Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ

Sprawozdanie Robotyka Kosmiczna

Model dynamiki układu o dwóch stopniach swobody zamontowany na swobodnej bazie

Autor:

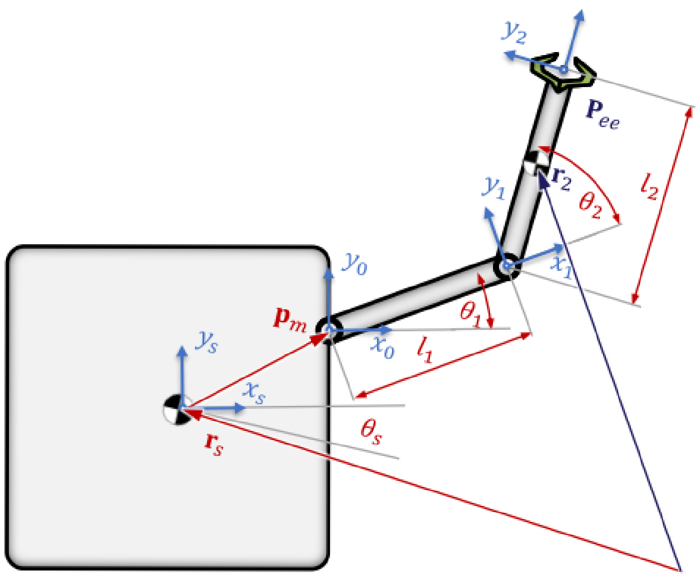
Dawid Lisek

Nr indeksu: 402382
Kierunek studiów: **Automatyka i Robotyka**
Specjalizacja: **Komputerowe Systemy Sterowania**
Grupa: pn 13:15 - 15:45

1. Cel ćwiczenia.

Celem ćwiczenia było zaimplementowanie modelu dynamiki manipulatora 2DoF zamontowanego na swobodnej bazie. Następnie należało przeprowadzić symulację dynamiki prostej przy pomocy procedury ode4. Wewnątrz procedury należało użyć momentów napędowych wygenerowanych dla manipulatora ziemskiego co umożliwiło porównanie uzyskanych wyników z manipulatorem ziemskim. Należało przeprowadzić kilka symulacji zmieniając parametry masowe satelity.

2. Model dynamiki manipulatora płaskiego o dwóch stopniach swobody zamontowany na swobodnej bazie.



Dane:

$l_1 = 0{,}6m$

$l_2 = 0{,}5m$

$m_1 = 2{,}5kg$

$m_2 = 1{,}5kg$

$g = 0$

$\mathbf{p}_m = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} 0\ m \\ 0\ m \end{bmatrix}$

$m_s = 35kg$

$I_s = 1.5kg.m^2$

Rysunek 1 Schemat manipulatora 2DoF zamontowany na swobodnej bazie ze zdefiniowanymi parametrami.

Model dynamiki został zaimplementowany przy pomocy poniższych równań:

Równanie 1 Równania opisujące model dynamiki prostej manipulatora 2DoF

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = Q$$

Gdzie:

$$q = [\, x_s \quad y_s \quad \theta_s \quad \theta_1 \quad \theta_2]^T$$
$$Q = \left[F_x F_y T_0 u_1 u_2\right]^T$$

Pierwsze 3 składowe wektora q opisują pozycję satelity w składowych X,Y oraz jego orientację. Składowe θ_1, θ_2 odnoszą się do pozycji kątowych przegubów manipulatora. Wektor Q zawiera siły, które oddziałują na poszczególne składowe wektora q. Macierze masowa M oraz wektor sił bezwładności i Coriolisa C były wyznaczane z udostępnionego skryptu.

W celu przeprowadzenia symulacji dynamiki prostej należało przekształcić powyższy model do poniższej postaci:

Równanie 2 Przekształcenie równania nr 1

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = Q$$

$$\ddot{q} = M(q)^{-1}(Q - C(q,\dot{q})\dot{q})$$

Umożliwiło to zastosowania procedury ode4, która numerycznie rozwiązywała powyższe równanie różniczkowe.

3. Skrypt źródłowy zaimplementowanego modelu dynamiki.

Do powyższego modelu dynamiki zostały zaimplementowane dwa skrypty.

Funkcja dynamics_matrix:
[M_matrix, C_matrix] = dynamics_matrix(p1, p2, m0, m1, m2, L1, L2, a1, a2, I0, I1, I2)
Jako parametry przyjmowała ona:

- współrzędne zamontowania manipulatora p1 i p2
- masę satelity oraz członów manipulatora m0, m1, m2
- długości członów manipulatora L1 i L2
- środki ciężkości członów manipulatora a1 i a2
- momenty bezwładności satelity oraz członów manipulatora I0, I1 oraz I2

Na ich podstawie wyznaczane były uchwyty na funkcję, które pozwalały na wyznaczenie macierzy M oraz wektora C wykorzystując składowe wektora $q \; oraz \; \dot{q}$ w postaci symbolicznej.

```
function [M_matrix, C_matrix] = dynamics_matrix(p1, p2, m0, m1, m2, L1, L2, a1, a2, I0, I1,
I2)
    % p1  % manipulator mounting point in x axis
    % p2  % manipulator mounting point in y axis
    % m0  % the mass of the satellite
    % m1  % the mass of the first kinematic pair
    % m2  % the mass of the second kinematic pair
    % a1  % center of mass of the first link L1/2
    % a2  % center of mass of the second link L2/2
    % L1  % the length of the first link
    % L2  % the length of the second link
    % I0  % Satellite moment of inertia
    % I1  % Moment of inertia of the first kinematic pair 1/12 mL^2
    % I2  % Moment of inertia of the second kinematic pair
    syms x1  % x-positon of the satellite's center of mass [m]
    syms x2  % y-positon of the satellite's center of mass [m]
    syms x3  % satellite orientation [rad]
    syms x4  % angular position of the first joint [rad]
    syms x5  % angular position of the second joint [rad]
    syms x6  % the first derivative of x1 [m/sec]
    syms x7  % the first derivative of x2 [m/sec]
    syms x8  % the first derivative of x3 [rad/sec]
    syms x9 % the first derivative of x4 [rad/sec]
    syms x10 % the first derivative of x5 [rad/sec]

    % Base (Satellite)
    J0v = [1 0 0 0 0 ;
           0 1 0 0 0 ];
    J0vt = J0v';
    J0w = [0 0 1 0 0 ];
    J0wt = J0w';

    % First link
    var1 = - p1*sin(x3) -p2*cos(x3) - a1*sin(x3+x4);
    var2 = -a1*sin(x3+x4);
    var3 = -p2*sin(x3) + p1*cos(x3)+a1*cos(x4+x3);
    var4 = a1*cos(x4+x3);
    J1v = [1 0  var1  var2 0 ;
           0 1  var3  var4 0 ];
    J1vt = J1v';
    J1w = [0 0 1 1 0];
    J1wt = J1w';

    % Second link
     var5 = -p2*cos(x3) - L1*sin(x4+x3) - p1*sin(x3) - a2*sin(x4+x5+x3);
     var6 = -L1*sin(x4+x3) - a2*sin(x4+x5+x3);
     var7 = -a2*sin(x4+x5+x3);
     var8 = -p2*sin(x3) + a2*cos(x4+x5+x3) + L1*cos(x4+x3)+p1*cos(x3);
```

```
    var9 =  a2*cos(x4+x5+x3) + L1*cos(x4+x3);
    var10 = a2*cos(x4+x5+x3);
    J2v = [1 0   var5 var6  var7 ;
           0 1   var8 var9 var10 ];
    J2vt = J2v';
    J2w = [0 0 1 1 1 ];
    J2wt = J2w';

    % M matrix

    Tv = m0*J0vt*J0v +  m1*J1vt*J1v +  m2*J2vt*J2v;
    Tw = J0wt*I0*J0w +  J1wt*I1*J1w +  J2wt*I2*J2w;
    M  = Tv+Tw;

    % C matrix

    Qdot = [x6; x7; x8; x9; x10];
    Qdott =[x6 x7 x8 x9 x10 ];
    Q = [x1; x2; x3; x4; x5];

    for i=1:1:size(Qdot,1)
        for j=1:1:size(Qdot,1)
            sum = 0;
            for k=1:1:size(Qdot,1)
               sum = sum + diff(M(i,j),Q(k))*Qdot(k);
            end
            N1(i,j)=sum;
        end
    end
    for i=1:1:size(Qdot,1)
        for j=1:1:size(Qdot,1)
            for k=1:1:size(Qdot,1)
              P(j,k) = diff(M(j,k),Q(i));
            end
        end
        N2(i,1) = Qdott*P*Qdot;
        clear P;
    end

    C = N1*Qdot - 0.5*N2;

    M_matrix = matlabFunction(M);
    C_matrix = matlabFunction(C);
end
```

