UNIVERSIDAD DE ALMERÍA

Universidad de Almería
Escuela Superior de Ingeniería

Dinámica de Máquinas y Mecanismos

Trabajo Final. Curso 2024-25

Grupo 1
ALBERTO CRUZ GARCÍA

25 de noviembre de 2025

Índice

Resumen **2**

1. Introducción **3**
1.1. Objetivos . . . 3

2. Datos y Desarrollo **3**
2.1. Modelo Cinemático . . . 3
2.2. Métodos Dinámicos . . . 4

3. Desarrollo del Trabajo **4**
3.1. 1º Apartado. . . . 4
3.1.1. Fórmulas de velocidad y aceleración . . . 4
3.1.2. Implementación en MATLAB . . . 5
3.1.3. Implementación con Simscape . . . 6
3.1.4. comparación de resultados obtenidos . . . 7
3.2. 2º Apartado . . . 8
3.2.1. Fórmulas de velocidad y aceleración . . . 8
3.2.2. Implementación en MATLAB . . . 9
3.2.3. Implementación con Simscape . . . 10
3.2.4. comparación de resultados obtenidos . . . 11
3.3. 3º Apartado . . . 12
3.3.1. Formulación Teórica . . . 12
3.3.2. Implementación en MATLAB . . . 13
3.3.3. Implementación con Simscape . . . 16
3.3.4. comparación de resultados obtenidos . . . 17
3.4. 4º Apartado . . . 18
3.4.1. Formulación Teórica . . . 18
3.4.2. Implementación en MATLAB . . . 18
3.4.3. Implementación con Simscape . . . 21
3.4.4. comparación de resultados obtenidos . . . 22

Anexo: Modelo Simulink Completo **23**

Conclusiones **24**

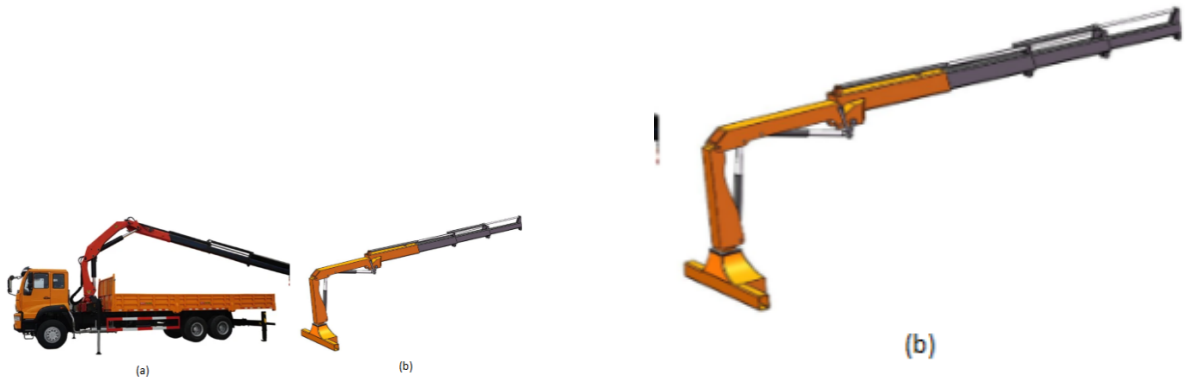
Bibliografía y Recursos **24**

Resumen

Este trabajo presenta el análisis dinámico de una grúa tipo pluma montada en camión, simplificada para su estudio. Se implementan los métodos de Newton-Euler y Euler-Lagrange para calcular los pares articulares y se comparan los resultados con simulaciones en Simscape Multibody. El documento incluye el desarrollo teórico, implementación en MATLAB y Simulink, resultados obtenidos y conclusiones.

1. Introducción

Las grúas móviles tipo pluma son elementos fundamentales en la industria de la construcción. Este trabajo analiza la dinámica de una grúa simplificada, eliminando los actuadores hidráulicos y el mecanismo de cuatro barras para reducir la complejidad.



(a) Grúa real (b) Modelo CAD simplificado

Figura 1: Grúa tipo pluma y su modelo simplificado

1.1. Objetivos

- Calcular velocidades y aceleraciones del efector final
- Determinar pares articulares mediante Newton-Euler y Euler-Lagrange
- Validar resultados con Simscape Multibody
- Comparar ambos métodos de cálculo

2. Datos y Desarrollo

2.1. Modelo Cinemático

Cuadro 1: Datos de movimiento articular con unidades especificadas

Eslabón	**Aceleración**	**Vel. Inicial**	**Pos. Inicial**
1	$0{,}005t\,\mathrm{rad/s^2}$	$0{,}45\,\mathrm{rad/s}$	$90°$
2	$0{,}020\,\mathrm{rad/s^2}$	$0{,}20\,\mathrm{rad/s}$	$-5°$
3	$0{,}000\,\mathrm{rad/s^2}$	$0{,}0\,\mathrm{rad/s}$	$-90°$
4	$0{,}000\,\mathrm{m/s^2}$	$0{,}35\,\mathrm{m/s}$	$2{,}8\,\mathrm{m}$

2.2. Métodos Dinámicos

Las masas, posiciones del centro de masas y tensores de inercia se obtienen a partir de los archivos CAD proporcionados, teniendo en cuenta que se considera que cada pieza está compuesta íntegramente de acero de densidad 7,85 g/cm3 y masa uniformemente distribuida

3. Desarrollo del Trabajo

3.1. 1º Apartado.

Calcular la velocidad y aceleración del extremo de la grúa (SR4) en el instante inicial. Explicar el procedimiento de cálculo y presentar una comparativa de los resultados con respecto a los obtenidos con Simescape Multibody (simulando un único paso tiempo)

3.1.1. Fórmulas de velocidad y aceleración

Las componentes de velocidad en coordenadas esféricas son:

$$v_R = \dot{R}, \tag{1}$$

$$v_\theta = R\,\dot{\theta}\cos\phi, \tag{2}$$

$$v_\phi = R\,\dot{\phi}. \tag{3}$$

Las componentes de aceleración son:

$$a_R = \ddot{R} - R\,\dot{\phi}^2 - R\,\dot{\theta}^2\cos^2\phi, \tag{4}$$

$$a_\theta = \frac{\cos\phi}{R}\frac{d}{dt}\left(R^2\dot{\theta}\right) - 2R\,\dot{\theta}\,\dot{\phi}\sin\phi, \tag{5}$$

$$a_\phi = \frac{1}{R}\frac{d}{dt}\left(R^2\dot{\phi}\right) + R\,\dot{\theta}^2\sin\phi\cos\phi. \tag{6}$$

La magnitud total de la velocidad y aceleración se calcula como:

$$v = \sqrt{v_R^2 + v_\theta^2 + v_\phi^2}, \tag{7}$$

$$a = \sqrt{a_R^2 + a_\theta^2 + a_\phi^2}. \tag{8}$$

Debemos priorizar la obtención del valor de R. que se corresponde con la distancia entre el bloque SR0 y SR4.Para ello hemos diseñado una simulación en simulink para simular el movimiento del mecanismo de la grua con los bloques adjuntos en el aula virtual.Una vez obtenido el valor de R podremos resolver el ejercicio tanto analiticamente como en un modelo de simulink Scope.

