Introduction to Robotics

Lab 5a

Shaaf Farooque 08405, Mysha Zulfiqar 08443

Task 5.1 Manipulator Jacobian (20 points)

Use the homogeneous transformation, ${}^{0}T_{4}$, obtained in the previous lab to find either the space Jacobian or the body Jacobian[a] for the manipulator in the lab.

[a] Singularity analysis in the next task is easier with the body Jacobian.

```
syms l_1 l_2 l_3 l_4 l_5

M = [1 0 0 0;
     0 1 0 0;
     0 0 1 l_1+l_2+l_3+l_4+l_5;
     0 0 0 1;
    ];
w1 = [0;0;1];
q1 = [0;0;l_1];
S1 = [w1;cross(-w1,q1)];

w2 = [1;0;0];
q2 = [0;0;l_1+l_2];
S2 = [w2;cross(-w2,q2)];

w3 = [1;0;0];
q3 = [0;0;l_1+l_2+l_3];
S3 = [w3;cross(-w3,q3)];

w4 = [1;0;0];
q4 = [0;0;l_1+l_2+l_3+l_4];
S4 = [w4;cross(-w4,q4)];

syms theta_1 theta_2 theta_3 theta_4
T = exp(S1,theta_1)*exp(S2,theta_2)*exp(S3,theta_3)*exp(S4,theta_4)*M;
T = simplify(T)
```

T =

$$\begin{pmatrix} \cos(\theta_1) & -\sigma_1 \sin(\theta_1) & \sigma_3 \sin(\theta_1) & \sin(\theta_1)\,\sigma_2 \\ \sin(\theta_1) & \sigma_1 \cos(\theta_1) & -\sigma_3 \cos(\theta_1) & -\cos(\theta_1)\,\sigma_2 \\ 0 & \sigma_3 & \sigma_1 & l_1 + l_2 + l_4 \cos(\theta_2+\theta_3) + l_3 \cos(\theta_2) + l_5\,\sigma_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = \cos(\theta_2 + \theta_3 + \theta_4)$$

$$\sigma_2 = l_4 \sin(\theta_2 + \theta_3) + l_3 \sin(\theta_2) + l_5\,\sigma_3$$

$$\sigma_3 = \sin(\theta_2 + \theta_3 + \theta_4)$$

```
% Calculate the body Jacobian J_b
J_b = sym(zeros(6, 4));

% First column: S1 (already in body frame)
J_b(:, 1) = S1;

% Second column: Adjoint of exp(S1, theta_1) applied to S2
T1 = exp(S1, theta_1);
Ad_T1 = adjoint_transformation(T1);
J_b(:, 2) = Ad_T1 * S2;

% Third column: Adjoint of exp(S1, theta_1) * exp(S2, theta_2) applied to S3
T2 = exp(S1, theta_1) * exp(S2, theta_2);
Ad_T2 = adjoint_transformation(T2);
J_b(:, 3) = Ad_T2 * S3;

% Fourth column: Adjoint of exp(S1, theta_1) * exp(S2, theta_2) * exp(S3, theta_3) applied to S4
T3 = exp(S1, theta_1) * exp(S2, theta_2) * exp(S3, theta_3);
Ad_T3 = adjoint_transformation(T3);
J_b(:, 4) = Ad_T3 * S4;

J_b = simplify(J_b);
disp('J_b:')
disp(J_b)
```

```
J_b:
```

$$\begin{pmatrix} 0 & \cos(\theta_1) & \cos(\theta_1) & \cos(\theta_1) \\ 0 & \sin(\theta_1) & \sin(\theta_1) & \sin(\theta_1) \\ 1 & 0 & 0 & 0 \\ 0 & -\sin(\theta_1)\ (l_1 + l_2) & -\sin(\theta_1)\,\sigma_2 & -\sin(\theta_1)\,\sigma_1 \\ 0 & \cos(\theta_1)\ (l_1 + l_2) & \cos(\theta_1)\,\sigma_2 & \cos(\theta_1)\,\sigma_1 \\ 0 & 0 & l_3 \sin(\theta_2) & l_4 \sin(\theta_2 + \theta_3) + l_3 \sin(\theta_2) \end{pmatrix}$$

where

$$\sigma_1 = l_1 + l_2 + l_4 \cos(\theta_2 + \theta_3) + l_3 \cos(\theta_2)$$

$$\sigma_2 = l_1 + l_2 + l_3 \cos(\theta_2)$$

Task 5.2 Rank of Jacobian (5 points)

(a) What is the maximum possible rank of this Jacobian?