Listing 1 Kod źródłowy funkcji dynamics_matrix

Kolejna funkcja ode4 na podstawie modelu przeprowadzała symulację dynamiki prostej rozwiązując numerycznie równanie 2.

[q, dq] = ode4(Q, q_0, dq_0, C_handle, M_handle, h)

Jako argumenty przyjmowała ona wektor sił oraz momentów obecnych w układzie Q, wartości początkowe wektora q oraz wartości początkowe jego pochodnej, uchwyty funkcyjne do wyznaczenia macierzy M oraz C, krok symulacji h. Zwracany był wyznaczony wektor q oraz jego pochodna dq.

```
function [q, dq] = ode4(Q, q_0, dq_0, C_handle, M_handle, h)


    q(:, 1) = q_0;
    dq(:, 1) = dq_0;

    for i = 1:1:length(Q)-1


        M = M_handle(q(3, i), q(4, i), q(5, i));
        C = C_handle(q(3, i), q(4, i), q(5, i), dq(1, i), dq(2, i), dq(3, i), dq(4, i), dq(5, i));

        K_11 = h .* dq(:, i);
        K_12 = h .* (inv(M) * (Q(:, i) - C));

        K_21 = h .* (dq(:, i) + K_11/2);
        k_12_dq = dq(:, i) + (K_12/2);
        C_22 = C_handle(q(3, i), q(4, i), q(5, i), k_12_dq(1), k_12_dq(2), k_12_dq(3), k_12_dq(4), k_12_dq(5));
        K_22 = h .* (inv(M) * (Q(:, i) - C_22));

        K_31 = h .* (dq(:, i) + K_21/2);
        k_22_dq = dq(:, i) + (K_22/2);
        C_32 = C_handle(q(3, i), q(4, i), q(5, i), k_22_dq(1), k_22_dq(2), k_22_dq(3), k_22_dq(4), k_22_dq(5));
        K_32 = h .* (inv(M) * (Q(:, i) - C_32));

        K_41 = h .* (dq(:, i) + K_31);
        k_32_dq = dq(:, i) + (K_32);
        C_42 = C_handle(q(3, i), q(4, i), q(5, i), k_32_dq(1), k_32_dq(2), k_32_dq(3), k_32_dq(4), k_32_dq(5));
        K_42 = h .* (inv(M) * (Q(:, i) - C_42));

        K_1 = (1/6) * (K_11 + 2*K_21 + 2*K_31 + K_41);
        K_2 = (1/6) * (K_12 + 2*K_22 + 2*K_32 + K_42);

        q(:, i + 1) = q(:,i) + K_1;
        dq(:, i + 1) = dq(:,i) + K_2;
    end

end
```

Listing 2 Kod źródłowy funkcji ode4

Ponieważ wektor sił bezwładności i Coriolisa C był iloczynem macierzy C oraz pochodnej wektora q konieczne było czterokrotne wyznaczenie wektora C dla wektora pochodnych dq powiększonego o współczynniki K12/2, K22/2, K32.

Wektor sił oraz momentów działających w układzie miał postać: $Q = [000u_1u_2]^T$ gdzie momenty napędowe $u_1\ i\ u_2$ były momentami wygenerowanymi dla manipulatora ziemskiego na poprzednich zajęciach.

4. **Wykresy momentów użytych w punkcie 2 instrukcji.**
Do użycia funkcji ode4() potrzebne momenty napędowe dla dwóch przegubów manipulatora. W tym celu została wywołana funkcja Trajectory_Generation dla dwóch różnych zestawów danych.

- **Przegub 1**

```
q_p_1 = 0;          % pozycja początkowa przegubu nr 1 [rad]
dq_p_1 = 0.0;       % prędkość początkowa przegubu nr 1 [rad]
q_k_1 = 1/2 * pi;   % pozycja końcowa przegubu nr 1 [rad/s]
dq_k_1 = 0.0;       % prędkość końcowa przegubu nr 1 [rad/s]
Tk_1 = 10;          % Czas trwania ruchu przegubu [s]
Ta_1 = 4;           % Czas trwania fazy przyspieszania [s]
Tb_1 = 4;           % Czas trwania fazy hamowania [s]
dt_1 = 0.01;        % Krok czasowy [s]
V_1 = 0.2;          % Prędkość przegubu w fazie drugiej [rad/s]
[q_1,dq_1,ddq_1, t_1] = Trajectory_Generation(solved, q_p_1, dq_p_1,...
q_k_1, dq_k_1, Tk_1, Ta_1, Tb_1, dt_1, V_1);
```