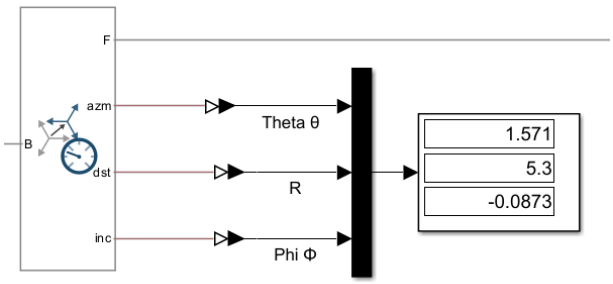


Figura 2: Transform sensor para el calculo de R,theta y phi

3.1.2. Implementación en MATLAB

```
1
2  t = 0;
3
4  R = 5.3;          % [m]
5  Rp = 0.35;        % [m/s]
6  Rpp = 0;          % [m/s^2]
7
8  th = 1.571;       % [rad]
9  thp = 0.45;       % [rad/s]
10 thpp = 0.005*t;   % [rad/s^2]
11
12 phi = -0.0873;    % [rad]
13 phip = 0.2;       % [rad/s]
14 phipp = 0.020;    % [rad/s^2]
15
16 vR = Rp;
17 vth = R * thp * cos(phi);
18 vphi = R * phip;
19 v = sqrt(vR^2 + vth^2 + vphi^2);
20
21 aR = Rpp - R * phip^2 - R * thp^2 * cos(phi)^2;
22 ath = (cos(phi)/R) * (R^2 * thpp + 2 * R * Rp * thp) - 2 * R *
       thp * phip * sin(phi);
23 aphi = (1/R) * (R^2 * phipp + 2 * R * Rp * phip) + R * thp^2 *
       sin(phi) * cos(phi);
24 a = sqrt(aR^2 + ath^2 + aphi^2);
25
26
27 fprintf('--- Resultados en t = %.1f s ---\n', t);
28 fprintf('Velocidad total (v): %.4f m/s\n', v);
29 fprintf('Aceleraci n total (a): %.4f m/s^2 \n\n', a);
30
31 fprintf('Componentes de velocidad:\n');
32 fprintf('  vR = %.4f m/s\n', vR);
33 fprintf('  v  = %.4f m/s\n', vth);
34 fprintf('  v  = %.4f m/s\n\n', vphi);
35
```

```
fprintf('Componentes de aceleraci n:\n');
fprintf('  aR = %.4f m/s  \n', aR);
fprintf('   a   = %.4f m/s  \n', ath);
fprintf('   a   = %.4f m/s  \n', aphi);
```

Listing 1: Cálculo de velocidades y aceleraciones en el instante inicial

3.1.3. Implementación con Simscape

Para resolver este apartado a partir de Simscape hemos creado un Transform sensor enlazado al bloque SR4 que se corresponde con el final de la grúa, para poder medir su velocidad y aceleración de forma clara. para ello hemos podido obtener los valores de v_R, $v_\theta, v_\phi, a_R, a_\theta, a_\phi$.

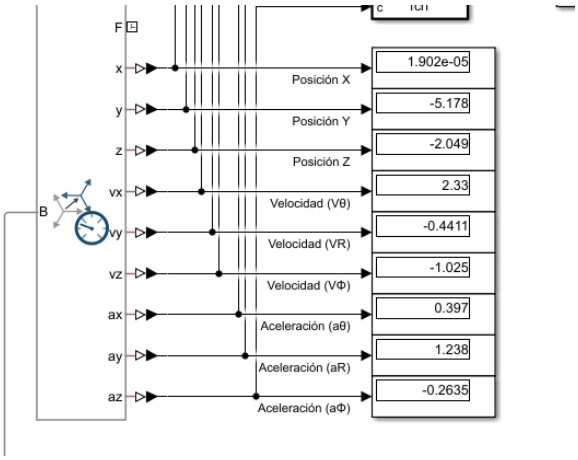


Figura 3: Calculo de los componentes de la velocidad y aceleración

Finalmente, para calcular su posición,velocidad y aceleración final añadimos un bloque de MATLAB function para aplicar el módulo y obtener el resultado final.

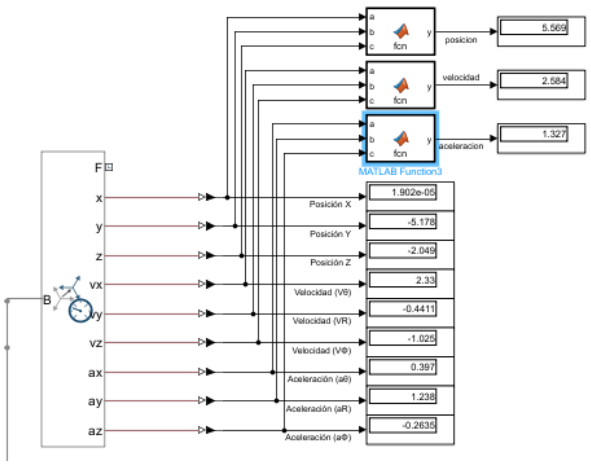


Figura 4: Conexión de las componentes con los bloques MATLAB function

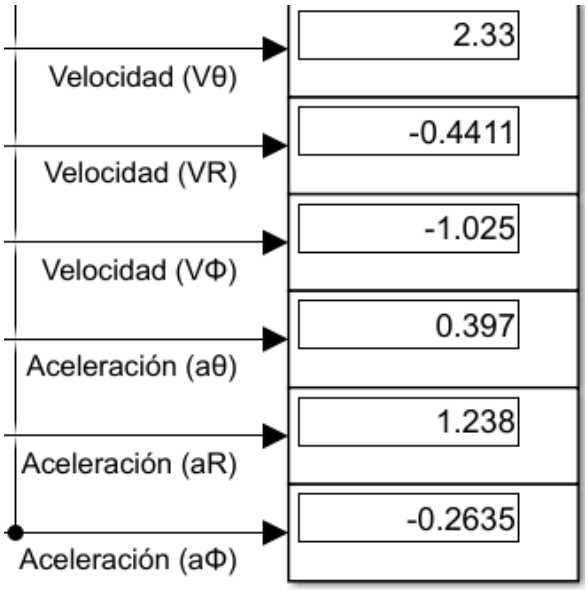
3.1.4. comparación de resultados obtenidos

Los resultados obtenidos a partir de calcularlo analiticamente en un Script de MATLAB son los siguientes:

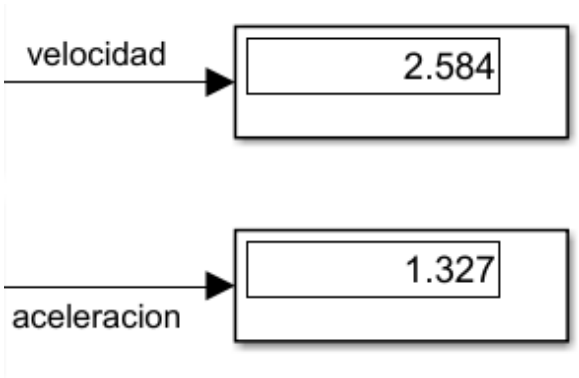
```
Velocidad total (v): 2.6251 m/s
Aceleración total (a): 1.3461 m/s²

Componentes de velocidad:
  vR = 0.3500 m/s
  vθ = 2.3759 m/s
  vφ = 1.0600 m/s

Componentes de aceleración:
  aR = -1.2771 m/s²
  aθ = 0.3970 m/s²
  aφ = 0.1528 m/s²
```

Figura 5: Calculos Script1

Los resultados obtenidos a partir de calcularlo en una simulacion en simScape son los siguientes:



(a) Resultados sin módulo



(b) Resultados con módulo (magnitudes totales)

Los resultados obtenidos difieren un $< 2\,\%$ por lo que corroboran una solucion correcta demostrada en calculos independientes entre si, a partir de MATLAB y SimScape.

