**Robotics Simulation Package User Manual**

**MAE547 Final Project, Fall 2020**

**Team 1**

**Authors:** Lucas Petersen, Anna Rothweil, Steven Chasen, Jaehyuk Choi

**Special Thanks to:** Dr. Hamid Marvi, Peter Corke

Table of Contents

[Introduction 3](#_Toc57660225)

[Installation 5](#_Toc57660226)

[Troubleshooting 5](#_Toc57660227)

[Welcome 6](#_Toc57660228)

[Arm Definition 6](#_Toc57660229)

[Simulation – Home Page 8](#_Toc57660230)

[Plot Robot Arm 10](#_Toc57660231)

[Drive Robot Arm 10](#_Toc57660232)

[Print Equations of Motion 11](#_Toc57660233)

[Simulate System Dynamics 12](#_Toc57660234)

[Indirect Force Control – Compliance 13](#_Toc57660235)

[Indirect Force Control – Impedance 14](#_Toc57660236)

# Group Member Contributions

|  |  |
| --- | --- |
| **Member Name** | **Contribution** |
| Lucas | Developed GUI, implemented arm definition functionality, assisted with integrating other components into GUI, assisted with dynamics simulation, wrote the user manual |
| Anna | Developed function and Simulink model for compliance control, assisted with impedance control and overall Simulink integration, assisted with GUI elements |
| Steven | Developed DH parameter finding utility for Arm Definition, developed function that returns equations of motion, assisted with impedance and compliance control, assisted with GUI elements |
| Jae | Developed dynamics simulation function and plots, developed function and Simulink model for impedance control, assisted with GUI elements |

# Introduction

This manual's objective is to familiarize the user with the functionality of the robotics simulation package created by Team 1. The main capabilities of the application are as follows:

1. Set up a serial robotic arm comprised of any combination of links with revolute or prismatic joints and define a wide variety of parameters for each link and joint (inertia, mass, gear ratio, friction coefficients, etc.). If the Denavit-Hartenberg (DH) parameters are unknown, they will be calculated.
2. Output symbolic equations of motion governing the dynamics of the robotic arm, with or without friction.
3. Simulate system dynamics by plotting the position, velocity, and acceleration of the joint variables based on input torques.
4. Perform indirect force control through compliance control: PD control with gravity compensation, based on environmental forces.
5. Perform indirect force control through impedance control: inverse dynamics control based on environmental forces.
6. Plot the robotic arm, adjust the joint variables, and display a 3D animation of the arm.

The simulation package uses a wide variety of code from the Corke Robotics Toolbox, Version 9 (included with the installation). The application implements the toolbox code in a straightforward and easy to use GUI that simplifies the process of performing these standard calculations.

# Installation

To install and open the application, un-zip the compressed folder containing the files. Run the MATLAB application called “RoboticsPackage.mlapp”. Please do not remove or modify files within the unzipped directory, or it may cause errors.



# Troubleshooting

The application has been tested for a wide variety of robotic arm configurations. However, you may encounter an error. To troubleshoot errors, considering performing the following steps:

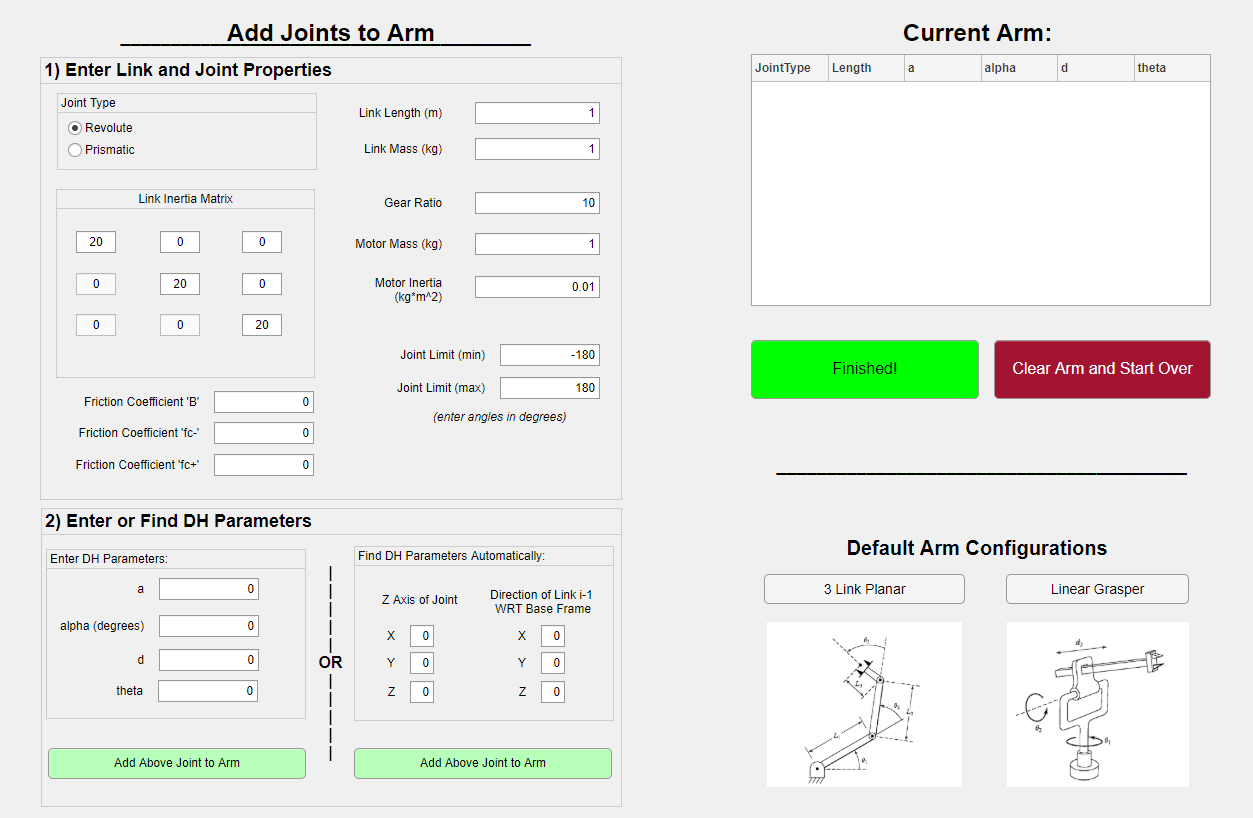
* Consult the user manual and on-screen prompts to ensure that you have correctly entered all inputs
* Close the application and re-open it
  + This is likely to address any issues that may arise when performing an analysis with a one-arm configuration, clearing it, and starting again
* “Re-install” the application by copying the compressed file and extracting to a clean directory
* Make sure you have correctly set up the arm before attempting other analyses

# Welcome

This screen is displayed upon opening the application. Please press “Start” to proceed to the arm setup screen.

# Arm Definition

This screen is used for setting up the robotic arm, one link and joint at a time.



Proceed as follows:

1. Enter the Link and Joint Properties
   1. Fill out all the boxes in this panel. If you are unsure of a certain parameter's value, you can leave the value at the default.
   2. If you do not wish to consider friction in your analyses, leave the friction coefficients as zeros.
2. Enter or Find the DH Parameters
   1. If you know the DH Parameters for your arm, enter them.
   2. If you do not know the DH parameters, enter the direction vectors for the Z-axis of the joint and the direction of Link i-1 with respect to the base frame (as in the Denavit-Hartenberg convention).
3. Press “Add Above Joint to Arm”
   1. The table on the right will update with a summary of that link
4. Continue until you have set up all links
5. Press “Finished” to proceed to other analyses

