# Homework 4

Graded

4 Days, 23 Hours Late

**Student**

Rohin Siddhartha Palaniappan Venkateswaran

**Total Points**

**62.5 / 70 pts**

**Question 1**

Problem 1 **17.5** / 20 pts

1.1 **Screw Axis 1** **3.33** / 3.33 pts

✔ **+ 3.33 pts** Correct

**+ 0 pts** No report provided

**+ 0 pts** Final screw axes not provided

1.2 **Screw Axis 2** **3.34** / 3.34 pts

✔ **+ 3.34 pts** Correct

**+ 0 pts** No report provided

**+ 0 pts** Final screw axes not provided

1.3 **Screw Axis 3** **3.33** / 3.33 pts

✔ **+ 3.33 pts** Correct

**+ 0 pts** No report provided

**+ 0 pts** Final screw axes not provided

1.4 **Body Jacobian** **5** / 5 pts

✔ **+ 5 pts** Correct

**− 5 pts** No code provided

**+ 0 pts** Silent code

**+ 0 pts** Provided code doesn't work

1.5 **Inverse Kinematics** **2.5** / 5 pts

**+ 5 pts** Correct

**+ 0 pts** No information regarding IK solutions in report and silent code

✔ **+ 2.5 pts** No information about IK solution in report

**+ 0 pts** No code provided

**+ 0 pts** Unable to run the code

**Question 2**

Problem 2 — **35** / 40 pts

2.1 **Screw Axis 1** — **2.85** / 2.85 pts

- ✔ **+ 2.85 pts** Correct
- **+ 0 pts** Blank
- **+ 1.4 pts** Correct - not implemented in Matlab
- **+ 0.5 pts** Frames are not correct. There is no Matlab code.
- **+ 0.5 pts** Missing screw axis wrt body frame. There is no Matlab code
- **+ 0.5 pts** Screw matrix is not correct. Code does not run
- **+ 2 pts** Screw axis correct - Code does not run.

2.2 **Screw Axis 2** — **2.85** / 2.85 pts

- ✔ **+ 2.85 pts** Correct
- **+ 1.4 pts** Correct - not implemented in Matlab
- **+ 0.5 pts** Frames are not correct. There is no Matlab code.
- **+ 0.8 pts** Frames are not correct
- **+ 0.5 pts** Missing screw axis wrt body frame. There is no Matlab code
- **+ 0.5 pts** Screw matrix is not correct. Code does not run
- **+ 0 pts** Blank
- **+ 2 pts** Screw axis correct - Code does not run.

2.3 **Screw Axis 3** — **2.85** / 2.85 pts

- ✔ **+ 2.85 pts** Correct
- **+ 0.5 pts** Frames are not correct. There is no Matlab code.Blank
- **+ 1.4 pts** Correct - not implemented in Matlab
- **+ 0.5 pts** Missing screw axis wrt body frame. There is no Matlab code
- **+ 0.5 pts** Screw matrix is not correct. Code does not run
- **+ 0 pts** Blank
- **+ 1.4 pts** Screw axis correct - Code does not run.

2.4 **Screw Axis 4** **2.85** / 2.85 pts

- ✔ **+ 2.85 pts** Correct
- **+ 0 pts** Blank
- **+ 1.4 pts** Correct - not implemented in Matlab
- **+ 0.5 pts** Missing screw axis wrt body frame. There is no Matlab code
- **+ 0.5 pts** Frames are not correct. There is no Matlab code.
- **+ 0.5 pts** Screw matrix is not correct. Code does not run
- **+ 1.4 pts** Screw axis correct - Code does not run.
- **+ 2 pts** Screw axis correct - Code does not run.

2.5 **Screw Axis 5** **2.85** / 2.85 pts

- ✔ **+ 2.85 pts** Correct
- **+ 0.5 pts** Frames are not correct. There is no Matlab code.
- **+ 0 pts** v5 is not correct
- **+ 0.5 pts** Missing screw axis wrt body frame. There is no Matlab code
- **+ 0.5 pts** Screw matrix is not correct. Code does not run
- **+ 0 pts** Blank
- **+ 2 pts** Screw axis correct - Code does not run.

2.6 **Screw Axis 6** **2.85** / 2.85 pts

- ✔ **+ 2.85 pts** Correct
- **+ 0 pts** Blank
- **+ 1.4 pts** Correct - not implemented in Matlab
- **+ 0.5 pts** Screw matrix is not correct. Code does not run
- **+ 0.5 pts** Missing screw axis wrt body frame. There is no Matlab code
- **+ 2 pts** Screw axis correct - Code does not run.