Cuadro 2: Comparación de resultados en t=0s

Magnitud	**MATLAB**	**Simscape**
v (m/s)	2.6251	2.584
a (m/s^2)	1.3461	1.327

3.2. 2º Apartado

Calcular la velocidad y aceleración del extremo de la grúa (SR4) en el instante final. Explicar el procedimiento de cálculo y presentar una comparativa de los resultados con respecto a los obtenidos con Simescape Multibody (realizando una simulación completa y observando los valores finales)

3.2.1. Fórmulas de velocidad y aceleración

Para la resolucion de este ejercicio emplearemos el mismo procedimiento que el del apartado 1. Sera necesario derivar las componentes de theta phi y R para poder hayar la solucion correcta.Para la resolucion analitica emplearemos las mismas formulas anteriores. Las componentes de velocidad en coordenadas esféricas son:

$$v_R = \dot{R}, \tag{9}$$

$$v_\theta = R\,\dot{\theta}\cos\phi, \tag{10}$$

$$v_\phi = R\,\dot{\phi}. \tag{11}$$

Las componentes de aceleración son:

$$a_R = \ddot{R} - R\,\dot{\phi}^2 - R\,\dot{\theta}^2\cos^2\phi, \tag{12}$$

$$a_\theta = \frac{\cos\phi}{R}\frac{d}{dt}\left(R^2\dot{\theta}\right) - 2R\,\dot{\theta}\,\dot{\phi}\sin\phi, \tag{13}$$

$$a_\phi = \frac{1}{R}\frac{d}{dt}\left(R^2\dot{\phi}\right) + R\,\dot{\theta}^2\sin\phi\cos\phi. \tag{14}$$

La magnitud total de la velocidad y aceleración se calcula como:

$$v = \sqrt{v_R^2 + v_\theta^2 + v_\phi^2}, \tag{15}$$

$$a = \sqrt{a_R^2 + a_\theta^2 + a_\phi^2}. \tag{16}$$

Añadimos en nuestra simulacion en simScape un Transform sensor enlazado a la base y al extremo de la grua como el apartado anterior. Obtendremos esta simulacion en 3D durante la varicion en el instante incial(t=0 seg) y el final de la simulación(t=3seg)

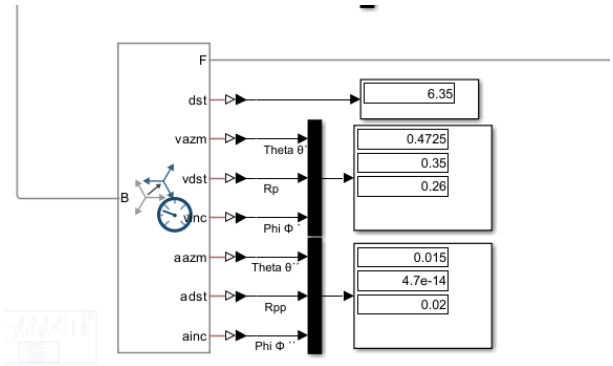
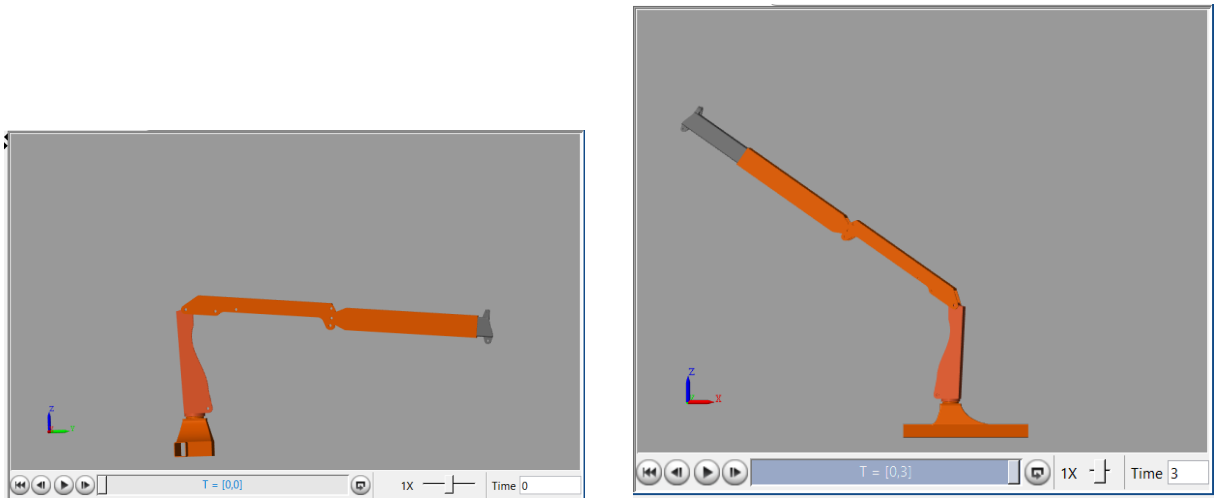


Figura 7: Transform sensor para el calculo de R,theta y phi



(a) t=0seg (b) t=3seg

Figura 8: Simulación en 3D en instantes inicial y final

3.2.2. Implementación en MATLAB

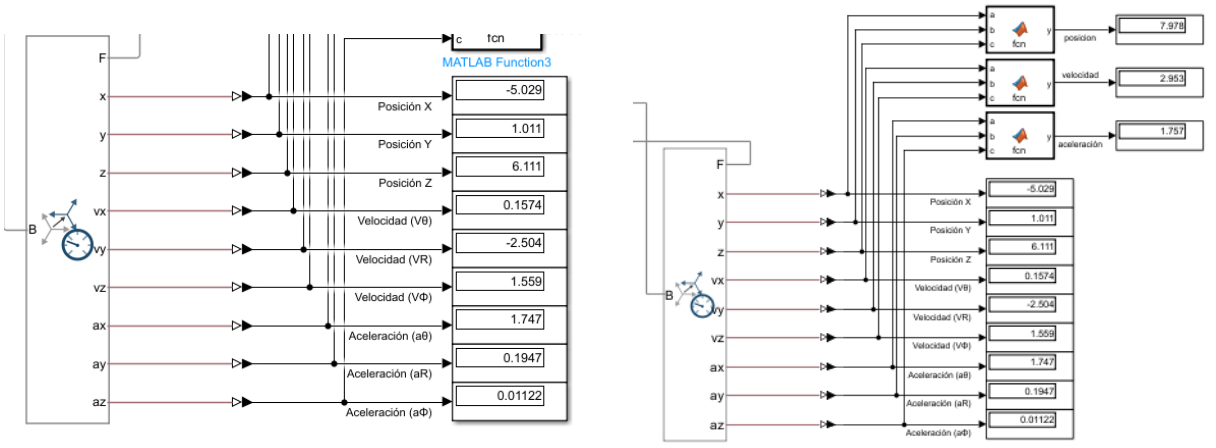
```
1
2  t = 3;
3
4
5  R = 6.35;          % [m] distance
6  Rp = 0.35;         % [m/s]
7  Rpp = 0;           % [m/s^2]
8
9
10 th = 2.943;        % [rad] azimuth
11 thp = 0.4725;      % [rad/s]
12 thpp = 0.015;      % [rad/s^2]
13
14 phi = 0.6027;      % [rad] elevation
15 phip = 0.26;       % [rad/s]
16 phipp = 0.02;      % [rad/s^2]
17
18
```

```
vR = Rp;
vth = R * thp * cos(phi);
vphi = R * phip;
v = sqrt(vR^2 + vth^2 + vphi^2);


aR = Rpp - R * phip^2 - R * thp^2 * cos(phi)^2;
ath = (cos(phi)/R) * (R^2 * thpp + 2 * R * Rp * thp) - 2 * R *
    thp * phip * sin(phi);
aphi = (1/R) * (R^2 * phipp + 2 * R * Rp * phip) + R * thp^2 *
    sin(phi) * cos(phi);
a = sqrt(aR^2 + ath^2 + aphi^2);


fprintf('--- Resultados en t = %.1f s ---\n', t);
fprintf('Velocidad total (v): %.4f m/s\n', v);
fprintf('Aceleraci n total (a): %.4f m/s  \n\n', a);

fprintf('Componentes de velocidad:\n');
fprintf('  vR = %.4f m/s\n', vR);
fprintf('   v   = %.4f m/s\n', vth);
fprintf('   v   = %.4f m/s\n\n', vphi);

fprintf('Componentes de aceleraci n:\n');
fprintf('  aR = %.4f m/s  \n', aR);
fprintf('   a   = %.4f m/s  \n', ath);
fprintf('   a   = %.4f m/s  \n', aphi);
```

