.0000	0.0350	0.0000	0.0000	0.0000	J5
12	EF_Left_Motor	0.0300	0.0000	0.0270	0.0000	-90.0000	0.0000	EF_Spindle
13	EF_Left_Support	0.0300	0.0000	-0.0200	-90.0000	0.0000	0.0000	EF_Left_Motor
14	EF_Right_Motor	-0.0300	0.0000	0.0270	0.00	-90.0000	0.0000	EF_Spindle
15	EF_Right_Support	0.0300	0.0000	0.0200	-90.00	0.0000	0.0000	EF_Right_Motor
16	tip	0.0000	0.0000	0.1000	0.00	0.0000	0.0000	EF_Spindle
17	Target	-0.1250	0.0000	0.0750	0.000	0.0000	0.0000	World

Table 3: List of D.H. Parameter of Custom Robot

Link	q	d(meter)	a(meter)	a(Deg.)
1	0.0000	0.0400	0.0000	0.0000
2	0.0000	0.0000	0.0000	90.0000
3	0.0000	0.0000	0.0550	0.0000
4	0.0000	0.0000	0.0400	0.0000
5	0.0000	0.0000	0.0000	0.0000
6	0.0000	0.0270	0.0300	0.0000
7	0.0000	0.0270	-0.0300	-90.0000

6. RECOMMENDATIONS :

We should follow some suggestions to create a custom robot to get better and perfect results.

- ❖ Create a D.H. parameter first. It makes us easy to design the robot.
- ❖ All parameters in the entry box are in meter metrics.
- ❖ When we set various Object parameter, need to be careful on decimal position.
- ❖ It is a serial manipulator and series of ARM and joints. After one joint and ARM integration, we can test using the RUN simulation button. It will not mess up with other's object parameters and easy to debug.
- ❖ After recreating the experiment using given D.H. parameter, we can change the D.H. Parameter and continue experiment.
- ❖ At first approach, I would recommend observing the complete ARM first. Then to proceed to create a new one. It will build up our confidence and willing power to replicate it.
- ❖ We have listed our used parameters depicted in Fig. 5.3 and Fig. 5.4 for researcher reference. Advance users of CoppeliaSim can use and replicate experiments easily.
- ❖ Fig. 5.5, we listed out the D.H. parameter of our custom Robot. Changing parameter value researchers can be made for their custom robot.
- ❖ To download the scene file, we can visit the below link: https://github.com/sudipchakraborty/Custom-Robotics-Arm-in-CoppeliaSim.

7. CONCLUSION :

The robot simulator is an easy way to experiment with the robot in the field of robotics study. One good Simulator can get perfect and desired results. When the practical robot is not available, the only way to carried out the research is Simulator. Getting an excellent and reliable simulator demands a lot of effort.

Here we introduced CoppeliaSim, one of the best robot simulators now. It is free and Open source. Using it, we can experiment with robotic ARM for our research purpose.

REFERENCES

[1] Huck, T. P., Ledermann, C. and Kröger, T. (2020). Simulation-based Testing for Early Safety-Validation of Robot Systems. IEEE Symposium on Product Compliance Engineering - (SPCE Portland), Portland, OR, USA, 2020, pp. 1-6, DOI: 10.1109/SPCE50045.2020.9296157.

[2] Kortmann, M., Zeis, C., de Alba-Padilla, C. A., Grzesik, B., Schroeder, K. U. and Stoll E. (2020). New approach on robotic arm design: fully modular arm architecture utilizing novel space interface. i-SAIRAS2020-Papers (2020). Virtual Conference 19–23 October 2020.

[3] Javier Pinzon-Arenas, Robinson Jimenez-Moreno, and Astrid Rubiano (2020). Virtual environment for smart robotic applications. *ARPN Journal of Engineering and Applied Sciences*, 15(22), 2698-2705.

[4] Igor Shardyko, Maria Samorodova, Victor Titov (2020). Decentralized Control of Robotic Arm with Elastic Joints. International Russian Automation Conference (RusAutoCon), 978-1-7281-6130-3/20, pp. 615-620.

[5] John Oyekan, Michael Farnsworth, Windo Hutabarat, David Miller and Ashutosh Tiwari (2020). Applying a 6 DoF Robotic Arm and Digital Twin to Automate Fan-Blade Reconditioning for Aerospace Maintenance, Repair, and Overhaul. *Sensors*, 20 (1), 4637, DOI: 10.3390/s20164637.

[6] Jiawei Hou, Yizheng Zhang, Andre Rosendo and S¨oren Schwertfeger (2020). Mobile Manipulation Tutorial. https://robotics.shanghaitech.edu.cn/static/robotics2020/MoManTu_Intro.pdf

[7] Jan Schneider, Tim Schneider, Boris Belousov, Georgia Chalvatzaki, Samuele Tosatto, Bastian Wibranek, "Architectural Assembly with Tactile Skills: Simulation and Optimization," https://www.ias.informatik.tu-darmstadt.de/uploads/Team/BorisBelousov/schneider_ip.pdf. Retrieved in 01/03/2021.