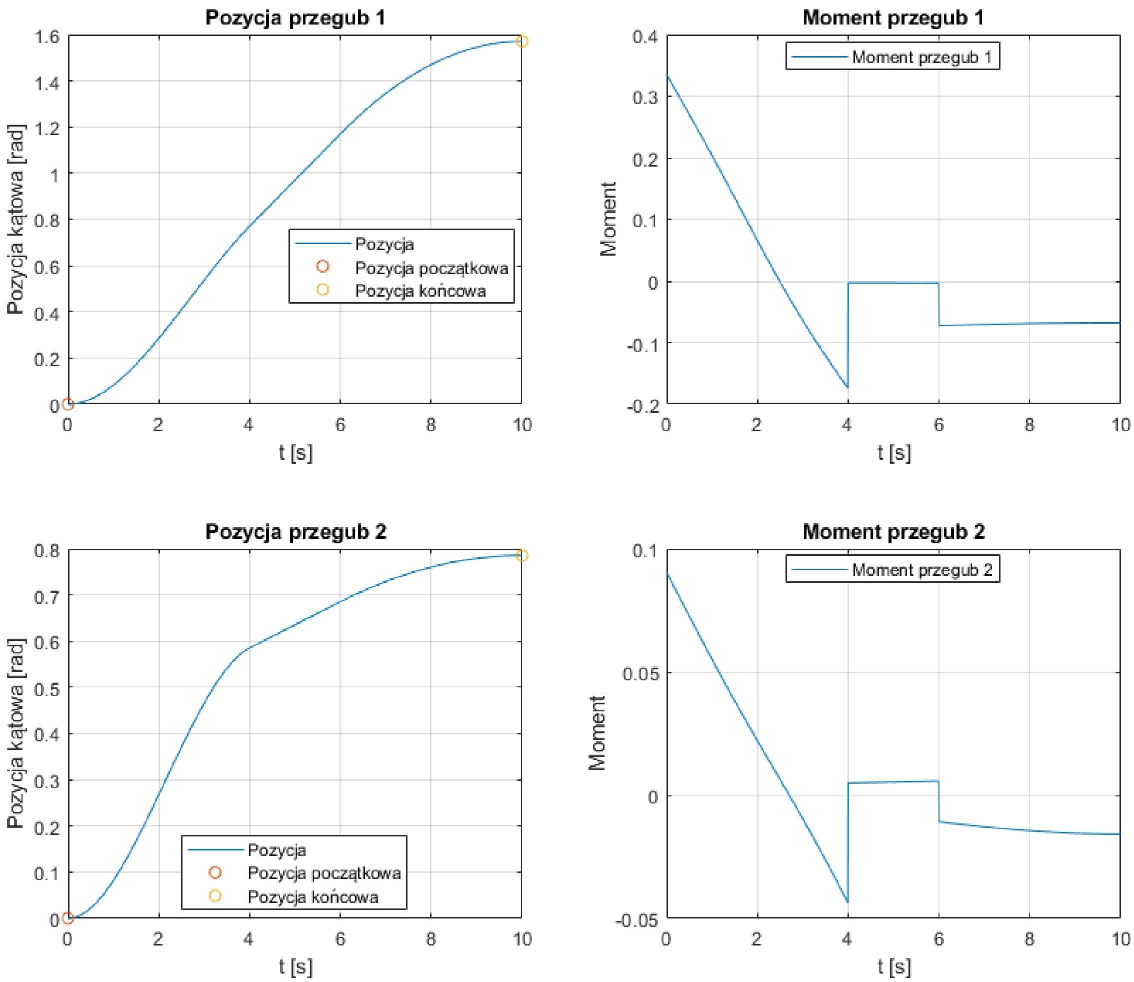
- **Przegub 2**

```
q_p_2 = 0;          % pozycja początkowa przegubu nr 1 [rad]
dq_p_2 = 0.0;       % prędkość początkowa przegubu nr 1 [rad]
q_k_2 = 1/4 * pi;   % pozycja końcowa przegubu nr 1 [rad/s]
dq_k_2 = 0.0;       % prędkość końcowa przegubu nr 1 [rad/s]
Tk_2 = 10;          % Czas trwania ruchu przegubu [s]
Ta_2 = 4;           % Czas trwania fazy przyspieszania [s]
Tb_2 = 4;           % Czas trwania fazy hamowania [s]
dt_2 = 0.01;        % Krok czasowy [s]
V_2 = 0.05;          % Prędkość przegubu w fazie drugiej [rad/s]
[q_2,dq_2,ddq_2, t_2] = Trajectory_Generation(solved, q_p_2, dq_p_2, ...
    q_k_2, dq_k_2, Tk_2, Ta_2, Tb_2, dt_2, V_2);
```

Następnie dla wygenerowanych pozycji oraz prędkości przegubów manipulatora zostały wygenerowane momenty napędowe $u_1 \; i \; u_2$ dla manipulatora ziemskiego przy pomocy funkcji dynamics_model_earth():

```
function [u] = dynamics_model_earth(q1, q2, dq1, dq2, ddq1, ddq2, m1, m2, L1, L2)

    n = length(q1);
    for i=1:n
        M = [(((1/3) * m1) + m2) * L1^2 + ((1/3) * m2 * L2^2) + (m2 * L1 * L2 * cos(q2(i))), ...
       ((1/3) * m2 * L2^2) + ((1/2) * m2 * L1 * L2 * cos(q2(i))); ...
       ((1/3) * m2 * L2^2) + ((1/2) * m2 * L1 * L2 * cos(q2(i))), ...
       ((1/3) * m2 * L2^2)];

        ddq = [ddq1(i); ddq2(i)];

        C = [0, (-m2 * L1 * L2 * (dq1(i) + (1/2) * dq2(i)) * sin(q2(i))); ...
       ((1/2) * m2 * L1 * L2 * dq1(i) * sin(q2(i))), 0];

        dq = [dq1(i); dq2(i)];

        u(:, i) = M*ddq + C*dq;

    end
end
```

Listing 3 Kod źródłowy generujący momenty napędowe dla manipulatora ziemskiego



Rysunek 2 Wykres wygenerowanych pozycji oraz momentów napędowych dwóch przegubów manipulatora

5. Wyniki przeprowadzonej symulacji dynamiki prostej (przebiegi pozycji kątowych przegubów , przebiegi pozycji i orientacji końcówki roboczej oraz przebiegi pozycji i orientacji satelity).

Wygenerowane momenty zostały wstawione do wektora Q, który następnie posłużył do symulacji dynamiki prostej za pomocą funkcji ode4(). Wektor Q oraz funkcja ode4 została wywołana na następujący sposób:

```
F_x = zeros(1, length(u));
F_y = zeros(1, length(u));
T_0 = zeros(1, length(u));
Q = [F_x; F_y; T_0; u];
h = 0.01;
q0 = [0; 0; 0; 0; 0];
dq0 = [0; 0; 0; 0; 0];
[q_space, dq_space] = ode4(Q, q0, dq0, C_matrix, M_matrix, h);
```