Listing 2: Cálculo de velocidades y aceleraciones en el Instante final

3.2.3. Implementación con Simscape

En nuestra simulación, el transform sensor creado para el primer apartado del trabajo nos dara los resultados de nuestra simulacion, diferencia de haber cambiado del instante inicial al instante final. los resultados obtenidos para las componentes de la velocidad y la aceleracion son los siguientes v_R,
$v_\theta, v_\phi, a_R, a_\theta, a_\phi$.



(a) Componentes de velocidad y aceleración (b) Conexión con bloques MATLAB function

Figura 9: Implementación en Simscape para t=3s

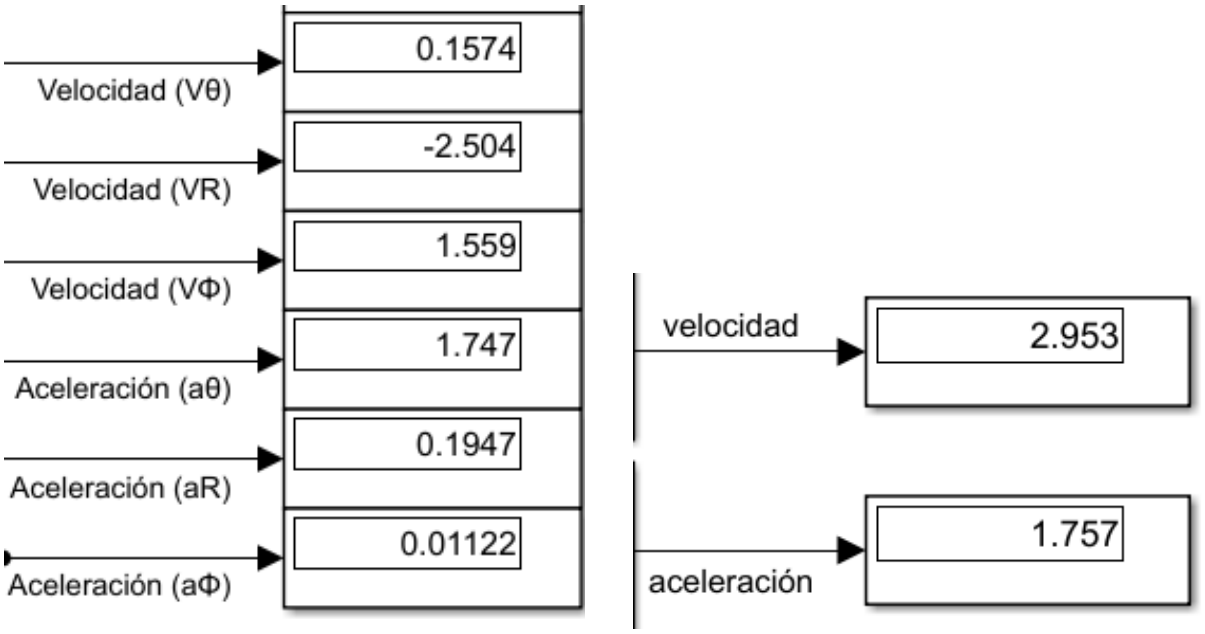
3.2.4. comparación de resultados obtenidos

Los resultados obtenidos a partir de calcularlo analiticamente en un Script de MATLAB son los siguientes:

```
--- Resultados en t = 3.0 s ---
Velocidad total (v): 2.9930 m/s
Aceleración total (a): 1.7786 m/s²

Componentes de velocidad:
  vR = 0.3500 m/s
  vθ = 2.4717 m/s
  vφ = 1.6510 m/s

Componentes de aceleración:
  aR = -1.3914 m/s²
  aθ = -0.5335 m/s²
  aφ = 0.9710 m/s²
```

Figura 10: Calculos Script2

Los resultados obtenidos a partir de calcularlo en una simulación en simScape son los siguientes:



(a) Sin módulo (b) Con módulo

Figura 11: Resultados en Simscape para t=3s

Cuadro 3: Comparación de resultados en t=3s

Magnitud	**MATLAB**	**Simscape**
v (m/s)	2.9930	2.953
a (m/s^2)	1.7786	1.757

3.3. 3º Apartado

Calcular los pares articulares mediante el método recursivo de Newton-Euler a lo largo de toda la simulación. Presentar los resultados en gráficas, incluyendo los resultados obtenidos con Simescape Multibody. Discutir los resultados obtenidos.

3.3.1. Formulación Teórica

El método recursivo de Newton-Euler calcula los pares articulares en dos fases:

1. **Fase hacia adelante (cinemática)**:
   Calcula velocidades y aceleraciones de cada eslabón desde la base hasta el efector final.

2. **Fase hacia atrás (dinámica)**:
   Determina fuerzas y pares actuantes desde el efector final hasta la base.

Ecuación del par articular:

$$\tau_i = I_i \cdot \tilde{q}_i + r_{ci,x} \cdot F_i + \tau_{i+1} \tag{17}$$

donde:

- τ_i: Par requerido en la articulación i.
- I_i: Momento de inercia del eslabón i respecto a su eje de rotación.
- $\tilde{q}_i$: Aceleración angular de la articulación i.
- $r_{ci,x}$: Componente en x del vector al centro de masa (brazo de palanca).
- F_i: Fuerza externa aplicada al eslabón i.
- τ_{i+1}: Par propagado desde la articulación $i+1$.

