Homework 2: Motion

24-760 Robot Dynamics & Analysis
Fall 2023

Name: SRECHARAN SELVAM

Note: For homework submission, please submit the PDF of the written portion to "Homework 2" and a zipped folder of your Matlab code to "Homework 2 Programming" in Gradescope.

Problem 1) Lunar Motion

Consider a simplified model of the motion of the earth and the moon. Attach a stationary coordinate frame to the center of the earth (s), a frame with the same origin that rotates with the earth (e), and a frame to the moon (m). The axes of rotation are all aligned with each other and pointing in the $+z$ direction of each frame. Assume the moon's orbit around the earth is circular with radius l_m. The earth's radius is r_e and the moon's radius is r_m. The moon rotates about the earth at a rate of 1 revolution per 28 days, and about it's own axis at a rate of 1 revolution per 28 days. The earth rotates about it's own axis at a rate of 1 revolution per day. *Hint: Draw a figure to keep track of the different frames.*

1.1) Just consider the earth's rotation to start. At time t, assume the earth is rotated so that the earth's $+x$ axis is aligned with the stationary $-y$. What is R_{se}? What is g_{se}? Use this configuration for the other parts of this question.

1.2) For a point q on the surface of the earth, $q_e = [0, r_e, 0]^T$, calculate the location of this point in the stationary frame using a rigid body transformation.

1.3) What is the body velocity of the earth's rotation, V_{se}^b? What is the spatial velocity V_{se}^s?

1.4) Using that body and spatial velocity, what is the instantaneous velocity of the point q_e in the earth's frame, v_{qe}? What is the velocity in the stationary frame, v_{qs}?

1.5) Now consider the position of the moon relative to the earth. Assume at time t that the moon is located at $[l_m, 0, 0]^T$ in the stationary frame, with the moon's x axis pointing to the earth. Calculate g_{sm}, then calculate g_{em} based on g_{sm} and g_{se}.

1.6) What is the body velocity of the moon's motion, V_{sm}^b?

1.7) Calculate the spatial velocity, V_{sm}^s, using an adjoint operation. What is special about v_{sm}^s?

1.8) Based on the rotation of the earth and the orbit of the moon, how long is a lunar day on earth? That is, from the earth's perspective, how long does the moon take to come back over the same spot on the surface of the earth?

1. LUNAR MOTION

1.1 Given that;

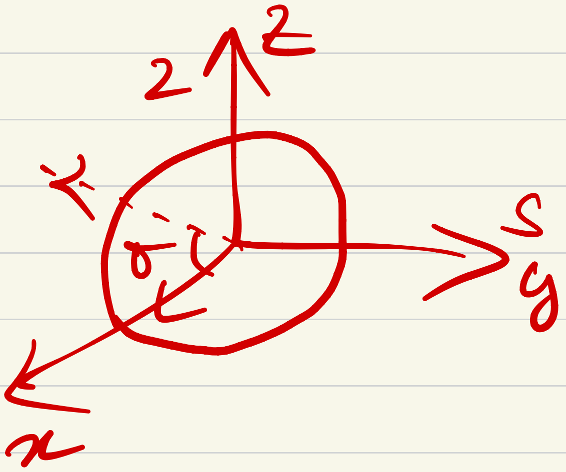
e = Rotates with the earth.

S = stationary.

m = Frame to moon.

⇒ In this configuration; Frame e is aligned with frame s, but it is rotated by an angle θ around the z-axis of frame s, since the earth's rotation axis is the z-axis of frame s.

⇒ The rotation angle 'θ' can be calculated as $\theta = -\omega t$, where ω is the angular velocity of the earth's rotation and 't' is the time.



Now Rotation matrix R_{se} is;

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

However, the frame is rotated by $\theta = 90^\circ$,

$$\boxed{\therefore R_{SE} = \begin{bmatrix} 0 & +1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}_{3\times3}}$$

Now, For g_{se}, which represents the acceleration due to gravity in Frame 'e'. Gravity points towards the center of the Earth and is represented as $(0, 0, -g)$.

However, the origins are same for both frames.

$$\therefore g = \left[\begin{array}{c:c} R_{se} & 0 \\ 0\ 0\ 0 & 1 \end{array}\right]$$

$$\boxed{\therefore \overline{g_{se}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times4}}$$

1.2 Given pt; $q_e = \begin{bmatrix} 0 \\ r_c \\ 0 \end{bmatrix}$

we can use a rigid Body transformation.