(b) Will we able to achieve arbitrary linear velocity and arbitrary angular velocity of the end-effector?

(a) Maximum Rank of Jb :

The maximum possible rank of Jb is 4, as the manipulator has 4 joints.

(b) The manipulator can achieve arbitrary linear velocity in 3D space. The manipulator can achieve arbitrary angular velocity in 3D space. However, it cannot simultaneously achieve arbitrary linear and angular velocity in 3D space due to the lack of sufficient independent degrees of freedom (only 4 joints). This means the manipulator can control only 4 independent degrees of freedom at any given time.

Task 5.3 Singular Configurations (20 points)

Determine the conditions at which the rank of Jacobian drops below its maximal rank and the corresponding singular configurations.

- We'll choose the origin of our end-effector frame on the axis of our last joint. If you chose it earlier at the center of the end-effector, you can effectively shift it by setting the last length to be zero.
- You can use the function `subs` to substitute $a_4 = 0$, e.g. if the Jacobian is stored in `J`, then we use `subs(J,'a_4',0)`.
- Remember to make judicious use of `det`, `simplify` and `expand`.

(a) Show that the Jacobian loses rank when $\sin\theta_3 = 0$.

(b) How would you describe the shape of the arm at singular configurations? At singularity, in which direction is the arm unable to move?

(a)

Sin(theta_3) = 0, theta_3 = 0, pi

```
J_b_subs = subs(J_b, l_5, 0);
J_b_subs = simplify(J_b_subs);
J_b_subs = subs(J_b, theta_3, 0);
J_b_subs = simplify(J_b_subs)
```

```
J_b_subs =
```

$$\begin{pmatrix} 0 & \cos(\theta_1) & \cos(\theta_1) & \cos(\theta_1) \\ 0 & \sin(\theta_1) & \sin(\theta_1) & \sin(\theta_1) \\ 1 & 0 & 0 & 0 \\ 0 & -\sin(\theta_1)\ (l_1 + l_2) & -\sin(\theta_1)\,\sigma_2 & -\sin(\theta_1)\,\sigma_1 \\ 0 & \cos(\theta_1)\ (l_1 + l_2) & \cos(\theta_1)\,\sigma_2 & \cos(\theta_1)\,\sigma_1 \\ 0 & 0 & l_3 \sin(\theta_2) & \sin(\theta_2)\ (l_3 + l_4) \end{pmatrix}$$

where

$$\sigma_1 = l_1 + l_2 + l_3 \cos(\theta_2) + l_4 \cos(\theta_2)$$

$$\sigma_2 = l_1 + l_2 + l_3 \cos(\theta_2)$$

Columns 2, 3, and 4 share identical first two rows. Column 4 is a linear combination of Column 3 and Column 2: In the linear velocity components (rows 4–6), Column 4 depends on Column 3 and Column 2.

Columns 3 and 4 become linearly dependent because:

$$\text{Column 4} = \text{Column 3} + \text{Column 2} \cdot k,$$

where $k = \frac{l_4 \cos\theta_2}{l_1 + l_2}$.

This dependency reduces the effective degrees of freedom from 4 to 3.