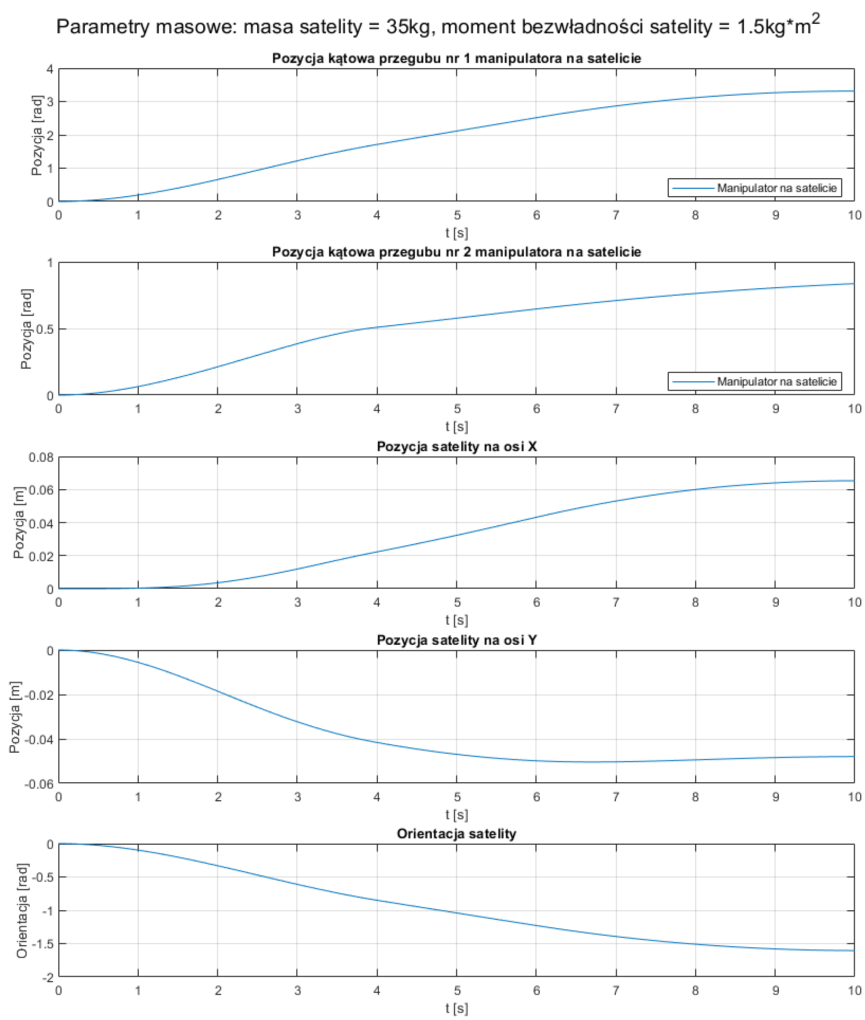
Listing 4 Wywołanie funkcji ode4

Macierz M oraz wektor C zostały policzone wewnątrz funkcji dynamics_matrix() z poniższymi parametrami:

```
p1 = 0; % p1  % manipulator mounting point in x axis
p2 = 0; % p2  % manipulator mounting point in y axis
m0 = 35; % m0  % the mass of the satellite
m1 = 2.5; % m1  % the mass of the first kinematic pair
m2 = 1.5; % m2  % the mass of the second kinematic pair
L1 = 0.6; % L1  % the length of the first link
L2 = 0.5; % L2  % the length of the second link
a1 = L1/2; % a1  % center of mass of the first link L1/2
a2 = L2/2; % a2  % center of mass of the second link L2/2
I0 = 1.5; % I0  % Satellite moment of inertia
I1 = (1/12) * m1 * L1^2; % I1  % Moment of inertia of the first kinematic pair 1/12 mL^2
I2 = (1/12) * m2 * L2^2; % I2  % Moment of inertia of the second kinematic pair
```

Listing 5 Parametry modelu wykorzystane w funkcji dynamics_matrix()

Uzyskane wyniki symulacji dynamiki prostej:



Rysunek 3 Przebiegi pozycji kątowych przegubów, pozycji satelity oraz jej orientacji.

Aby uzyskać przebieg pozycji oraz orientacji członu roboczego utworzona została funkcja direct_2DoF_space(), która na podstawie pozycji kątowych przegubów, pozycji satelity oraz orientacji satelity zwracała pozycję i orientację członu roboczego.

```
function [Pee, Psiee] = direct_2DoF_space(q, x_s, y_s, q_s, L1, L2)

%   q – dwuelementowy wektor zawierający pozycje kątowe przegubów [rad]
%   dq – dwuelementowy wektor zawierający prędkości kątowe przegubów[rad/s].
%   q_s - orientacja satelity [rad]
%   x_s, y_s - składowa X i składowa Y pozycji satelity [m]
%   L1, L2 – długości członów manipulatora [m]
%   Pee – dwuelementowy wektor zawierający składową X i składową Y pozycji członu roboczego [m].
%   Psiee – orientacja członu roboczego [rad].

    Pee = [cos(q(:, 1)) .* L1 + cos(q(:,1) + q(:,2)) .* L2 + x_s, sin(q(:,1)) .* L1 + sin(q(:,1) + q(:,2)) .* L2 + y_s];
    Psiee = q(:,1) + q(:,2) + q_s;
end
```

Listing 6 Funkcja direct_2DoF_space

Poszczególne składowe pozycji członu roboczego są wyznaczane w następujący sposób:

Równanie 3 Równania na pozycję członu roboczego manipulatora na satelicie

$$x_{ee_{space}} = x_{ee} + x_s$$
$$y_{ee_{space}} = y_{ee} + y_s$$
$$P_{ee_{space}} = \left[x_{ee_{space}}, y_{ee_{space}}\right]$$

Orientacja członu roboczego jest obliczana w analogiczny sposób:

Równanie 4 Równanie na orientację członu roboczego manipulatora na satelicie

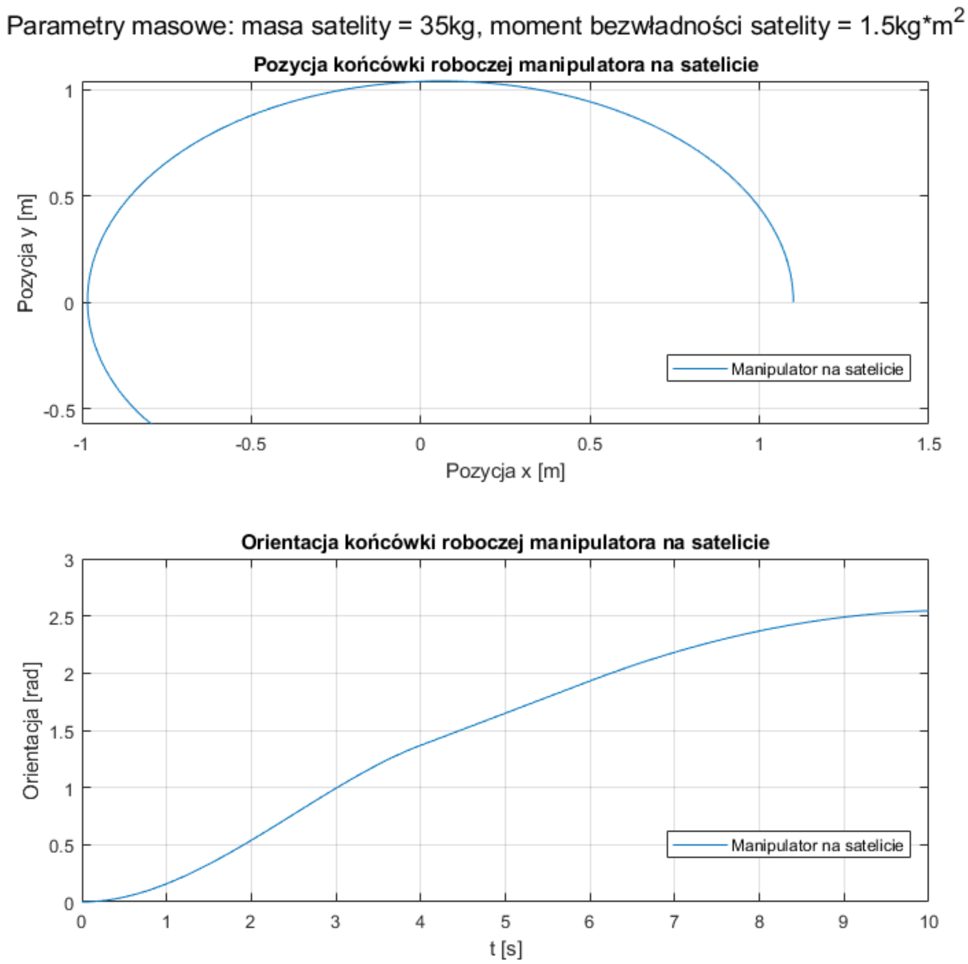
$$P_{siee} = \theta_1 + \theta_2 + \theta_s$$

Funkcja została wywołana w poniższy sposób:

```
%%% Pozycja i orientacja końcówki manipulatora na satelicie
[Pee_space, Psiee_space] = direct_2DoF_space([q_space(4, :); q_space(5, :)]', q_space(1, :)', ...
    q_space(2, :)', q_space(3, :)', L1, L2);
```

Listing 7 Wywołanie funkcji direct_2DoF_space.