Now, for $\bar{q}_s$ ↳ $\bar{g}_{se}\,\bar{q}_e$

Now,

$$\underset{4\times4}{\bar{q}_s = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}} \underset{4\times1}{\begin{bmatrix} 0 \\ r_e \\ 0 \\ 1 \end{bmatrix}} = \underset{4\times1}{\begin{bmatrix} r_e \\ 0 \\ 0 \\ 1 \end{bmatrix}}$$

Now $\boxed{q_s = \begin{bmatrix} r_e \\ 0 \\ 0 \end{bmatrix}}$ } In 'S'-coordinate frame

1.3 To calculate the body velocity of the earth's rotation (V^b_{se}) and the spatial velocity (V^s_{se}) of a point on the earth's surface due to its rotation, we will consider the rotation of the earth.

$$\bar{V}^b_{se} = \bar{g}^{-1}_{se} \times \bar{g}_{se}$$

$$\bar{V}^b_{se} = \begin{bmatrix} R^{-1} & P^{-1} \\ 000 & 1 \end{bmatrix} \begin{bmatrix} R & P \\ 000 & 0 \end{bmatrix} = \begin{bmatrix} R^{-1}R & P^{-1}P \\ 000 & 0 \end{bmatrix}$$

$$\boxed{\bar{V}^b_{se} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}_{3\times 3}}$$

1.4 Instantaneous Velocity of point q_e in the earth's Frame (V_{q_e}):

$$V_{q_e} = \omega \times r_e$$

And, $V_{q_s} \Rightarrow$ Velocity of point q_e in the stationary Frame (V_{q_s})

Now, $V_{q_s} = \hat{V}^s_{se} \times q_s$

$\bar{V}_{qs}$

$$\Rightarrow \begin{bmatrix} 0 & -\theta & 0 & 0 \\ \theta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}_{4\times4} \begin{bmatrix} re \\ 0 \\ 0 \\ 1 \end{bmatrix}_{4\times1} = \begin{bmatrix} 0 \\ \theta c = 0 \\ 0 \\ 0 \end{bmatrix}_{4\times1}$$

$$\boxed{\bar{V}_{qs} = \begin{bmatrix} 0 \\ 2\pi re \\ 0 \\ 0 \end{bmatrix}}$$

Now, $V_{qe} = \bar{V}^{b}_{se}\, qe \Rightarrow \begin{bmatrix} 0 & -\theta & 0 & 0 \\ \theta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}_{4\times4} \begin{bmatrix} 0 \\ re \\ 0 \\ 1 \end{bmatrix}_{4\times1}$

$$\boxed{V_{qe} = \begin{bmatrix} -2\pi re \\ 0 \\ 0 \\ 0 \end{bmatrix}_{4\times1}}$$

1.5

$g_{sm} = [0, -g\_moon, 0] \rightarrow$ acceleration due to gravity on moon's surface.

Gravitational acceleration of the earths relative to the moon in the stationary frame, based on g_{sm} & g_{se}

The origin of the moon is only translated along x-axis only.

$$\therefore \bar{P}_{sm} = \begin{bmatrix} l_m \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

when $\theta = 180^\circ$, (z-axis); $R_{sm} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Now, $\begin{bmatrix} R_{sm} & P_{sm} \\ 0\;0\;0 & 1 \end{bmatrix} \Rightarrow g_{sm}$

$$\Downarrow$$

$$\boxed{\begin{bmatrix} -1 & 0 & 0 & l_m \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times 4} = g_{sm}}$$

Now, for $g_{em} \Rightarrow g_{es} \times g_{sm}$

$$\begin{bmatrix} R^T & \vdots & -P \\ 0\,0\,0 & \vdots & 1 \end{bmatrix} \Leftarrow g_{se}^{-1} \times g_{sm}$$

$$\Rightarrow \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times 4} \begin{bmatrix} -1 & 0 & 0 & l_m \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times 4}$$

$$\therefore\ g_{sm} = \boxed{\begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & l_m \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times4}}$$

1.6 Finding V^b_{sm}, we need to use

$$V^b_{sm} = g_{sm}^{-1} \times \dot{g}_{sm}$$

$$\Rightarrow \begin{bmatrix} R^{-1}_{sm} & -p \\ 000 & 1 \end{bmatrix}_{4\times4} \begin{bmatrix} \dot{R}_{sm} & \dot{p} \\ 000 & 0 \end{bmatrix}_{4\times4} = \begin{bmatrix} R^{-1}_{sm}\dot{R}_{sm} & -p.\dot{p} \\ 000 & 1 \end{bmatrix}$$

