Term Project

The term project will be based on the following robotic platform (you may choose another robot of sufficient complexity subject to my permission and approval). The end effector of this robot is the tip of the sword.

This term project is to be completed individually with one exception. Students may work in a group, with the condition that each additional group member completes an extra credit requirement. For example, a group of four should complete the main requirements and 3 extra credit requirements.

Source: https://mymodernmet.com/david-bowen-plant-machete/

Requirements:

You must submit a code package (in any programming language) along with a clearly written report.

The code package should include the following functionality:

1. Forward Kinematics: At a minimum, the code includes a function that takes the joint angles as the **input** and computes the position of each joint and the end effector position as the **output**.
2. Inverse Kinematics: At a minimum, the code includes a function that takes the **end effector position and orientation** as an **input** and computes the joint angles as an **output**. If this is not possible, the function should communicate this to the user somehow (e.g., output NaN as the output while also showing an error message “Desired end effector position and orientation out of workspace of robot.”
3. Visualization: At a minimum, the code should display a stick-figure like drawing of the robot links (“connect the dots”).

The code package should include both the above functions and a script/notebook/etc. code file that demonstrates how the code is used to simulate this robot. It can be simple as the pseudocode below:

```
load robot.parameters;

forward_input = [0 0.1 0.2 0.3 0.4 0.5];

ForKinResult = comp_ForKin(forward_input); % run forward kinematics
computation

display(ForKinResult,robot.parameters); % visualize answer

... more code here
```

The forward and inverse kinematics should be coded by yourself using what you have learned in this course. Otherwise, this project would be too easy! However, the visualization can be done with any existing code package for graphics. This is to save your time so you can focus on what you have learned rather than making the result look nice for visualization purposes.

The accompanying report should include the following elements:

1. A user manual on how to operate the code.
2. Documentation of the major functions including descriptions of the mathematical methods you learned in this course that were implemented in those functions.

There is no standard template for the report, but you can look to existing code documentation (for example, MATLAB documentation https://www.mathworks.com/help/matlab/ref/ode45.html) for inspiration. There is no length requirement either other than including the above requirements.

Extra Credit: Adding these following functionalities to your code package will receive extra credit. You should clearly indicate which extra credit items you are adding to your report to receive full credit. These functions also have to work properly!

- Inverse Dynamics with the Jacobian: Compute the joint torques based on the wrench (force and moment) applied on the end effector (specified by point of application on the blade and components of the wrench)
- Inverse Kinematics with the Jacobian: Compute the joint angles and joint angular velocities for a given set of desired end effector positions and velocities. Also indicate when no solution exists.
- Workspace Computation: Display the workspace of the robot with a reasonable resolution.
- Trajectory Optimization: Generate a motion profile that goes from a given end-effector position or pose to another given end-effector position or pose according to some criteria (minimum time, minimum torque, etc.)
- I will add any other interesting suggestions of yours to this list…