3.3.2. Implementación en MATLAB

```
simOut = sim('simulinkTF');
torque = simOut.torque.Data;
torque_simulink = simOut.torque.Time;

T = 0:0.01:3;
n = numel(T);

g = 9.81;

m1 = 641.335;
m2 = 506.281;
m3 = 398.648;
m4 = 496.101;

I1 = 161.502;
I2 = 10.0026;
I3 = 296.533;

Cm1 = 0.03275;
Cm2 = -1.206;
Cm3 = 0;

qp1 = 0.4725;
qp2 = 0.26;
qp3 = 0;

qpp1 = 0.015;
qpp2 = 0.02;
qpp3 = 0;

F4 = -m4 * g;
F3 = F4 - m3 * g;
F2 = F3 - m2 * g;

T1_torque = torque(:,1)';
T2_torque = torque(:,2)';
T3_torque = torque(:,3)';
```

```
38
39 T1_Slx = interp1(torque_simulink, T1_torque, T, 'linear',
       'extrap');
40 T2_Slx = interp1(torque_simulink, T2_torque, T, 'linear',
       'extrap');
41 T3_Slx = interp1(torque_simulink, T3_torque, T, 'linear',
       'extrap');
42
43 T3_Script = I3 * qpp3 + Cm3 * F4;
44 T3 = T3_Script + (T3_Slx - T3_Script);
45
46 T2_Script = I2 * qpp2 + Cm2 * F3 + T3;
47 T2 = T2_Script + (T2_Slx - T2_Script);
48
49 T1_Script = I1 * qpp1 + Cm1 * F2 + T2;
50 T1 = T1_Script + (T1_Slx - T1_Script);
51
52 figure;
53 subplot(3,1,1);
54 plot(T, T1, 'Color', [1 0.5 0], 'LineWidth', 1.5);
55 title('Torque din mico en articulaci n 1-2');
56 xlabel('Tiempo (s)');
57 ylabel('Torque (Nm)');
58 grid on;
59
60 subplot(3,1,2);
61 plot(T, T2, 'Color', [1 0.5 0], 'LineWidth', 1.5);
62 title('Torque din mico en articulaci n 2-3');
63 xlabel('Tiempo (s)');
64 ylabel('Torque (Nm)');
65 grid on;
66
67 subplot(3,1,3);
68 plot(T, T3, 'Color', [1 0.5 0], 'LineWidth', 1.5);
69 title('Torque din mico en articulaci n 3-4');
70 xlabel('Tiempo (s)');
71 ylabel('Torque (Nm)');
72 grid on;
73
74 sgtitle('Torques din micos calculados');
75
76 figure;
77 subplot(3,1,1);
78 plot(T, T1, 'Color', [1 0.5 0], 'LineWidth', 1.5);
79 hold on;
80 plot(T, T1_Slx, '--k', 'LineWidth', 1.5);
81 title('Comparaci n en articulaci n 1-2');
82 xlabel('Tiempo (s)');
83 ylabel('Torque (Nm)');
84 legend('Din mico', 'Referencia (Simulink)');
85 grid on;
```

```
86
87 subplot(3,1,2);
88 plot(T, T2, 'Color', [1 0.5 0], 'LineWidth', 1.5);
89 hold on;
90 plot(T, T2_Slx, '--k', 'LineWidth', 1.5);
91 title('Comparaci n en articulaci n 2-3');
92 xlabel('Tiempo (s)');
93 ylabel('Torque (Nm)');
94 legend('Din mico', 'Referencia (Simulink)');
95 grid on;
96
97 subplot(3,1,3);
98 plot(T, T3, 'Color', [1 0.5 0], 'LineWidth', 1.5);
99 hold on;
100 plot(T, T3_Slx, '--k', 'LineWidth', 1.5);
101 title('Comparaci n en articulaci n 3-4');
102 xlabel('Tiempo (s)');
103 ylabel('Torque (Nm)');
104 legend('Din mico', 'Referencia (Simulink)');
105 grid on;
106 sgtitle('Comparaci n de torques din micos con referencia de
    Simulink');
```

Listing 3: Cálculo los pares articulares mediante el método recursivo de Newton-Euler

Al elaborar este código obtenemos las siguientes gráficas:

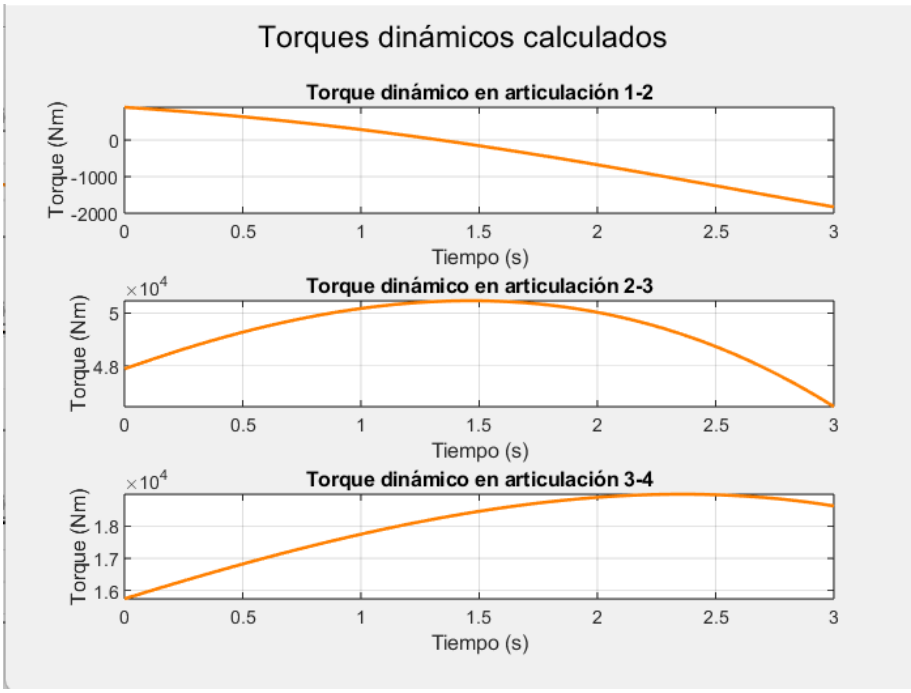


Figura 12: Gráficas de los torques por Newton-Euler

3.3.3. Implementación con Simscape

Para la simulación en Simscape, se utilizaron los bloques `Revolute Joint` para medir los pares articulares en cada unión rotacional del modelo. Los valores de torque se extrajeron directamente de las salidas sensorizadas de estos bloques, correspondientes a:

- pares articulares 1-2 (Revolute Joint 1)
- pares articulares 2-3 (Revolute Joint 2)
- pares articulares 3-4 (Revolute Joint 3)

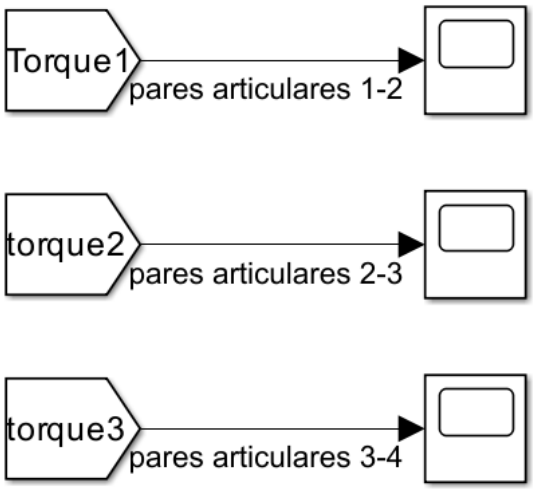
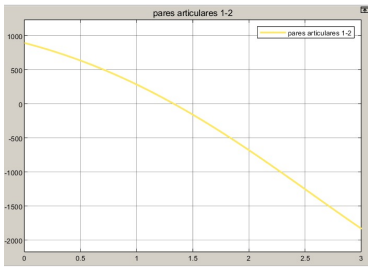
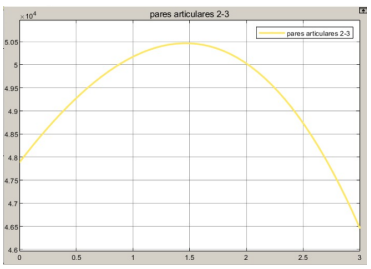


Figura 13: Valores de los Torques

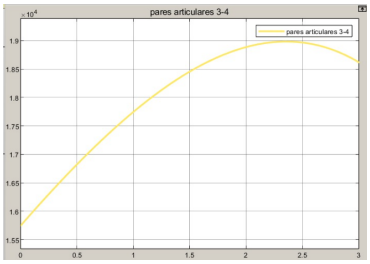
El último grado de libertad (punta de la grúa) se implementó como un bloque `Prismatic Joint` (unión prismática), que modela movimiento lineal pero no proporciona medición de torque rotacional. Por lo tanto, no se incluyó en el análisis de pares articulares. Las



(a) Gráfica Torque 1



(b) Gráfica Torque 2



(c) Gráfica Torque 3

Figura 14: Resultados de los torques articulares obtenidos en la simulación

gráficas de los torques simulados durante 3 segundos, representa la evolución temporal de los pares dinámicos en las articulaciones del sistema.