Here, $\dot{p} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -l_m \sin\theta.\dot{\theta} \\ l_m \cos\theta.\dot{\theta} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -l_m\pi/14 \\ 0 \end{bmatrix}$

Now, $R^{-1}_{sm} \times \dot{R}_{sm} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}_{3\times3} \begin{bmatrix} 0 & \dot{\theta} & 0 \\ \dot{\theta} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{3\times3}$

$$\Rightarrow \begin{bmatrix} 0 & -\theta & 0 \\ \theta & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{3\times3}$$

Now, $-R^{1}_{sm} \cdot p$

$$\Rightarrow \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -l_m \pi/14 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ l_m \pi/14 \\ 0 \end{bmatrix}$$

Now,

$$[V^b_{sm}] = \begin{bmatrix} 0 & \overset{\pi/14}{-\theta} & 0 & 0 \\ \underset{\pi/14}{\theta} & 0 & 0 & l_m \pi/14 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}_{4\times 4}$$

$$\boxed{V^b_{sm} = \begin{bmatrix} v^b_{sm} \\ \omega_{sm} \end{bmatrix} \Rightarrow \begin{bmatrix} 0 \\ l_m \pi/14 \\ 0 \\ 0 \\ 0 \\ -\pi/14 \end{bmatrix}_{6\times 1}}$$

1.7 Calculate the moon's body velocity in its own rotating frame

$$V^s_{sm} = adj(g_{sm}) V^b_{sm}$$

$$adj(g_{sm}) = \begin{bmatrix} R_{sm} & \hat{P}R_{sm} \\ 000 & R_{sm} \end{bmatrix}$$

where, $R_{sm} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$; $\hat{P} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -l_m \\ 0 & l_m & 0 \end{bmatrix}$

$$\hat{P}R_{sm} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -l_m \\ 0 & l_m & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

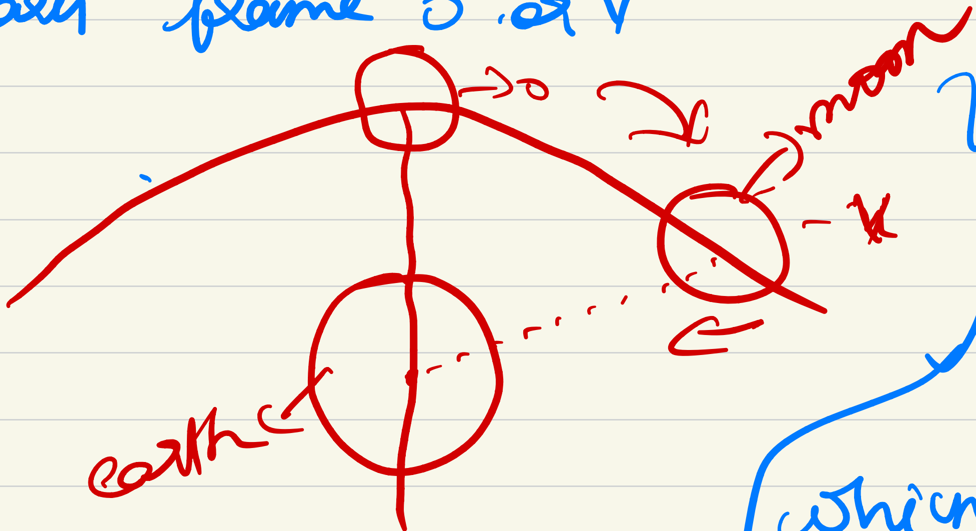
$$\hat{P}.R_{sm} \Rightarrow \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -l_m \\ 0 & -l_m & 0 \end{bmatrix}$$

$$adj(g_{sm}) = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l_m \\ 0 & 0 & 1 & 0 & -l_m & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{6\times6}$$

Now,

$$V^{s}_{sm} = \underset{6\times 6}{adj(g_{sm})} \times \underset{6\times 1}{\begin{bmatrix} 0 \\ l_n \pi/14 \\ 0 \\ 0 \\ 0 \\ -\pi/14 \end{bmatrix}} \Rightarrow \underset{6\times 1}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -\pi/14 \end{bmatrix}}$$

Now, the speciality of V^{s}_{sm} is that the velocity of translation of moon w.r.t stationary frame 0, or V



There will be some side of the moon which always faces the earth's surface.

1.8 A lunar day, is the time it takes for the moon to return to same position in the sky relative to the earth & the sun. This is approx <u>29.5 days</u>.

