German University in Cairo (GUC)
Faculty of Engineering & Materials Science (EMS)
Mechatronics Department (MCTR)

Project Milestone 02

Objective:

The second milestone of the project is with the following details:

Milestone 02	Description	Deadline
M-02 (7%)	**Requirements:** ● Choose either to import the **URDF file** provided by course instructors on the CMS or use **CoppeliaSim robots**. ● Assign the **frames** on the assigned arm (**on paper** or by the help of the frames present in **Simscape Multibody model** or in **CoppeliaSim model**). ● Develop the **DH convention** of the robotic arm to obtain the **Forward Position Kinematics** equations using **MATLAB/Simulink** or **Python** codes. ● Place the **CAD assembly** of the arm in **Simscape Multibody** or **CoppeliaSim or any other simulator** through the linking between **SOLIDOWRKs** and **MATLAB/Simulink** and upload the **URDF file** to **CoppeilaSim** by integrating **Python**. VODs are provided on CMS. ● Test the kinematic equations obtained on the **chosen simulator** by **inputting joint angles and sensing the position of the end effector** to be approximately the same as that obtained by the **kinematics equations**. ● **Start** building the simulation's **GUI** by inserting objects in the scene related to your application (ex: chairs, tables, people, conveyors, etc.)	**Friday 18th of October, 2024**

Requirements:

The requirements for this milestone of the project are as follows:

1. Each team is required to choose either to import the **URDF file** provided by course instructors on the CMS or use **CoppeliaSim robots** please choose a limited DoF (up to 4 DoF).
2. Each team is required to assign the robot's frames following the DH-convention (X-Z axes ONLY). This requirement should be done **on paper, on Simscape or on CoppeliaSim or any other chosen simulator**. **(You have to make sure that the frames placed on the Revolute joints of Simscape or CoppeliaSim or any other chosen simulator while actuating the joints by angle Zero (initial position of the robot) are realistic and same as the ones in the simulator to avoid any problems in the analysis)** (it is preferred to place a screenshot of your robot of the paper work, Simscape or CoppeliaSim on a Word document having arrows inserted on the joints to represent the X-Z axes of each joint)).
3. Make sure that the assigned axes for each joint in the robot done in the first requirement of this milestone are the same to those placed on each joint on the **Simscape Multibody** or **CoppeliaSim** or any chosen simulator of your arm. If the axes are not the same (rotated by a certain angle that would be positive or negative 90 degrees in most to the cases), then use a rigid transform block before the revolute joint that has the transformed frame and edit the rotation matrix to transform the frame to be identical to that assigned in the previous requirements for the simulation to be the same as the system's mathematical representation.

Project Milestone 02

4. Evaluate the **DH-parameters** to fill the DH-parameters table for your robot using the assigned frames.
5. Evaluate the **Forward Position Kinematics** of your robot by performing the following steps on **Python environment** as all of the team used **CoppeliaSim**:
   a. Prepare a function (Python) to evaluate the Total Homogeneous Transformation Matrix (T) between each joint (i) and its preceding joint ($i-1$). The script should be represented as a function that takes as input each row in the obtained DH-parameters table as angles and lengths of the robot and outputs the corresponding homogeneous transformation function. The function should be named as: **T = transformation_func(θ,d,a,α)**. <u>(Note that the function should take as input the angles of the robot in a symbolic manner and the lengths as constant numbers to output a symbolic transformation function).</u>
   b. Prepare a function (Python) to obtain the position of the end effector from the multiplication of the transformation functions obtained. The function should be named as follows: **X = forward_kinematics_func()**. <u>(Note that the function should call the previously created function **T = transformation_func(θ,d,a,α)** several times (indicating the number of rows in the filled DH table) and multiply these matrices to obtain the overall matrix between the base frame and the end effector, then extract the position of the end effector. This function should output a vector having the X, Y and Z positions of the end effector in a symbolic manner).</u>

   <u>For CoppeliaSim:</u>

   a. <u>**Python to CoppeliaSim:**</u> In the scene hierarchy, right click on the joint or object you need to add a script to then press Add -> Script -> Simulation Script -> non-Threaded -> Python. A new icon will appear below the joint, double click it to open the python script. Edit the script with your code, save and run the simulation from play button in toolbar.
   b. <u>**CoppeliaSim:**</u> Simulate the motion of the robot using the same input angles used in the forward position kinematics equations obtained previously and visualize the motion of the robot's end effector and compare it with the output responses for the X, Y and Z axes of the robot obtained by the calculations of the forward position kinematics (DH-convention).
6. **Start** building the **simulation environment (GUI)** of the system by adding some objects which will serve to your application (ex: chairs, tables, other robots, people, conveyors, boxes, etc.).

<u>Submission:</u>

Create another folder in the previously created **GitHub repository** named **"Milestone 02"** including the following:

1. **Word file** includes the following. **Note build on the report done in the previous milestone:**
   a. The **coordinate frame** assignment.
   b. The **DH-convention table**, the **DH-convention final matrix** (Total Homogeneous Transformation Matrix between the end effector and the base frame).

Project Milestone 02

   c. Screenshots of the simulations performed and comments on the results of the cases tested. Results of **Simscape Multibody (MATLAB/Simulink)** or **CoppeliaSim (Python) or the other chosen simulator** including the comments on the system performance based on the different inputs provided and the stating the limitations/constraints of the system.
   d. The built **GUI.**
2. **Script folder** includes the following items
   a. **Python.py files** of the required python codes
3. **Media folder** that includes the following **Narrated Videos**:
   a. **Video1.mp4** for the **simulator used** and **MATLAB or Python functions** responses. You should include in the video the analyses of the motion of the robot, commenting on the performance in terms of the rotations performed for each joint (about which axis) taking into consideration the coordinate frames placed on the robot links and joints and commenting also on the limitations/constraints of the system's motion.
   b. **Video2.mp4** for the building process of the GUI process including the robot and any other object added to the environment.

The deadline of the **submission** is on **Friday 18th of October, 2024 at 11:59 PM**.
Late Submissions will result in deduction from the grade of this deliverable.