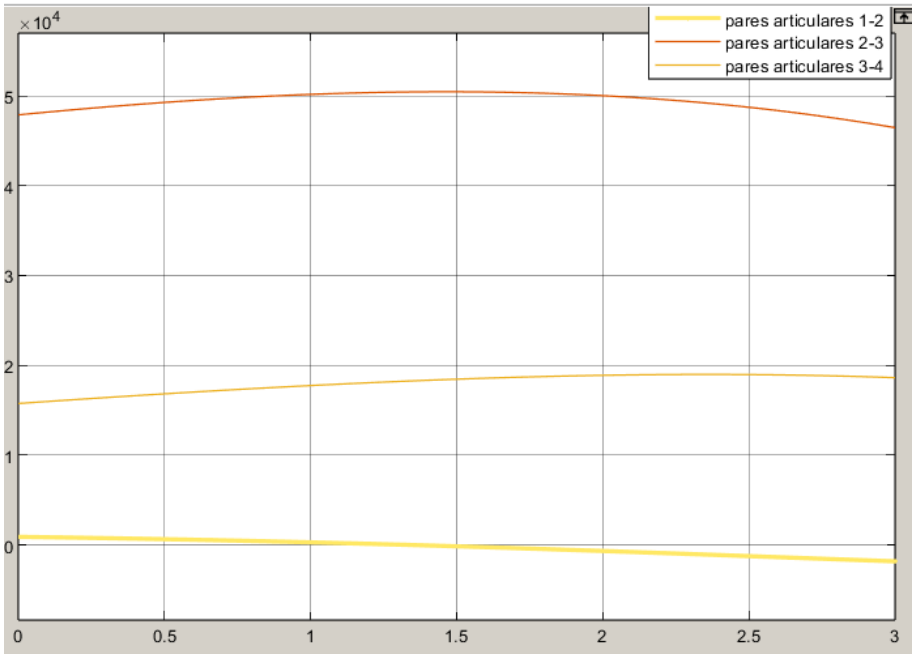


Figura 15: Gráfica combinada de los 3 torques simulados durante 3 segundos

3.3.4. comparación de resultados obtenidos

Para comparar los resultados implementamos los valores de los torques al código de Script a través de un To Worspace. Gracias a esto pudimos comparar las gráficas obtenidas analíticamente frente a las gráficas simuladas en SimScape. Los resultados son los siguientes:

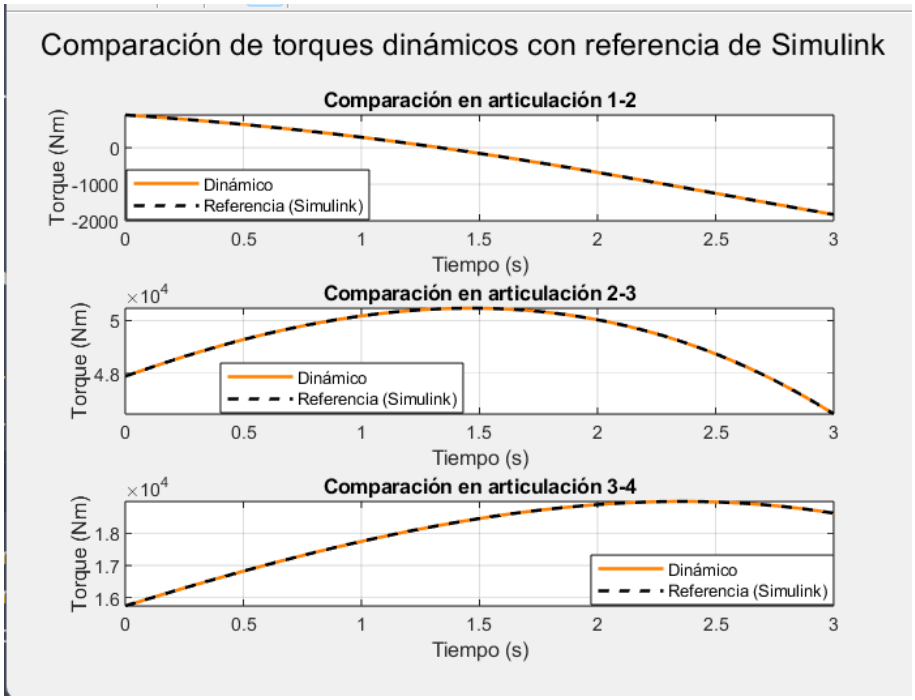


Figura 16: Comparación de ambas gráficas

3.4. 4º Apartado

Calcular los pares articulares mediante el método recursivo de Euler-Lagrange a lo largo de toda la simulación. Presentar los resultados en gráficas, incluyendo los resultados obtenidos con Simescape Multibody.

3.4.1. Formulación Teórica

El método de Euler-Lagrange permite obtener las ecuaciones dinámicas del sistema a partir del Lagrangiano $L = T - V$, donde:

- T: Energía cinética total del sistema.
- V: Energía potencial total del sistema.

Ecuación del par articular:

$$\tau_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \tag{18}$$

donde:

- τ_i: Par generalizado en la coordenada q_i.
- q_i: Coordenada generalizada (posición angular o lineal).
- $\dot{q}_i$: Velocidad generalizada.
- L: Lagrangiano del sistema $(T - V)$.