→ Length of a lunar day is longer than the moon's orbital period around the earth, which is about 27.3 days. This difference is due to combined motion of the earth and the moon in their orbits around the sun.

→ So only considering rotation vectors

$$\omega_{rc} - \omega_{sun} = \begin{bmatrix} 0 \\ 0 \\ \frac{27\pi}{14} \end{bmatrix}$$

→ Since the angle & earth will be 2π,

$$\Rightarrow \text{Time} = \frac{\text{angle}}{\text{angular velocity}} = \frac{2\pi}{\frac{27\pi}{14}} = \underline{1.036}$$

$$\boxed{\therefore \text{Lunar Day} = 1.036 \text{ days}}$$

2.1)

```
% Combined script for angvel2skew, skew2angvel, and unit test

% angvel2skew.m

function W_hat = angvel2skew(w)

   % Create a 3x3 skew-symmetric matrix from a 3-element vector w

   W_hat = [0, -w(3), w(2);

           w(3), 0, -w(1);

          -w(2), w(1), 0];

end

% skew2angvel.m

function w = skew2angvel(W_hat)

   % Extract the angular velocity vector w from a 3x3 skew-symmetric matrix W_hat

   w = [W_hat(3,2); W_hat(1,3); W_hat(2,1)];

end

% Unit test for angvel2skew and skew2angvel functions

% Generate a random angular velocity vector w

w = rand(3, 1);

% Call angvel2skew to convert w to a skew-symmetric matrix W_hat

W_hat = angvel2skew(w);

% Call skew2angvel to convert the skew-symmetric matrix W_hat back to an angular velocity vector w_prime

w_prime = skew2angvel(W_hat);

% Check if w and w_prime are approximately equal (within a tolerance)

tolerance = 1e-6;

is_equal = all(abs(w - w_prime) < tolerance);

if is_equal

   disp('Test is passed as angvel2skew and skew2angvel are inverse to each other.');

else

   disp('Test is failed as angvel2skew and skew2angvel are not inverse to each other.');

end
```

2.2)

```
% Combined script for twist2rbvel, rbvel2twist, and unit test

% twist2rbvel.m

function V_hat = twist2rbvel(V)

   % Create a 4x4 rigid body velocity matrix in homogeneous coordinates from a 6-element twist vector V

   V_hat = [skewSymmetricMatrix(V(1:3)), V(4:6);

            zeros(1, 4)];

end

function W_hat = skewSymmetricMatrix(w)

   % Create a 3x3 skew-symmetric matrix from a 3-element vector w

   W_hat = [0, -w(3), w(2);

            w(3), 0, -w(1);

           -w(2), w(1), 0];

end

% rbvel2twist.m

function V = rbvel2twist(V_hat)

   % Extract the 6-element twist vector V from a 4x4 rigid body velocity matrix in homogeneous coordinates V_hat

   V = [V_hat(1:3, 4); unskewSymmetricMatrix(V_hat(1:3, 1:3))];

end

function w = unskewSymmetricMatrix(W_hat)

   % Extract a 3-element vector w from a 3x3 skew-symmetric matrix W_hat

   w = [W_hat(3, 2); W_hat(1, 3); W_hat(2, 1)];

end

% Unit test for twist2rbvel and rbvel2twist functions

% Generate a random twist vector V

V = rand(6, 1);

% Call twist2rbvel to convert V to a 4x4 rigid body velocity matrix in homogeneous coordinates V_hat

V_hat = twist2rbvel(V);
```

```
% Call rbvel2twist to convert the rigid body velocity matrix V_hat back to a twist vector V_prime

V_prime = rbvel2twist(V_hat);

% Check if V and V_prime are approximately equal (within a tolerance)

tolerance = 1e-6;

is_equal = all(abs(V - V_prime) < tolerance);

if is_equal

    disp('Test is passed as twist2rbvel and rbvel2twist are inverse of each other.');

else

    disp('Test is failed as twist2rbvel and rbvel2twist are not inverse of each other.');

end
```

2.3)

```
% tform2adjoint.m

function Adg = tform2adjoint(g)

   % Extract the rotation matrix R and translation vector p from the homogeneous transformation matrix g

   R = g(1:3, 1:3);

   p = g(1:3, 4);

   % Create the 6x6 adjoint transformation matrix Adg

   Adg = zeros(6, 6);

   % Populate the upper-left 3x3 block with R

   Adg(1:3, 1:3) = R;

   % Populate the lower-right 3x3 block with R

   Adg(4:6, 4:6) = R;

   % Compute the 3x3 skew-symmetric matrix p_hat from the translation vector p

   p_hat = [0, -p(3), p(2);

            p(3), 0, -p(1);

           -p(2), p(1), 0];

   % Populate the upper-right 3x3 block with p_hat

   Adg(1:3, 4:6) = p_hat;

end
```