2.7 **Home Configuration** **2.9** / 2.9 pts

- ✔ **+ 2.9 pts** Correct
- **+ 0.5 pts** Partially correct - not implemented in Matlab
- **+ 1.4 pts** Correct - not implemented in Matlab
- **+ 0 pts** Blank
- **+ 2 pts** Correct - Code does not run.

It is correct in the code

2.8 **Body Jacobian** **10** / 10 pts

- ✔ **+ 10 pts** Correct
- **+ 0 pts** Not implemented in Matlab
- **+ 0 pts** Code does not run

2.9 **Inverse Kinematics** **5** / 10 pts

- **+ 10 pts** Correct
- ✔ **+ 5 pts** Code partially run
- **+ 0 pts** Not implemented
- **+ 0 pts** Code does not run

Robot does not follow the trajectory

Question 3

**Extra Credit** **10** / 10 pts

- ✔ **+ 10 pts** Correct
- **+ 0 pts** No submission
- **+ 0 pts** Incomplete response
- **+ 0 pts** Incorrect response

Worcester Polytechnic Institute

# RBE/ME 501 – ROBOT DYNAMICS
# HOMEWORK 4
# Spring 2023 – Instructor: L. Fichera

## INSTRUCTIONS

- **This homework includes 4 preliminary programming questions and 2 problems (*)**
- **To solve the 2 problems, you will have to have Peter Corke's MATLAB Robotics toolbox installed on your system** (ver 10.4 or newer). See https://petercorke.com/toolboxes/robotics-toolbox/.
    - **Submit your solutions to the programming questions on MATLAB Grader:** https://grader.mathworks.com/courses/96837-rbe-501-robot-dynamics-spring-2023/assignments/284160-homework-4
    - **Submit your solutions to the other problems on Gradescope.**
      **You will find separate entries to submit:**
        - A zip file containing your MATLAB code
            - All source files must be professionally commented; ideally, anyone not familiar with the code should be able to understand what it does just from reading the comments;
            - All functions **must** include a help section;
            - Use I/O and/or figures to illustrate what your code does. Code that runs silently, that is, without generating any visible output, will not be graded;
        - A PDF file detailing your solution; the PDF **must** contain:
            - All necessary schematics/diagrams used to derive the solution;
            - Commentary to explain your thought process (for instance: "*To solve the inverse kinematics, I apply method X*"); if unsure how much level of detail is enough, err on the side of verbosity; no points will be awarded for just reporting the right solution;
            - All calculations (even though they are numerically performed in MATLAB);
    - Check your code before submitting it! **Code that runs with an error will not be graded.**
    - The PDF must be clearly legible. Plots and figures must be of professional quality. Figure DPI >= 300. Remember to add axis labels and units.
- **Soft deadline**: Wednesday 29-Mar-23 at 8:00 pm
- **Hard deadline**: Monday 3-Apr-23 at 6:00 pm

(*) Plus the usual extra credit question.

*The remainder of this page was intentionally left blank.*

**Preliminary Programming Questions (40 points total – 10 points for each correct solution)**

This section must be completed online at:
https://grader.mathworks.com/courses/96837-rbe-501-robot-dynamics-spring-2023/assignments/284160-homework-4

**1. Angular Velocity in the Body Frame**

**2. Twists in the Body Frame**

**3. Product of Exponentials in the Body Frame**

**4. Jacobian in the Body Frame**

**Problem 1 (20 points):**

The robot in Figure 1 consists of a three degrees-of-freedom manipulator, with link lengths as follows: L1: 0.3 m, L2: 0.3 m, L3: 0.3 m[1]. A model of this manipulator was pre-created for your convenience: open the zip archive with the starting MATLAB code provided on Canvas, then run `hw3problem1.m`.

***Note***: *Running this script will first display the robot, then it will result in an error. The error is expected, and it occurs because the forward kinematics of this robot is not implemented yet – we will work on this next.*