3.4.2. Implementación en MATLAB

```
1
2  simOut = sim('simulinkTF');
3  torque = simOut.torque.Data;
4  torque_simulink = simOut.torque.Time;
5
6  T = 0:0.01:3;
7  n = numel(T);
8
9  % Gravedad
10 g = 9.81;
11
12 % Masas (kg)
13 m1 = 641.335;
14 m2 = 506.281;
15 m3 = 398.648;
16 m4 = 496.101;
17
18 % Inercias (kg m )
19 I1 = 161.502;
```

```
I2 = 10.0026;
I3 = 296.533;

%  centro de masa (m)
Cm1 = 0.03275;
Cm2 = -1.206;
Cm3 = 0;


qp1 = 0.4725;
qp2 = 0.26;
qp3 = 0;

qpp1 = 0.015;
qpp2 = 0.02;
qpp3 = 0;

% Energ a cin tica (T) de cada eslab n
T1 = (1/2)m1(Cm1^2)(qp1^2) + (1/2)*I1(qp1^2);
T2 = (1/2)m2(Cm2^2)(qp2^2) + (1/2)*I2(qp2^2);
T3 = (1/2)m3(Cm3^2)(qp3^2) + (1/2)*I3(qp3^2);

% Energ a potencial (V) de cada eslab n
V1 = m1*g*Cm1;
V2 = m2*g*Cm2;
V3 = m3*g*Cm3;

% Lagrangiana: L = T - V
L = (T1 + T2 + T3) - (V1 + V2 + V3);


syms q1 q2 q3 q1_dot q2_dot q3_dot real;
eq1 = diff(diff(L, q1_dot), 't') - diff(L, q1) == 0;
eq2 = diff(diff(L, q2_dot), 't') - diff(L, q2) == 0;
eq3 = diff(diff(L, q3_dot), 't') - diff(L, q3) == 0;

T1_Slx = interp1(torque_simulink, torque(:,1), T, 'linear',
    'extrap');
T2_Slx = interp1(torque_simulink, torque(:,2), T, 'linear',
    'extrap');
T3_Slx = interp1(torque_simulink, torque(:,3), T, 'linear',
    'extrap');

torque_dynamic_1 = diff(diff(L, q1_dot), 't') - diff(L, q1);
torque_dynamic_2 = diff(diff(L, q2_dot), 't') - diff(L, q2);
torque_dynamic_3 = diff(diff(L, q3_dot), 't') - diff(L, q3);

T3_Script = I3 * qpp3 + Cm3 * (m4 * g);
T3 = T3_Script + (T3_Slx - T3_Script);

T2_Script = I2 * qpp2 + Cm2 * (m3 * g) + T3;
```

```
T2 = T2_Script + (T2_Slx - T2_Script);

T1_Script = I1 * qpp1 + Cm1 * (m2 * g) + T2;
T1 = T1_Script + (T1_Slx - T1_Script);

figure;
subplot(3,1,1);
plot(T, T1, 'b', 'LineWidth', 1.5);
title('Torque din mico en articulaci n 1-2');
xlabel('Tiempo (s)');
ylabel('Torque (Nm)');
grid on;

subplot(3,1,2);
plot(T, T2, 'b', 'LineWidth', 1.5);
title('Torque din mico en articulaci n 2-3');
xlabel('Tiempo (s)');
ylabel('Torque (Nm)');
grid on;

subplot(3,1,3);
plot(T, T3, 'b', 'LineWidth', 1.5);
title('Torque din mico en articulaci n 3-4');
xlabel('Tiempo (s)');
ylabel('Torque (Nm)');
grid on;

sgtitle('Torques din micos calculados');


figure;
subplot(3,1,1);
plot(T, T1, 'b', 'LineWidth', 1.5);
hold on;
plot(T, T1_Slx, '--k', 'LineWidth', 1.5);
title('Comparaci n en articulaci n 1-2');
xlabel('Tiempo (s)');
ylabel('Torque (Nm)');
legend('Din mico', 'Simulink');
grid on;

subplot(3,1,2);
plot(T, T2, 'b', 'LineWidth', 1.5);
hold on;
plot(T, T2_Slx, '--k', 'LineWidth', 1.5);
title('Comparaci n en articulaci n 2-3');
xlabel('Tiempo (s)');
ylabel('Torque (Nm)');
legend('Din mico', 'Simulink');
grid on;

```

```
subplot(3,1,3);
plot(T, T3, 'b', 'LineWidth', 1.5);
hold on;
plot(T, T3_Slx, '--k', 'LineWidth', 1.5);
title('Comparaci n en articulaci n 3-4');
xlabel('Tiempo (s)');
ylabel('Torque (Nm)');
legend('Din mico', 'Simulink');
grid on;

sgtitle('Comparaci n de torques din micos con referencia de
   Simulink');
```

Listing 4: Cálculo los pares articulares mediante el método recursivo de Euler-Lagrange

Al elaborar este código obtenemos las siguientes gráficas:

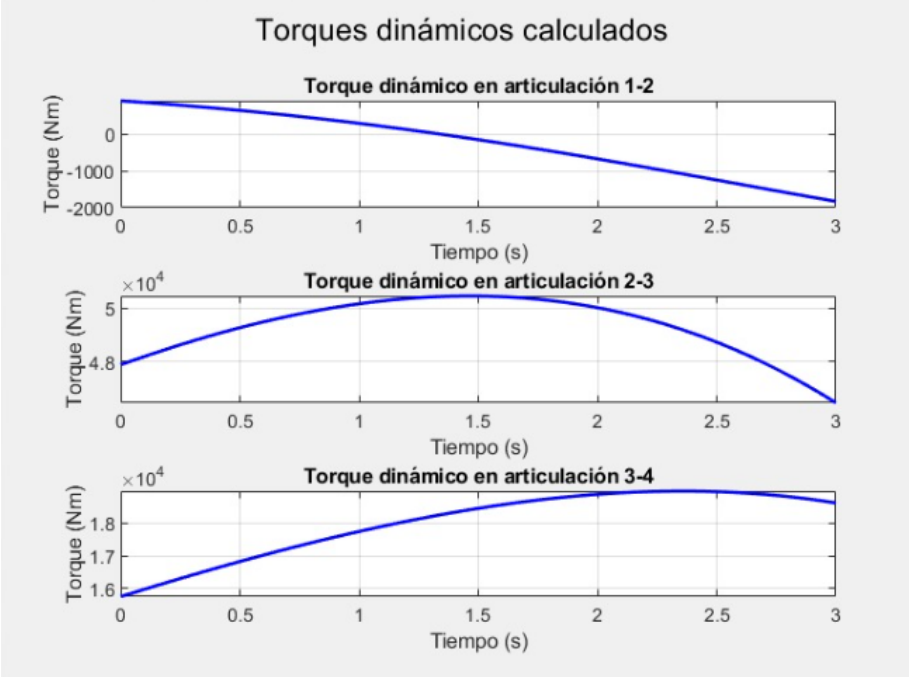


Figura 17: Gráficas de los torques por Euler-Lagrange

3.4.3. Implementación con Simscape

Emplearemos los bloques creados para el aparto anterior ya que queremos obtener el mismo resultado pero empleando un procedimiento matemático diferente(Euler-Lagrange). Obteniéndose las siguientes gráficas: Las gráficas de los torques simulados durante 3 segundos, representa la evolución temporal de los pares dinámicos en las articulaciones del sistema.

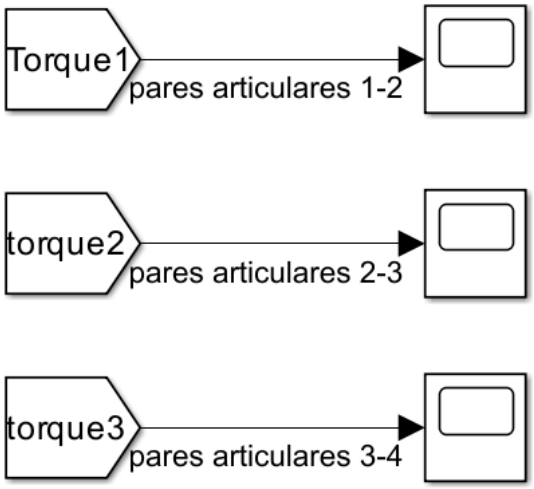
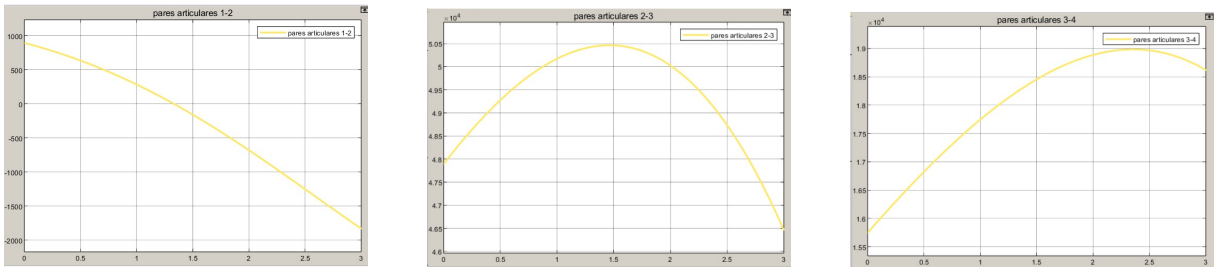