[8] Tamir Blum, Kazuya Yoshida (2020). PPMC RL Training Algorithm: Rough Terrain Intelligent Robots through Reinforcement Learning. arXiv:2003.02655.

[9] Mehrnoosh Askarpour, Matteo Rossi, Omer Tiryakiler (2020). Co-Simulation of Human-Robot Collaboration: from Temporal Logic to 3D Simulation. Robots for reliable Engineered Autonomy (AREA'20). EPTCS 319, 2020, pp. 1–8, DOI:10.4204/EPTCS.319.1.

[10] Omar Gamal, Xianglin Cai, Hubert Roth (2020). Learning from Fuzzy System Demonstration: Autonomous Navigation of Mobile Robots in Static Indoor Environment using Multimodal Deep Learning. International Conference on System Theory, Control and Computing (ICSTCC), 978-1-7281-9809-5/20, pp 218-225.

[11] Md Nizamuddin Ahmed, Veladri, K. (2016). Modeling and Simulation of 7-DOF Robotic Manipulator", National Conference on Technological Advancements in Mechanical Engineering, ISBN: 978-93-85100-57-4.

[12] Samuel Abhishek S. Veladri, K. (2016). Trajectory Planning of a Mobile Robot. National Conference on Technological Advancements in Mechanical Engineering, ISBN: 978-93-85100-57-4.

[13] Boris Bogaerts, Seppe Sels, Steve Vanlanduit, Rudi Penne (2020). Connecting the CoppeliaSim robotics simulator to virtual reality, *SoftwareX*, 11(1), 100426. https://doi.org/10.1016/j.softx.2020.100426.

[14] Fernando Joventino, de Oliveira, C. A. S., Alberto Fabro, J. and Pereira, J. H. M. (2020). Application of a ROS / CoppeliaSim Integration in a Practical "OBR" Competition Scenario. Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and

Workshop on Robotics in Education (WRE), Natal, Brazil, pp. 1-6, DOI: 10.1109/LARS/SBR/WRE51543.2020.9306988.

[15] Sun, Z., Li, D., Huang, L., Liu, B., and R. Jia (2020). Construction of intelligent visual coal and gangue separation system based on CoppeliaSim. 5th International Conference on Automation, Control and Robotics Engineering (CACRE), pp. 560-564, DOI: 10.1109/CACRE50138.2020.9230077.

[16] Alshorman A. M., Alshorman O., Irfan M., Glowacz A., Muhammad F., Caesarendra W. (2020). Fuzzy-Based Fault-Tolerant Control for Omnidirectional Mobile Robot. *Machines*, *8*(3), 55. https://doi.org/10.3390/machines8030055

[17] Ivan Virgala, Michal Kelemen, Erik Prada, Marek Sukop, Tomáš Kot, Zdenko Bobovský, Martin Varga, Peter Ferenčík (2021). A snake robot for locomotion in a pipe using trapezium-like travelling wave. *Mechanism and Machine Theory*, 158 (1), 104221.

[18] James, S., Ma, Z., Arrojo D. R. and Davison, A. J. (2020). RLBench: The Robot Learning Benchmark & Learning Environment. *IEEE Robotics and Automation Letters*, 5(2), 3019-3026, DOI: 10.1109/LRA.2020.2974707.

[19] Scatambulo Costa, C. F. and Alexandre Franciscon, E. (2020). Data Mining applied to the navigation task in autonomous robots. IEEE Symposium on Computers and Communications (ISCC), pp. 1-6, DOI: 10.1109/ISCC50000.2020.9219731.

[20] Cheng, Hong, Ruixue Jia, Dandan Li, and Hongbin Li. (2019). The Rise of Robots in China. *Journal of Economic Perspectives*, 33(2), 71-88. DOI: 10.1257/jep.33.2.71

[21] Al, G. A., Estrela, P. and Martinez-Hernandez, U. (2020). Towards an intuitive human-robot interaction based on hand gesture recognition and proximity sensors. IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp.330-335, DOI: 10.1109/MFI49285.2020.9235264.

[22] Xinyang Tian, Qinhuan Xu, Qiang Zhan (2021). An analytical inverse kinematics solution with joint limits avoidance of 7-DOF anthropomorphic manipulators without offset. *Journal of the Franklin Institute*, 358(2), 1252-1272.

[23] Zhou, Y., Lin, J., Wang, S. and Zhang, C. (2021). Learning Ball-Balancing Robot through Deep Reinforcement Learning. International Conference on Computer, Control and Robotics (ICCCR), pp. 1-8, DOI: 10.1109/ICCCR49711.2021.9349369.

[24] Choi, J., Kim, H., Son, Y., Park, C. W., and Park, J. H. (2020). Robotic Behavioral Cloning Through Task Building. International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), pp. 1279-1281, DOI: 10.1109/ICTC49870.2020.9289148.

****