Wykres uzyskanej pozycji oraz orientacji członu roboczego:



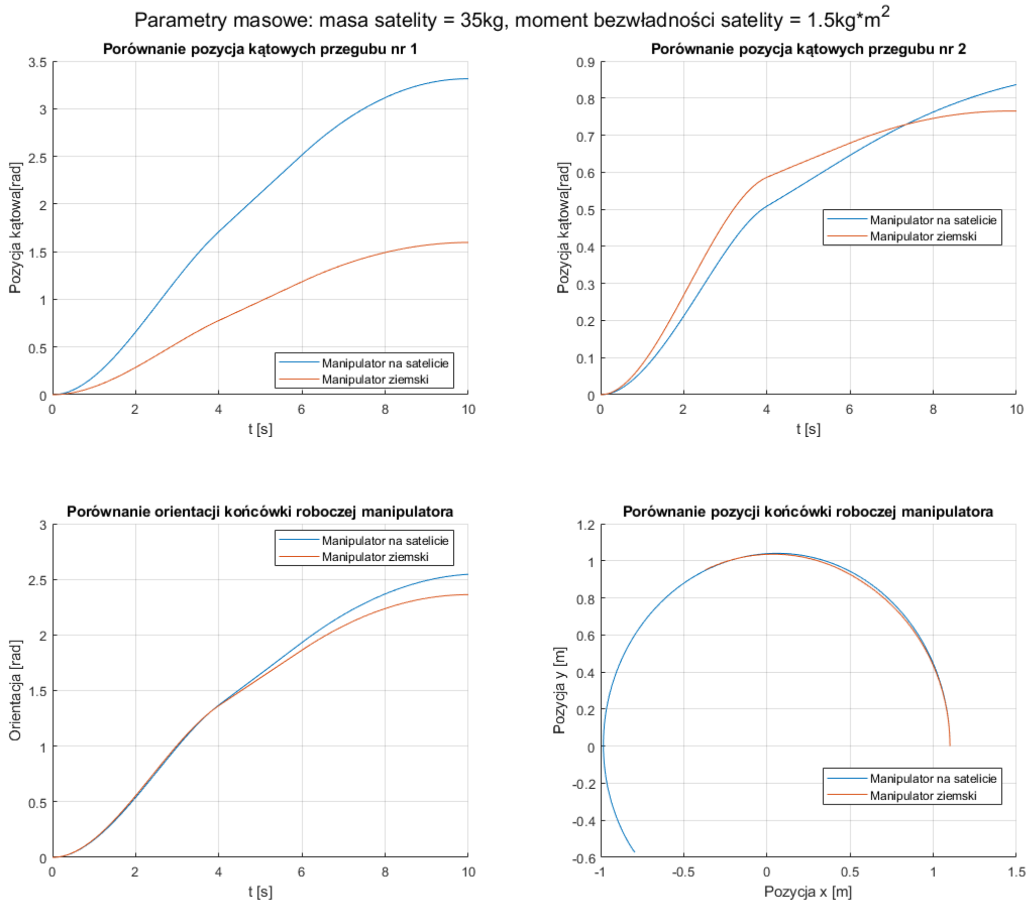
Rysunek 4 Wykres pozycji oraz orientacji członu roboczego manipulatora zamontowanego na swobodnej bazie.

6. Wyniki porównania przebiegów pozycji kątowych przegubów oraz przebiegów pozycji i orientacji końcówki roboczej uzyskanymi w punkcie 2 z przebiegami uzyskanymi na poprzednich zajęciach (dla manipulatora „ziemskiego”).

Uzyskane w poprzednim punkcie wyniki zostały porównane z wynikami uzyskanymi dla manipulatora ziemskiego. Dla momentów uzyskanych w punkcie 4 użyta została procedura ode4 stworzona na poprzednich zajęciach oraz funkcja direct_2DoF dla manipulatora ziemskiego zaimplementowana na zajęciach nr 2.

```
u = double(u);
[q_earth, dq_earth] = ode4_earth(u, q0, dq0, L1, L2, m1, m2);
[Pee_earth, Psiee_earth, Vee_earth, Omee_earth] = direct_2DoF_earth(q_earth', dq_earth', L1, L2);
```

Listing 8 Sposób uzyskania przebiegów pozycji oraz orientacji członu roboczego manipulatora na ziemi



Rysunek 5 Porównanie przebiegów pozycji kątowych przegubów oraz przebiegów pozycji i orientacji końcówki roboczej manipulatora ziemskiego i umieszczonego na satelicie

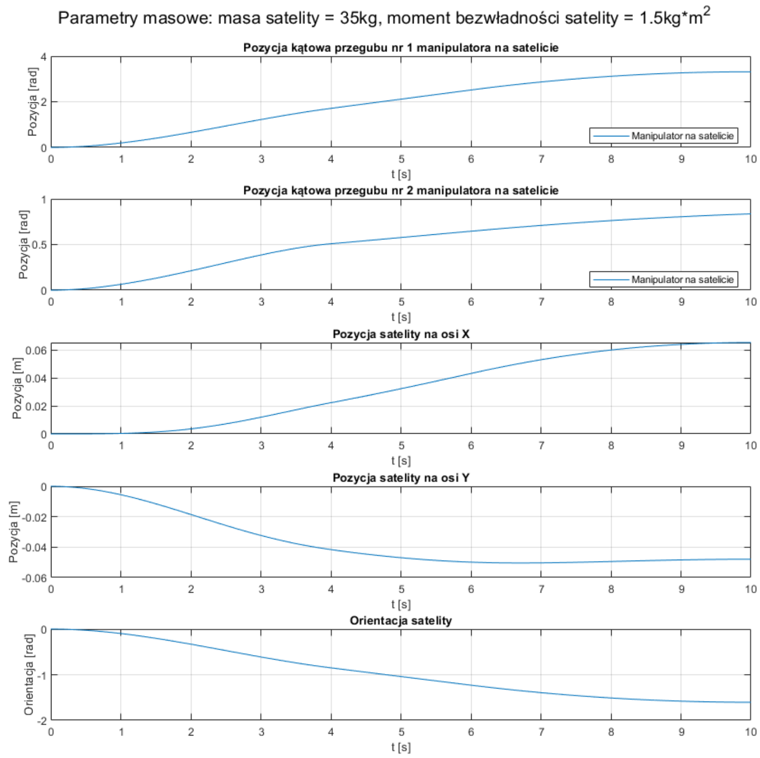
7. Przebiegi pozycji kątowych przegubów , przebiegi pozycji i orientacji końcówki roboczej oraz przebiegi pozycji i orientacji satelity dla trzech różnych parametrów masowych satelity. Wnioski z otrzymanych wyników.

Kolejnym krokiem była zmiana parametrów masowych satelity. Zmieniana była masa satelity oraz jej moment bezwładności. W tym celu zostały przygotowane 3 poniższe zestawy danych:

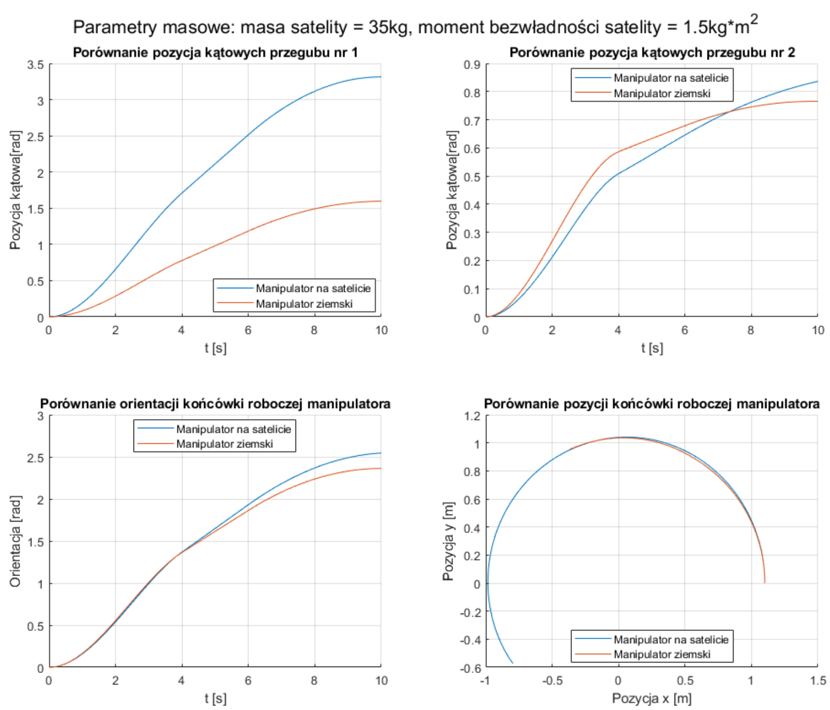
- Zestaw 1 – m_0 = 35 kg, I_0 = 1.5 kg*m^2
- Zestaw 2 – m_0 = 70 kg, I_0 = 30 kg*m^2
- Zestaw 3 – m_0 = 300 kg, I_0 = 100 kg*m^2