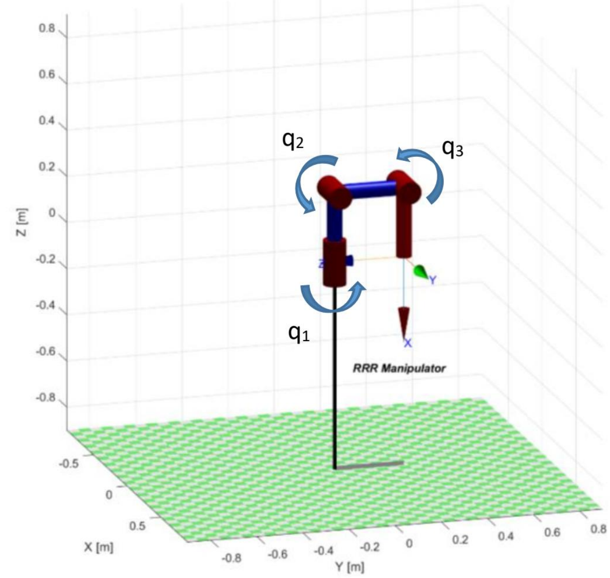



*Figure 1: Three DoF manipulator.*

**a. Calculate the forward kinematics using the product of exponentials in the <u>body</u> frame.**

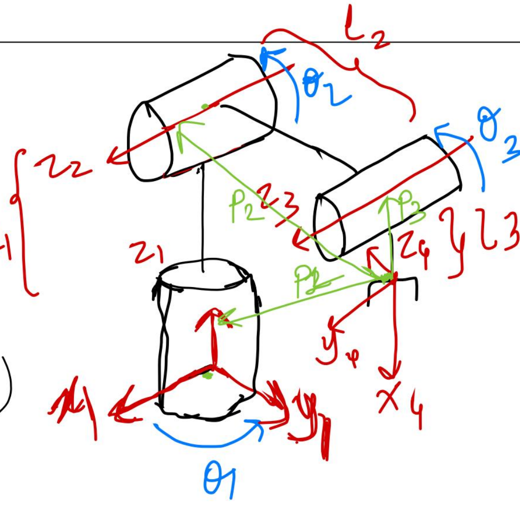
To complete this step, you will first have to calculate each of the screw axes $\boldsymbol{\xi_i} = (\boldsymbol{\omega_i}, \boldsymbol{v_i})$ with respect to the <u>body</u> frame and add these screw axes to a variable called S_body (line 41). Next, you should use the `fkine` function developed in Section 1 of this homework to calculate the forward kinematics (line 65). At this point you can run the script, which will test the correctness of the forward kinematics on 20 configurations generated at random.

$S_1 = \begin{bmatrix} w_1 \\ v_1 \end{bmatrix}$

$w_1 = [-1 \ \ 0 \ \ 0]$

$v_1 = -[w_1] p_1$

$= -\text{cross}([-1 \ 0 \ 0], [L_1 - L_3 \ \ 0 \ \ L_2])$




$S_2 = \begin{bmatrix} w_2 \\ v_2 \end{bmatrix}$

$w_2 = [0 \ 1 \ 0] \quad v_2 = -[w_2] p_2$

$v_2 = -\text{cross}([-1 \ 0 \ 0], [-L_3 \ \ 0 \ \ L_2])$

$\boldsymbol{\xi_1} = [-1 \ \ 0 \ \ 0 \ \ 0 \ \ -0.3 \ \ 0]^T$

$\boldsymbol{\xi_2} = [0 \ \ 1 \ \ 0 \ \ -0.3 \ \ 0 \ \ -0.3]^T$

$\boldsymbol{\xi_3} = [0 \ \ 1 \ \ 0 \ \ 0 \ \ 0 \ \ -0.3]^T$

[1] Yes, this is the same manipulator you saw in Homework 2. Who said you can't teach an old robot new tricks?



cont.d

b. **Calculate the <u>body</u> Jacobian of the manipulator.**
Calculate the body Jacobian of the robot (line 96). You should be able to use the `jacobe` function developed in Section 1 of this homework. Run the script again, which will test the correctness of the Jacobian on 20 configurations generated at random.

c. **Calculate the inverse position kinematics <u>using the Analytical Jacobian.</u>**
Implement the inverse kinematics algorithm starting from line 149. In your report, comment on what numerical method you chose to implement (Jacobian Transpose? Pseudo-Jacobian? Damped Least Squares?); take note of the configurations for which the inverse kinematics algorithm fails (if any), investigate the cause of the failure (ill-conditioned Jacobian? Unreachable configuration?) and attempt to find a different solution for these configurations.

$$S_3 = \begin{bmatrix} w_3 \\ v_3 \end{bmatrix}$$

$$w_3 = [0\ 1\ 0],\quad v_3 = -[w_3]p_3$$

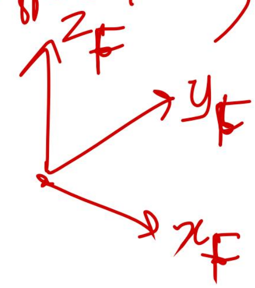
$$v_3 = -\text{cross}([0\ 1\ 0], [-L_3\ 0\ 0])$$

**Problem 2 (40 points):**
Repeat the same steps of problem 2 for the *Stanford Manipulator* (see Fig. 2). Run `hw3problem2.m` to create a model of the robot. For link lengths, refer to the robot model included with the Robotics Toolbox (type `stanf` in the Matlab terminal).