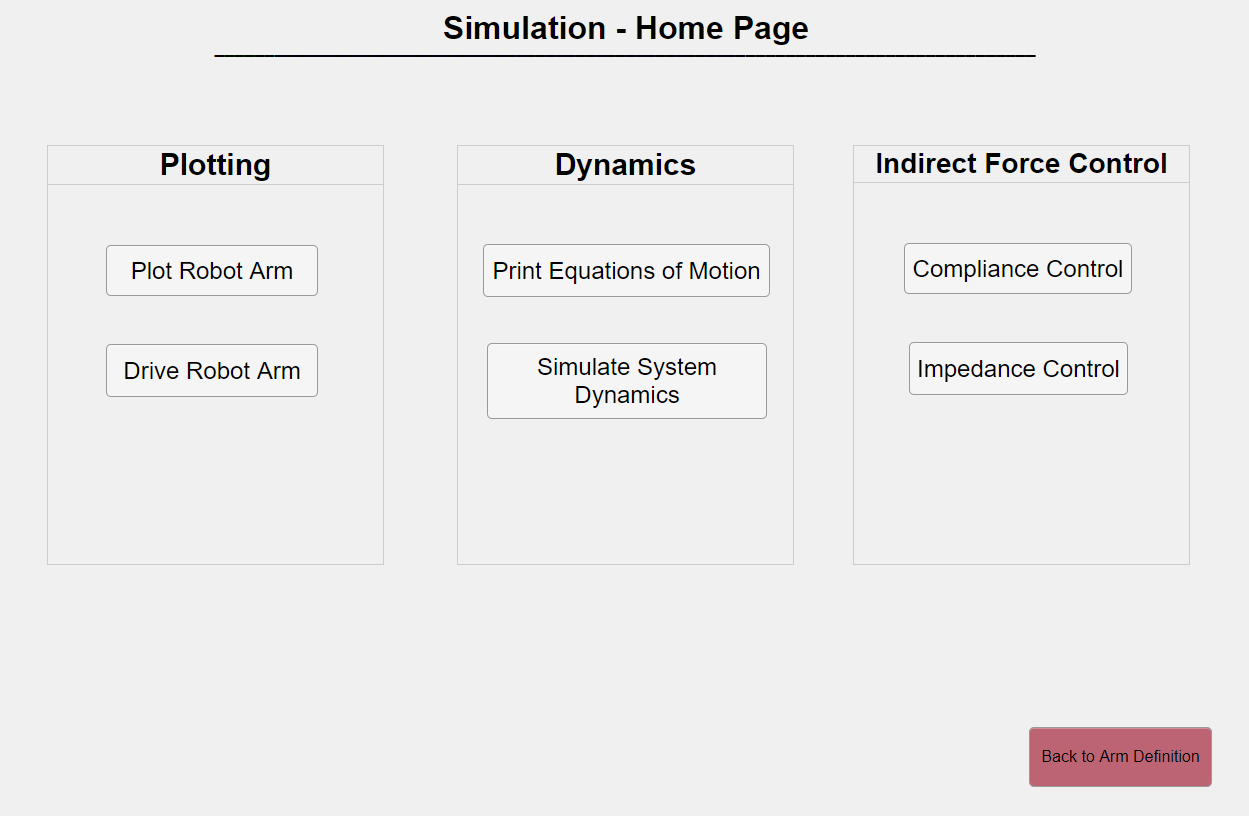
Alternatively, you may choose one of the “Default Arm Configurations”. The options are a 3-Link Planar Arm or a Linear Grasper. These arms are useful for plotting and testing, but the dynamic properties, link lengths, etc. cannot be changed, so manually setting up an arm is advised.

**Caution:**

* Angles (joint limits, alpha) should be entered in degrees
* Be mindful of units listed next to each quantity and convert if necessary. The link inertia matrix is in units of .
* Ensure that the DH parameters match the other properties (for example, link length and “a” should be the same in a planar arm with only revolute joints)
* The DH parameters should be entered in SI base units when relevant

# Simulation – Home Page

On this screen, you can use the package to perform a variety of simulations. This is the main landing page for the app once the arm has been set up.