Dodatkowo dla każdego zestawu danych przebiegi pozycji kątowych przegubów oraz przebiegi pozycji i orientacji końcówki roboczej były porównywane z manipulatorem ziemskim.

1) Zestaw 1 – m_0 = 35 kg, I_0 = 1.5 kg*m^2

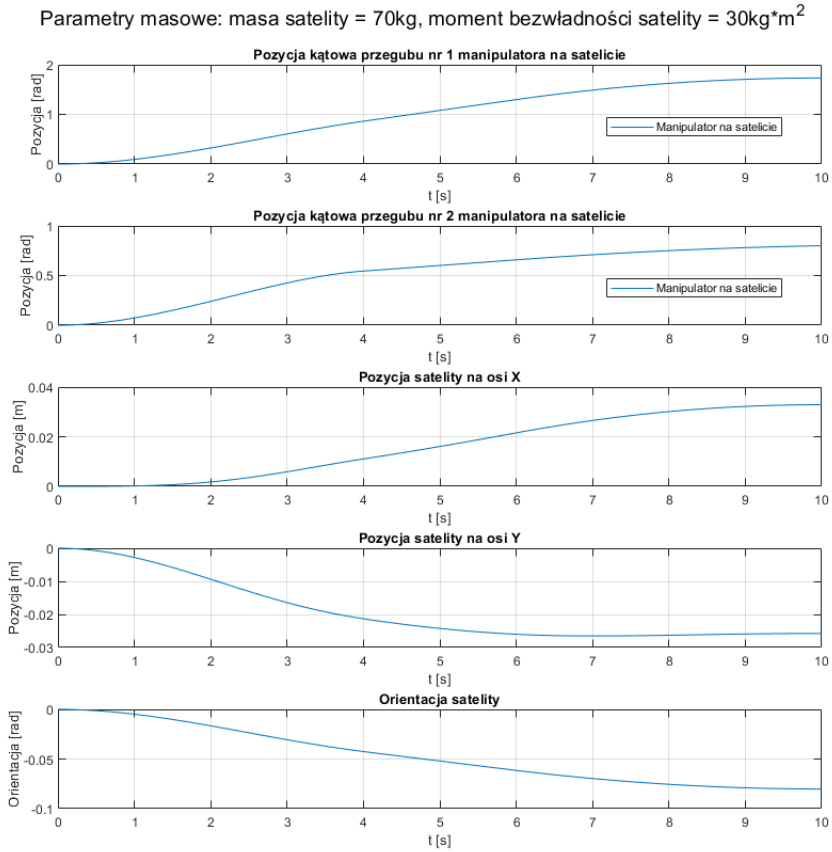


Rysunek 6 Przebiegi pozycji kątowych przegubów oraz przebiegi pozycji i orientacji satelity dla zestawu nr 1

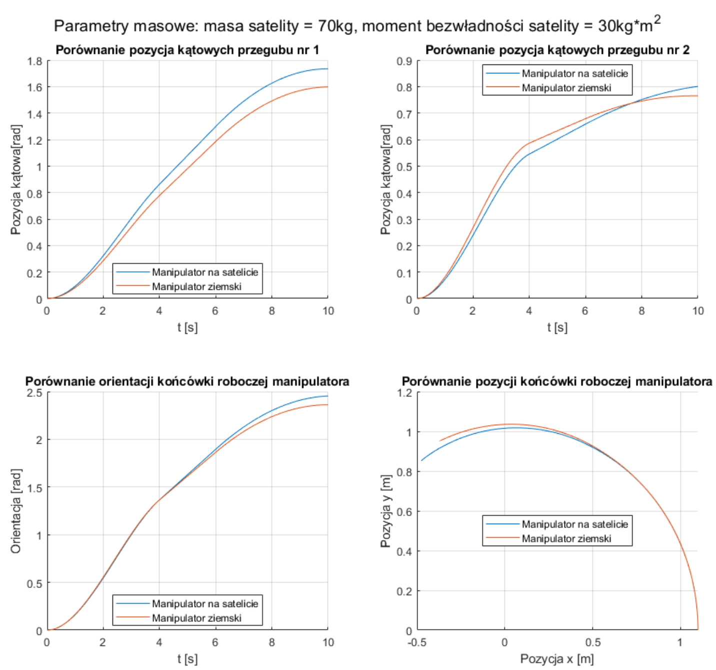


Rysunek 7 Przebiegi pozycji i orientacji końcówki roboczej manipulatora na satelicie dla zestawu nr 1 oraz manipulatora ziemskiego

2) Zestaw 2 – m_0 = 70 kg, I_0 = 30 kg*m^2

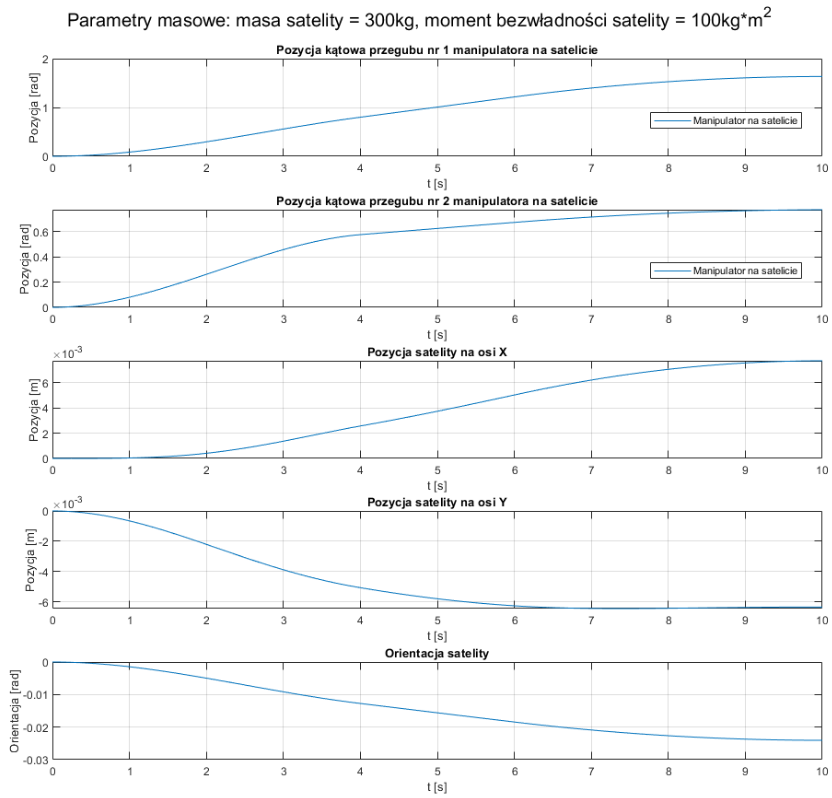


Rysunek 8 Przebiegi pozycji kątowych przegubów oraz przebiegi pozycji i orientacji satelity dla zestawu nr 2