**<u>Note #1</u>**: The figure below is reproduced from (Sciavicco, 2009). This textbook uses the convention of numbering DH frames starting from 0, while joint variables are numbered starting from 1.

**<u>Note #2</u>**: The configuration shown below is <u>not</u> the home configuration coded up in the Matlab Robotics toolbox. To see what the home configuration of the robot is, type stanf.teach(zeros(1,6)) in the Matlab terminal.

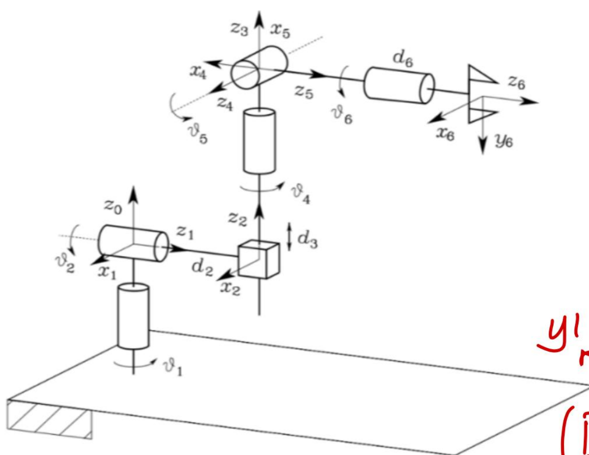
(end effector frame)

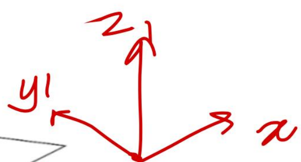



$L_1 = 0.412$

$L_2 = 0.154$

$L_6 = 0.263$




(Base frame)

*Figure 2: Stanford Manipulator*

(Calculation of screw axis in body frame just like previous question).

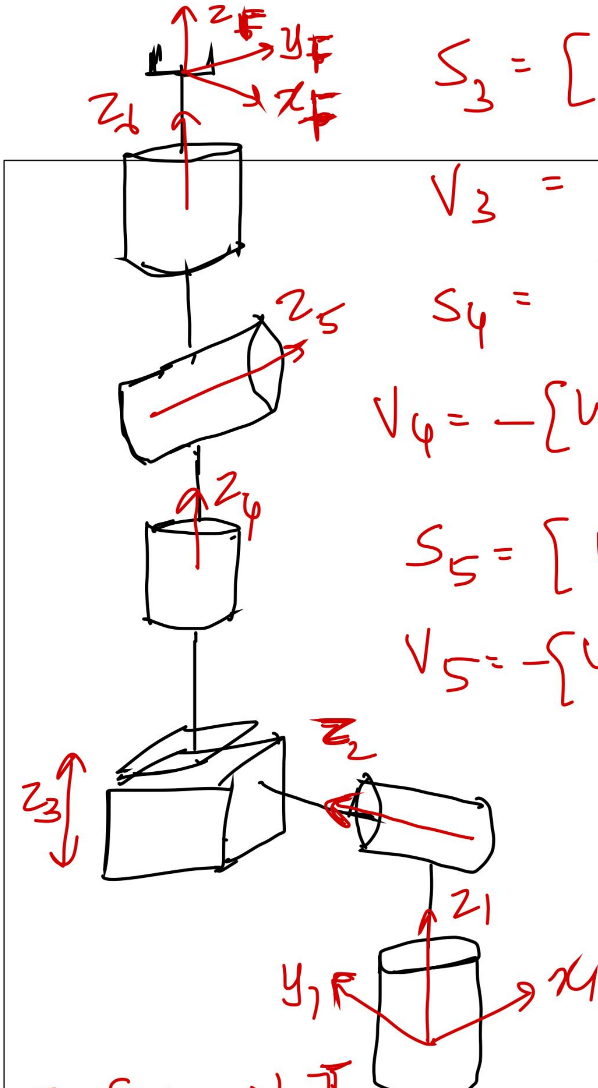
Calculate the screw axes and home configuration matrix here:

$S_1 = [w_1 \ v_1]^T \quad w_1 = [0 \ 0 \ 1]$

$v_1 = -[w_1]p_1 = -\text{cross}([0 \ 0 \ 1],[L_2 \ 0 \ -L_1 - L_6])$

$S_2 = [w_2 \ v_2]^T \quad w_2 = [-1 \ 0 \ 0]$

$v_2 = -[w_2]p_2 = -\text{cross}([-1 \ 0 \ 0][-L_2 \ 0 \ -L_6])$




$S_3 = [W_3 \ \ V_3]^T \quad W_3 = [0 \ 0 \ 0]$

$V_3 = [0 \ 0 \ 1]$

$S_4 = [W_4 \ \ V_4]^T \quad W_4 = [0 \ 0 \ 1]$

$V_4 = -[W_4]P_4 = -\mathrm{cross}([0 \ 0 \ 1][0 \ 0 \ -L_4])$

$S_5 = [W_5 \ \ V_5]^T \quad W_5 = [0 \ 1 \ 0]$

$V_5 = -[W_5]P_5 = -\mathrm{cross}([0 \ 0 \ 1]$
$[0 \ 0 \ -L_4])$

$S_6 = [W_6 \ \ V_6]^T$

$V_6 = -[W_6]P_6, \ W_6 = [0 \ 0 \ 1]$

$= -\mathrm{cross}([0 \ 0 \ 1][0 \ 0 \ -L_4])$

$\xi_1 = [0 \ 0 \ 1 \ \ 0 \ -0.154 \ 0]^T$

$\xi_2 = [-1 \ 0 \ 0 \ \ 0 \ 0.263 \ 0]^T$

$\xi_3 = [0 \ 0 \ 0 \ \ 0 \ 0 \ 1]^T$

$\xi_4 = [0 \ 0 \ 1 \ \ 0 \ 0 \ 0]^T$

$\xi_5 = [0 \ 1 \ 0 \ \ 0.263 \ 0 \ 0]^T$

$\xi_6 = [0 \ 0 \ 1 \ \ 0 \ 0 \ 0]^T$

$R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$p = [0 \ \ L_2 \ \ L_1 + L_4]^T$

$M = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & L_2 \\ 0 & 0 & 1 & L_1 + L_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

**Extra Credit (10 points):**

Let us consider an analytic Jacobian matrix $\boldsymbol{J_a}$ such that $\dot{\boldsymbol{p}} = \boldsymbol{J_a}\,\dot{\boldsymbol{\theta}}$, where $\boldsymbol{p} \in \mathbb{R}^3$ represents the end effector position in Cartesian coordinates, and $\boldsymbol{\theta} \in \mathbb{R}^n$ is a vector of joint coordinates. Prove that the relationship between $\boldsymbol{J_a}$ and the body Jacobian $\boldsymbol{J_b}$ is given by $\boldsymbol{J_a} = \boldsymbol{R}\,\boldsymbol{J_{vb}}$, where:

- $\boldsymbol{R} \in SO(3)$ describes the orientation of the end effector.
- $\boldsymbol{J_{vb}}$ denotes the bottom three rows of the body Jacobian $\boldsymbol{J}_b$.

*Proof*:

The relationship between endeffector velocities in the body frame is given by

$$V_b = J_b \dot{q}$$

Similarly in the space frame,

$$V_a = J_a \dot{q}$$

The relationship between $V_b$ and $V_a$ is

$$V_b = (R)^{-1} V_a$$ where $R$ is the rotation matrix.

$$\Rightarrow \quad V_a = R\, V_b$$



(cont.d)

**Extra Credit (10 points):**

Let us consider an analytic Jacobian matrix $\boldsymbol{J_a}$ such that $\dot{\boldsymbol{p}} = \boldsymbol{J_a}\,\dot{\boldsymbol{\theta}}$, where $\boldsymbol{p} \in \mathbb{R}^3$ represents the end effector position in Cartesian coordinates, and $\boldsymbol{\theta} \in \mathbb{R}^n$ is a vector of joint coordinates. Prove that the relationship between $\boldsymbol{J_a}$ and the body Jacobian $\boldsymbol{J_b}$ is given by $\boldsymbol{J_a} = \boldsymbol{R}\,\boldsymbol{J_{vb}}$, where:

- $\boldsymbol{R} \in SO(3)$ describes the orientation of the end effector.
- $\boldsymbol{J_{vb}}$ denotes the bottom three rows of the body Jacobian $\boldsymbol{J}_b$.

*Proof*:

We know $[J_b] = [W_b]$ (Bottom 3 rows of Body Jacobian $J_b$).

Also, $r \times [W_b] = V_b$ → $r$ is the position vector of end effector in body frame

Take transpose on both sides

$$[V_b]^T = ([r] \times [W_b])^T$$
$$= [W_b]^T [r]^T$$

(comparing)

$$[V_a]^T = [V_b]^T [R]^T$$

Here $[V_a]^T = ([W_b]^T [r]^T) R$

$\boxed{J_a = R\, J_b}$ Hence proved