Similarly, when I substitute theta_3 = pi instead of 0:

```
J_b_subs = subs(J_b, l_5, 0);
J_b_subs = simplify(J_b_subs);
J_b_subs = subs(J_b, theta_3, pi);
J_b_subs = simplify(J_b_subs)
```

J_b_subs =

$$\begin{pmatrix} 0 & \cos(\theta_1) & \cos(\theta_1) & \cos(\theta_1) \\ 0 & \sin(\theta_1) & \sin(\theta_1) & \sin(\theta_1) \\ 1 & 0 & 0 & 0 \\ 0 & -\sin(\theta_1)\ (l_1+l_2) & -\sin(\theta_1)\,\sigma_2 & -\sin(\theta_1)\,\sigma_1 \\ 0 & \cos(\theta_1)\ (l_1+l_2) & \cos(\theta_1)\,\sigma_2 & \cos(\theta_1)\,\sigma_1 \\ 0 & 0 & l_3\sin(\theta_2) & \sin(\theta_2)\ (l_3-l_4) \end{pmatrix}$$

where

$$\sigma_1 = l_1 + l_2 + l_3\cos(\theta_2) - l_4\cos(\theta_2)$$

$$\sigma_2 = l_1 + l_2 + l_3\cos(\theta_2)$$

Again, column 4 is linearly dependent on column 2 and 3.

Task 5.4 Grasp and Pre-grasp poses (10 points)

Determine an appropriate grasp pose, ${}^sT_g$, and a pre-grasp pose, ${}^sT_{pg}$, in terms of a given object pose ${}^sT_o$.

Grasp Pose:

$${}^sT_g = {}^sT_o . {}^oT_g$$

Where,

${}^sT_o$ is the object's transformation in the space frame.

${}^oT_g$ is the transformation from the object frame to the grasp frame

We are considering a top-down grasp i.e. the gripper will be picking the object from the top so keeping that in mind, ${}^oT_g$ is simply pure translation along the z axis.

$$
{}^{o}T_{g} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L + l_g \\ 0 & 0 & 0 & \end{bmatrix}
$$

Where l_g can be understood as the depth with respect to the camera frame at which the object is present.

Pre gasp Pose:

The pre grasp pose is slightly away from the grasp pose.

$$
{}^{s}\,T_{pg} = {}^{s}\;T_{o}.{}^{o}\;T_{pg}
$$

Where,

${}^{o}\,T_{pg}$ is the transformation from the object to the pre grasp frame.

$$
{}^{o}T_{g} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L + l_{\text{pg}} \\ 0 & 0 & 0 & \end{bmatrix}
$$

l_{pg} can be any value and we have decided to keep it in the range of 5-10 cm.

Task 5.5 Desired end-effector velocity (10 points)

Formulate an expression to determine the desired end-effector velocity, ${}^{s}v_{e}$, for straight-line motion, given the current position of the origin of our end-effector frame in the fixed frame, ${}^{s}p_{fk}$, as determined by our forward kinematics mapping and the desired position, ${}^{s}p_{pg}$, as determined by ${}^{s}T_{pg}$.

Provide a MATLAB `function ve = makeVE(p_cur,p_des,speed)` that accepts the current position, the desired position, and the linear speed as arguments and returns the desired end-effector velocity.

The velocity should be in the direction of the difference between the desired position and the current position scaled by speed. For the direction, we are going to use the unit vector of the difference between the two position vectors.

$$^{s}v_e = v_{\text{speed}} \cdot \frac{^{s}p_{\text{pg}} - {^{s}p_{\text{fk}}}}{\| {^{s}p_{\text{pg}}} - {^{s}p_{\text{fk}}} \|}$$

Where,

v_{speed} is the linear speed

$^{s}p_{pg} - {^{s}p_{fk}}$ is the displacement vector from the current position to the desired position

And the denominator is th

e Euclidean distance to normalize and give the direction vector.

```
function ve = makeVE(p_cur, p_des, speed)
% Compute the displacement vector
direction = p_des - p_cur;
% Compute the Euclidean distance
distance = norm(direction);
% Avoid division by zero (if current and desired positions are identical)
if distance == 0
ve = [0; 0; 0]; % No movement required
return;
end
% Normalize the direction and scale by speed
ve = (speed / distance) * direction;
end

% Usage Example
p_cur = [1; 2; 3]; % Current position
p_des = [4; 6; 3]; % Desired position
speed = 0.5; % Linear speed

ve = makeVE(p_cur, p_des, speed)
```

```
ve = 3×1

    0.3000
    0.4000
         0
```