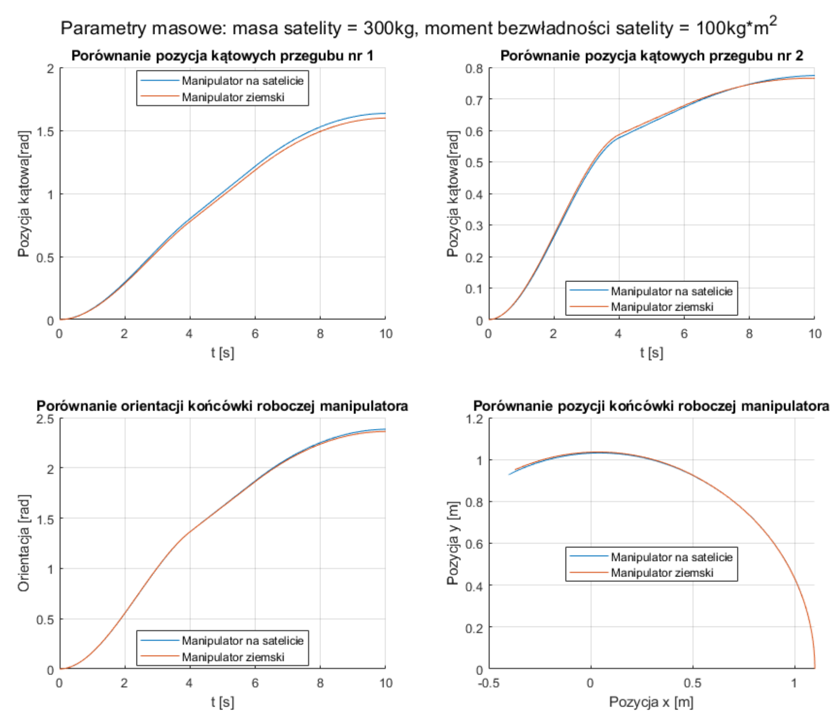


Rysunek 9 Przebiegi pozycji i orientacji końcówki roboczej manipulatora na satelicie dla zestawu nr 2 oraz manipulatora ziemskiego

3) Zestaw 3 – m_0 = 300 kg, I_0 = 100 kg*m^2



Rysunek 10 Przebiegi pozycji kątowych przegubów oraz przebiegi pozycji i orientacji satelity dla zestawu nr 3



Rysunek 11 Przebiegi pozycji i orientacji końcówki roboczej manipulatora na satelicie dla zestawu nr 3 oraz manipulatora ziemskiego

Z powyższych wykresów wynika, że wraz ze wzrostem parametrów masowych satelity manipulator osiąga pozycje kątowe przegubów oraz pozycję i orientację końcówki roboczej zbliżoną do manipulatora ziemskiego. Dla trzeciego zestawu danych manipulator umieszczony na satelicie osiąga praktycznie identyczne wyniki jak manipulator ziemski. Dla trzeciego zestawu danych pozycja satelity zmienia się zaledwie o kilka milimetrów, a orientacja o setne części radiana. Dla porównania w zestawie danych nr 1 pozycja satelity zmieniała się o kilka centymetrów, a orientacja o prawie dwa radiany.

Zjawisko to wynika z zasady zachowania pędu układu. Pęd pochodzący od ruchu członów manipulatora musi być równoważony przez pęd satelity. Jeżeli zwiększamy masę satelity to jego prędkość musi zmaleć aby zachowana została zasada zachowania pędu. Powyższe wykresy dobrze ilustrują to zjawisko dla zwiększających się parametrów masowych satelity.

8. Wyprowadzenie przykładowego elementu macierz M korzystając z równania Lagrange'a II rodzaju oraz porównanie otrzymanego wyniku z wynikiem otrzymanym z udostępnionego skryptu.

Równanie Lagranange'a II rodzaju ma postać:

Równanie 5 Równanie Lagranange'a II rodzaju

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial \dot{q}}\right) - \left(\frac{\partial L}{\partial q}\right) = 0$$

Aby wyprowadzić przykładowy element macierzy M należało wyznaczyć powyższe pochodne cząstkowe z równania Lagranżian układu, który jest równy różnicy energii kinetycznych i potencjalnych w układzie. W manipulatorze umieszczonym na satelicie energie potencjalne w układzie są równe 0, więc Lagranżian będzie równy sumie energii kinetycznych występujących w układzie. Energia kinetyczna w układzie jest opisana poniższym równaniem:

Równanie 6 Równanie na całkowitą energię kinetyczną w układzie.

$$E_k = \frac{1}{8} m_1 \left[\left(2\,\dot{y}_s + 2\,\dot{\theta}_s p_x cos(\theta_s) - 2\dot{\theta}_s p_y sin(\theta_s) + l_1 \dot{\theta}_s cos(\theta_s + \theta_1) + l_1 \dot{\theta}_1 cos(\theta_s + \theta_1) \right)^2 \right.$$

$$\left. + \left(2\dot{\theta}_s p_y cos(\theta_s) - 2\,\dot{x}_s + 2\dot{\theta}_s p_x sin(\theta_s) + l_1 \dot{\theta}_s sin(\theta_s + \theta_1) + l_1 \dot{\theta}_1 sin(\theta_s + \theta_1) \right)^2 \right]$$

$$+ \frac{1}{2} I_0 \dot{\theta}_2^2 + \frac{1}{8}\, m_2 \left[\left(2\,\dot{y}_s + 2\dot{\theta}_s p_x cos(\theta_s) - 2\dot{\theta}_s p_y sin(\theta_s) + l_2 \dot{\theta}_s cos(\theta_s + \theta_1 + \theta_2) - \right. \right.$$

$$\left. - + l_2 \dot{\theta}_1 cos(\theta_s + \theta_1 + \theta_2) + l_2 \dot{\theta}_2 cos(\theta_s + \theta_1 + \theta_2) + 2\,l_1 \dot{\theta}_s cos(\theta_s + \theta_1) + 2\,l_1 \dot{\theta}_1 cos(\theta_s + \theta_1) \right)^2$$

$$+ \left(2\dot{\theta}_s p_y cos(\theta_s) - 2\,\dot{x}_s + 2\dot{\theta}_s p_x sin(\theta_s) + l_2 \dot{\theta}_s sin(\theta_s + \theta_1 + \theta_2) + l_2 \dot{\theta}_1 sin(\theta_s + \theta_1 + \theta_2) - \right.$$

$$\left. \left. - + l_2 \dot{\theta}_2 sin(\theta_s + \theta_1 + \theta_2) + 2\,l_1 \dot{\theta}_s sin(\theta_s + \theta_1) + 2\,l_1 \dot{\theta}_1 sin(\theta_s + \theta_1) \right)^2 \right] + \frac{1}{2}\, m_s (\dot{x}_s^2 + \dot{y}_s^2)$$

$$+ \frac{1}{24} l_2^2 m_2 \left(\dot{\theta}_s + \dot{\theta}_1 + \dot{\theta}_2 \right)^2 + \frac{1}{24} l_1^2 m_1 \left(\dot{\theta}_s + \dot{\theta}_1 \right)^2$$