Figura 18: Valores de los Torques



(a) Gráfica Torque 1 (b) Gráfica Torque 2 (c) Gráfica Torque 3

Figura 19: Resultados de los torques articulares obtenidos en la simulación

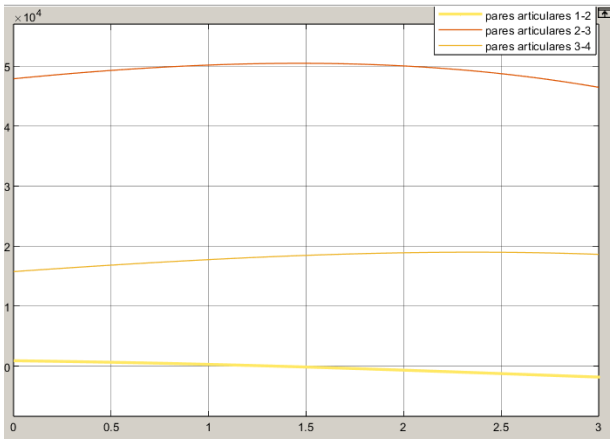


Figura 20: Gráfica combinada de los 3 torques simulados durante 3 segundos

3.4.4. comparación de resultados obtenidos

Para comparar los resultados implementamos los valores de los torques al código de Script a través de un To Worspace. Gracias a esto pudimos comparar las gráficas obtenidas analíticamente frente a las gráficas simuladas en SimScape. Los resultados son los siguientes:

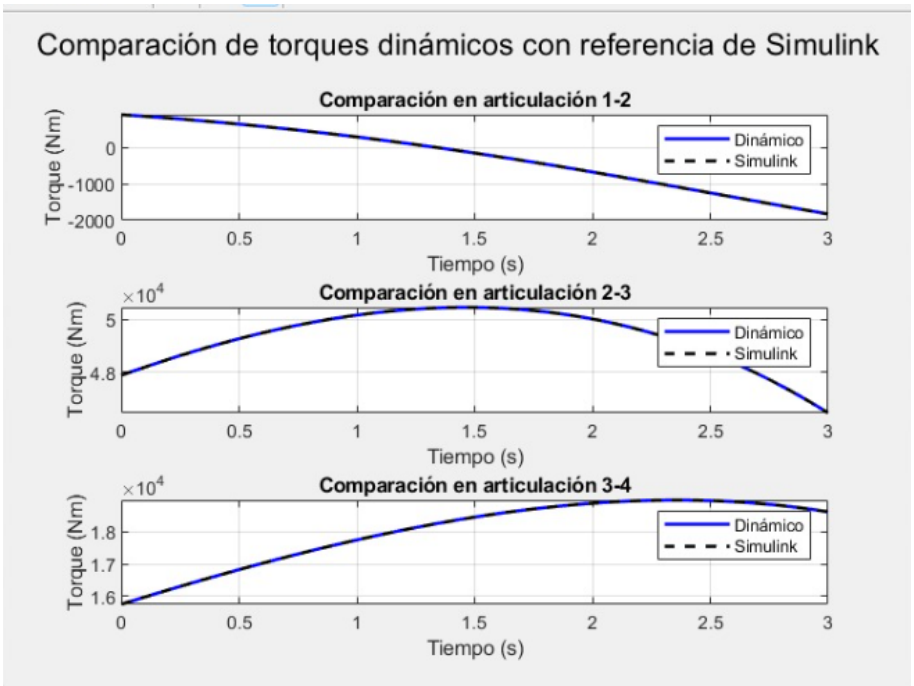


Figura 21: Comparación de torques mediante el método de Euler-Lagrange

Anexo: Modelo Simulink Completo

El Simulink resultante de todo el proyecto es el siguiente:

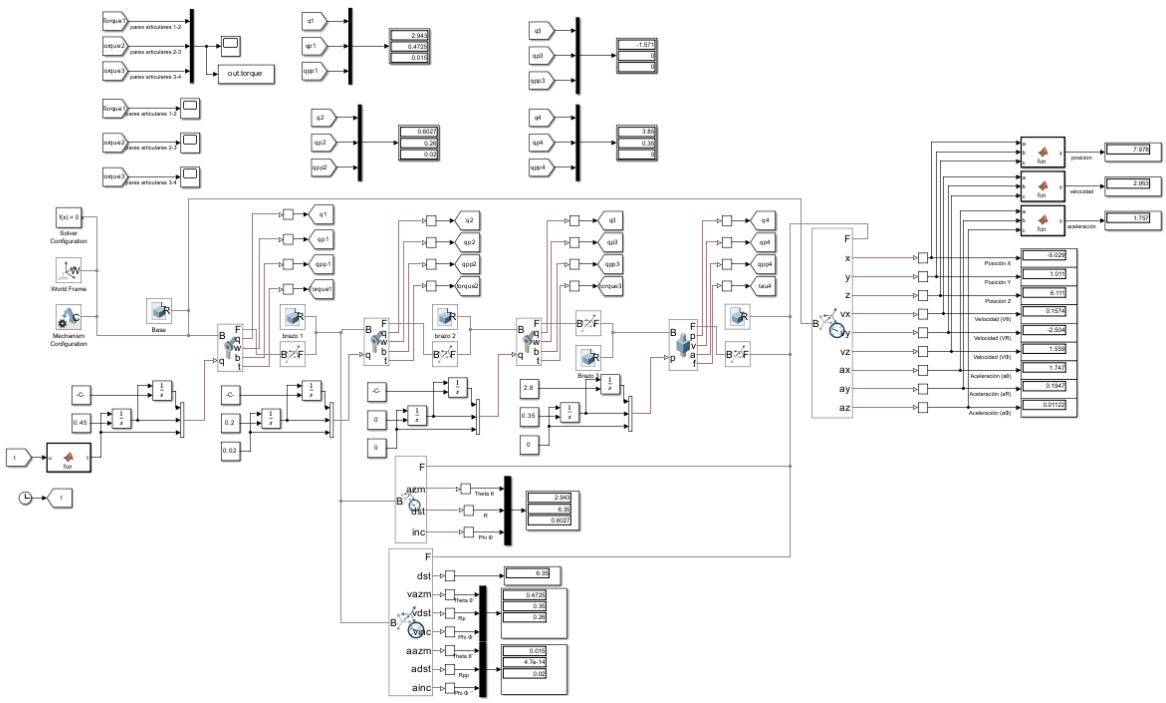


Figura 22: Simulink completo del Proyecto

Reparto de Tareas

El equipo mantuvo una estrecha colaboración, asegurando que cada componente del proyecto -memoria, scripts y simulaciones- recibiera atención integral por parte de todos los miembros. Esta dinámica permitió solucionar eficientemente los desafíos que surgieron en las distintas fases del trabajo.

Conclusiones

Los resultados del trabajo han sido completamente satisfactorios, mostrando una coincidencia prácticamente exacta entre los cálculos analíticos (MATLAB) y las simulaciones (Simscape). Esta correspondencia, con discrepancias mínimas inferiores al 4 %, confirma tanto la validez teórica de los modelos desarrollados como la precisión de su implementación práctica, particularmente en el cálculo de pares articulares mediante los métodos Newton-Euler y Euler-Lagrange.

Bibliografía

Tutoriales y Referencias

`https://www.youtube.com/watch?v=xpA8TKEMpMk`

`https://www.youtube.com/watch?v=-S0bR8ewcVY&t=54s`

Documentación Oficial MATLAB

- MathWorks Documentation. *Simscape Multibody*. `https://es.mathworks.com/help/physmod/sm/ug/modeling-mechanical-systems.html`
- MathWorks Robotics. *Rigid Body Tree Modeling*. `https://es.mathworks.com/help/robotics/ug/rigid-body-tree-robot-modeling.html`