2.4)

```
% compare_twist.m

function compare_twist()

  % Generate a random 4x4 homogeneous transformation matrix g

  g = random_homogeneous_matrix();

  % Compute the spatial velocity Vs_hat and body velocity Vb_hat

  Vs_hat = compute_spatial_velocity(g);

  Vb_hat = compute_body_velocity(g);

  % Convert the spatial velocity Vs_hat to a spatial twist Vs

  Vs = rbvel2twist(Vs_hat);

  % Convert the body velocity Vb_hat to a body twist Vb

  Vb = rbvel2twist(Vb_hat);

  % Display the generated transformation matrix g

  disp('Generated Transformation Matrix g:');

  disp(g);

  % Display the spatial and body twists

  disp('Spatial Twist Vs:');

  disp(Vs);

  disp('Body Twist Vb:');

  disp(Vb);

  % Check if the spatial twist and body twist are approximately equal (within a tolerance)

  tolerance = 1e-6;

  is_equal = all(abs(Vs - Vb) < tolerance);

  if is_equal

    disp('Test is passed as Spatial Twist and Body Twist are equal.');

  else

    disp('Test is Failed as Spatial Twist and Body Twist are not equal.');

  end

end
```

```
function g = random_homogeneous_matrix()
    % Generate a random 4x4 homogeneous transformation matrix
    R = random_rotation_matrix();
    p = rand(3, 1);
    g = eye(4);
    g(1:3, 1:3) = R;
    g(1:3, 4) = p;
end
function R = random_rotation_matrix()
    % Generate a random 3x3 rotation matrix
    theta = rand() * 2 * pi;
    v = rand(3, 1);
    v = v / norm(v);
    K = [0, -v(3), v(2);
         v(3), 0, -v(1);
        -v(2), v(1), 0];
    R = eye(3) + sin(theta) * K + (1 - cos(theta)) * K^2;
end
function Vs_hat = compute_spatial_velocity(g)
    % Compute the spatial velocity Vs_hat from the transformation matrix g
    Vs_hat = eye(4);
    Vs_hat(1:3, 4) = g(1:3, 4);
end
function Vb_hat = compute_body_velocity(g)
    % Compute the body velocity Vb_hat from the transformation matrix g
    Vb_hat = g;
    Vb_hat(1:3, 4) = [0; 0; 0];
end
```

2.5)

```
% compare_twist.m

function [g, Vs_Adg, Vs, Vb] = compare_twist()

    % Generate a random 4x4 homogeneous transformation matrix g

    g = random_homogeneous_matrix();

    % Compute the spatial velocity Vs_hat and body velocity Vb_hat

    Vs_hat = compute_spatial_velocity(g);

    Vb_hat = compute_body_velocity(g);

    % Convert the spatial velocity Vs_hat to a spatial twist Vs

    Vs = rbvel2twist(Vs_hat);

    % Convert the body velocity Vb_hat to a body twist Vb

    Vb = rbvel2twist(Vb_hat);

    % Compute the adjoint transformation matrix Adg

    Adg = tform2adjoint(g);

    % Compute Vs_Adg by applying the adjoint transformation

    Vs_Adg = Vs * Adg;

    % Display the generated transformation matrix g

    disp('Generated Transformation Matrix g:');

    disp(g);

    % Display Vs, Vs_Adg, and Vb

    disp('Spatial Twist Vs:');

    disp(Vs);

    disp('Vs_Adg (Spatial Twist after Adjoint Transformation):');

    disp(Vs_Adg);

    disp('Body Twist Vb:');

    disp(Vb);

    % Check if Vs and Vs_Adg are approximately equal (within a tolerance)

    tolerance = 1e-6;

    is_equal = all(abs(Vs - Vs_Adg) < tolerance);
```

```
    if is_equal
        disp('Test is passed as Vs and Vs_Adg are identical.');
    else
        disp('Test is failed as Vs and Vs_Adg are not identical.');
    end
end
% Rest of the functions (random_homogeneous_matrix, random_rotation_matrix,
% compute_spatial_velocity, compute_body_velocity, tform2adjoint) remain the same as in the previous answer.
```