The options are:

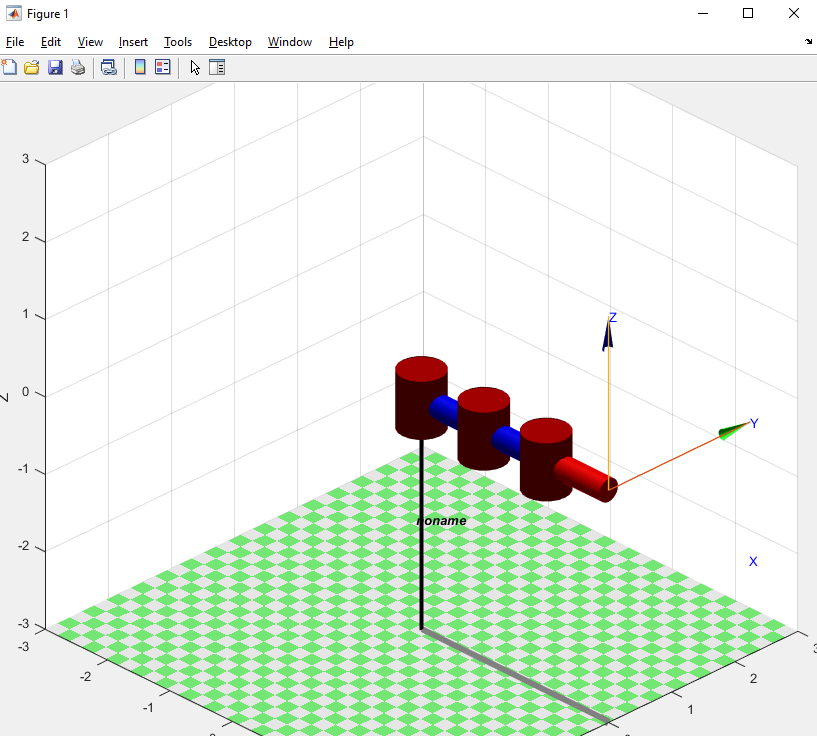
* **Plot Robot Arm:** Plots the arm in the default configuration (all joint variables set to zero). Useful for checking that the arm has been set up properly
* **Drive Robot Arm:** Adds a graphical overlay to the plot that allows the user to change the joint variables with sliders (up to the joint limits) and view an animation of the robotic arm moving in response
* **Print Equations of Motion:** Prints the relevant matrices and symbolic equations of motion to the MATLAB command window
* **Simulate System Dynamics:** Allows the user to input joint torques and view the resulting joint variable dynamics over time
* **Compliance Control:** Move the arm to a specified end effector position and pose using gravity compensated compliance control
* **Impedance Control:** Move the arm to a specified end effector position and pose using impedance control

You can return to this tab at any time using the tabs at the top of the application. Click the tab labeled “Simulation” to return here.

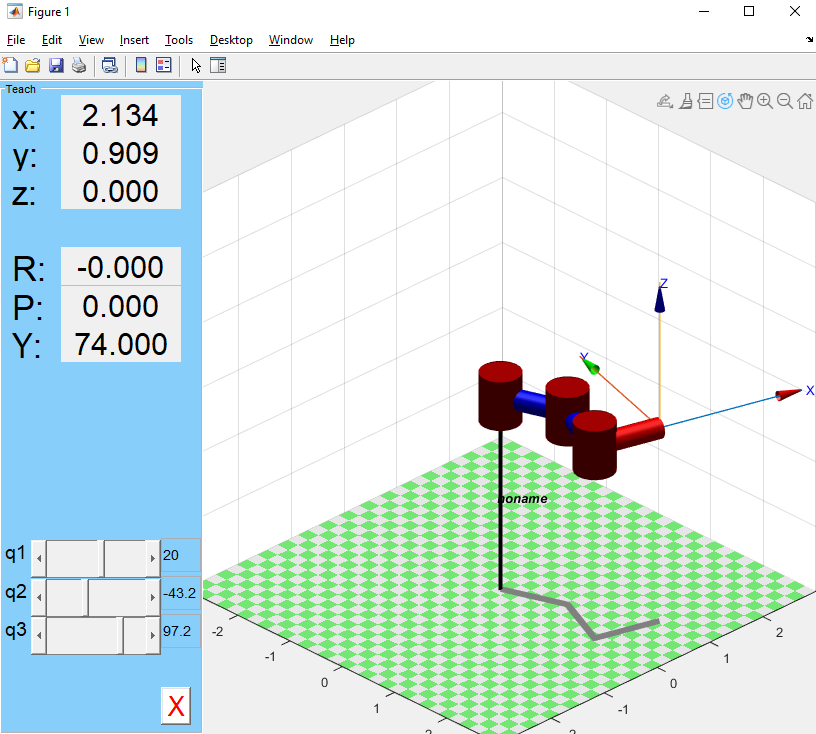


The “Back to Arm Definition” button can be used to return to the Arm Definition tab – alternatively, the Arm Definition tab at the top can be selected.

# Plot Robot Arm

 Pressing this button opens a graphical window with a current arm plot when all joint variables are equal to zero. See the example on the left for the default 3-link planar arm. Click and drag to rotate and see the view in 3D.

# Drive Robot Arm

 This button creates a new plot or a graphical overlay on the current plot that allows the user to change the joint variables with sliders and view the resulting configuration. See the example on the left for the 3-link planar arm. The joint variables can only be adjusted up to the joint limits set by the user (for the default configurations, the limits are 0 to 1 for prismatic joints and - to for revolute joints).

# Print Equations of Motion

This button produces symbolic forms of the matrices that are used to define dynamic effects on the robot arm. If all the information has been filled out in the Arm definition tab, just clicking the button will immediately compute the equations of motion for your arm. The outputs are shown in the command window and are as follows:

* **B:** The effects on the arm due to inertia effects on the arm
* **C:** The effects on the arm due to centrifugal and Coriolis effects
* **G:** The effects on the arm due to gravity effects
* **Je:** A matrix representing the relationship between joint velocities and end effector velocities, or generalized forces on the end effector and generalized torques for a static arm.
* **Fpos & Fneg:** The effects on the arm due to frictional effects for positive and negative joint velocities, respectively. It is possible for some joint velocities to be positive and some to be negative.
* **Full\_EOMpos & Full\_EOMneg:** The generalized torque on the joints when joint velocities are positive and negative respectively.

# Simulate System Dynamics

On this screen, the user inputs are used to perform a dynamics simulation based on applied joint torques and plot the resulting joint variable positions, velocities, and accelerations. Proceed as follows:

1. Define Simulation Parameters
   1. Enter the initial values of the joint variables as a comma separated list
   2. Enter the initial joint variable velocities as a comma separated list
   3. Enter the finish time for the simulation (begins at t = 0 sec)
2. Define Joint Torques
   1. The user can choose three methods for defining the joint torques:
      1. Enter Constant Values: Enter a comma separated list of the joint torques (in N-m)
      2. Enter Function by Hand: Enter a comma separated list of functions of “t”, that describe the joint torques as functions of time. For example: “t+1, sin(t), 3”
      3. User Provided Function File: The user can provide the name of a function that has been placed in the same directory which outputs the joint torques. For example: “myTorqFun”
3. Run the Simulation

**Caution:**

* When using the user provided function, make sure that the output has the correct dimensions. Follow the torque function instructions in the Corke Robotics Toolbox Manual under “SerialLink.fdyn”
* Do not forget the commas between joint torques, or an error will occur
* Do not use a variable other than “t” when entering the function by hand

# Indirect Force Control – Compliance

# Indirect Force Control – Impedance