Równanie to zostało przepisane do programu Matlab, a następnie przy wykorzystaniu pakietu Symbolic Math Toolbox zostały wyznaczone pochodne cząstkowe dla składowej X pozycji satelity oraz równanie Lagrange'a dla zmiennej x_1.

```
Ek(t) = (1/8)*m1*((2*x7(t) + 2*x8(t)*p1*cos(x3(t)) - 2*x8(t)*p2*sin(x3(t)) + L1*x8(t)*cos(x3(t)+ x4(t)) + L1*x9(t)*cos(x3(t)+ x4(t)))^2 ...
    + (2*x8(t)*p2*cos(x3(t)) - 2*x6(t) + 2*x8(t)*p1*sin(x3(t)) + L1*x8(t)*sin(x3(t)+x4(t)) + L1*x9(t)*sin(x3(t)+x4(t)))^2) ...
    + (1/2)*I0*x10(t)^2 + (1/8)*m2*((2*x7(t) + 2*x8(t)*p1*cos(x3(t)) - 2*x8(t)*p2*sin(x3(t)) + L2*x8(t)*cos(x3(t)+x4(t)+x5(t)) ...
    + L2*x9(t)*cos(x3(t)+x4(t)+x5(t)) + L2*x10(t)*cos(x3(t)+x4(t)+x5(t)) + 2*L1*x8(t)*cos(x3(t)+x4(t)) + 2*L1*x9(t)*cos(x3(t)+x4(t)))^2 ...
    + (2*x8(t)*p2*cos(x3(t)) - 2*x6(t) + 2*x8(t)*p1*sin(x3(t)) + L2*x8(t)*sin(x3(t)+x4(t)+x5(t)) + L2*x9(t)*sin(x3(t)+x4(t)+x5(t)) ...
    + L2*x10(t)*sin(x3(t)+x4(t)+x5(t)) + 2*L1*x8(t)*sin(x3(t)+x4(t)) + 2*L1*x9(t)*sin(x3(t)+x4(t)))^2) + (1/2)*m0*(x6(t)^2 + x7(t)^2) ...
    + (1/24)*L2^2*m2*(x8(t) + x9(t)+ x10(t))^2 + (1/24)*L1^2*m1*(x8(t)+x9(t))^2
dEk_dxs = diff(Ek, x1)
dEk_ddxs = diff(Ek, x6)
ddL_dtddxs = diff(dEk_ddxs, t)
L = ddL_dtddxs - dEk_dxs


L = subs(L, diff(x1(t), t), dx1(t));
L = subs(L, diff(x2(t), t), dx2(t));
L = subs(L, diff(x3(t), t), dx3(t));
L = subs(L, diff(x4(t), t), dx4(t));
L = subs(L, diff(x5(t), t), dx5(t));
L = subs(L, diff(x6(t), t), dx6(t));
L = subs(L, diff(x7(t), t), dx7(t));
L = subs(L, diff(x8(t), t), dx8(t));
L = subs(L, diff(x9(t), t), dx9(t));
L = subs(L, diff(x10(t), t), dx10(t))
```

Listing 9 Kod obliczający Lagranżian układu w programie Matlab

Następnie po podstawieniu parametrów liczbowych opisujących manipulator zostało wygenerowane poniższe równanie Lagrange'a manipulatora. Zmienną uogólnioną była składowa X pozycji satelity i odpowiadała ona pierwszemu wierszowi macierzy M.

Równanie 7 Równanie Lagranange'a II rodzaju dla pozycji satelity na osi X

$$39\,dx_6(t) - \frac{3\sin(\sigma_2)\ dx_8(t)}{8} - \frac{3\sin(\sigma_2)\ dx_9(t)}{8} - \frac{3\sin(\sigma_2)\ dx_{10}(t)}{8} - \frac{33\sin(\sigma_3)\ dx_8(t)}{20} - \frac{33\sin(\sigma_3)\ dx_9(t)}{20} - \frac{3\cos(\sigma_2)\ x_8(t)\,\sigma_1}{8} - \frac{3\cos(\sigma_2)\ x_9(t)\,\sigma_1}{8} - \frac{3\cos(\sigma_2)\ x_{10}(t)\,\sigma_1}{8} - \frac{33\cos(\sigma_3)\ x_8(t)\,\sigma_4}{20} - \frac{33\cos(\sigma_3)\ x_9(t)\,\sigma_4}{20}$$

Gdzie:

$$\sigma_1 = dx_3(t) + dx_4(t) + dx_5(t)$$

$$\sigma_2 = x_3(t) + x_4(t) + x_5(t)$$

$$\sigma_3 = x_3(t) + x_4(t)$$

$$\sigma_4 = dx_3(t) + x_4(t)$$

Biorąc pod uwagę równanie opisujące model dynamiki manipulatora:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = Q$$

Interesować nas będą współczynniki stojące przy pochodnych drugiego rzędu. Iloczyn $M(q)\ddot{q}$ dla zmiennej uogólnionej x_1 może być zapisany w postaci:

$$[\mathrm{M}_{11}\quad \mathrm{M}_{12}\quad \mathrm{M}_{13}\quad \mathrm{M}_{14}\quad \mathrm{M}_{15}] \cdot [dx_6(t)\quad dx_7(t)\quad dx_8(t)\quad dx_9(t)\quad dx_{10}(t)]^{\mathrm{T}}$$

Tak więc współczynniki macierzy M dla zmiennej uogólnionej x_1 będą równe:

$$\mathrm{M}_{11} = 39$$

$$\mathrm{M}_{12} = 0$$

$$\mathrm{M}_{13} = -\frac{3\sin\big(x_3(t) + x_4(t) + x_5(t)\big)}{8} - \frac{33\sin\big(x_3(t) + x_4(t) + x_5(t)\big)}{20}$$

$$\mathrm{M}_{14} = -\frac{3\sin\big(x_3(t) + x_4(t) + x_5(t)\big)}{8} - \frac{33\sin\big(x_3(t) + x_4(t)\big)}{20}$$

$$\mathrm{M}_{15} = -\frac{3\sin\big(x_3(t) + x_4(t) + x_5(t)\big)}{8}$$

Wprowadzając parametry liczbowe opisujące manipulator do udostępnionego skryptu obliczającego macierz M oraz wektor C otrzymamy symboliczne macierz M w postaci symbolicznej. Interesujące nas współczynniki mają postać:

M_11 = 39

M_12 = 0

M_13 =

$$-\frac{3\sin(x_3+x_4+x_5)}{8}-\frac{33\sin(x_3+x_4)}{20}$$

M_14 =

$$-\frac{3\sin(x_3+x_4+x_5)}{8}-\frac{33\sin(x_3+x_4)}{20}$$

M_15 =

$$-\frac{3\sin(x_3+x_4+x_5)}{8}$$

Rysunek 12 Współczynniki macierzy M z udostępnionego skryptu.

9. Skrypt źródłowy programu głównego.

```
solved = Z;
q_p_1 = 0;          % pozycja początkowa przegubu nr 1 [rad]
dq_p_1 = 0.0;       % prędkość początkowa przegubu nr 1 [rad]
q_k_1 = 1/2 * pi;   % pozycja końcowa przegubu nr 1 [rad/s]
dq_k_1 = 0.0;       % prędkość końcowa przegubu nr 1 [rad/s]
Tk_1 = 10;          % Czas trwania ruchu przegubu [s]
Ta_1 = 4;           % Czas trwania fazy przyspieszania [s]
Tb_1 = 4;           % Czas trwania fazy hamowania [s]
dt_1 = 0.01;        % Krok czasowy [s]
V_1 = 0.2;          % Prędkość przegubu w fazie drugiej [rad/s]
[q_1,dq_1,ddq_1, t_1] = Trajectory_Generation(solved, q_p_1, dq_p_1,...
  q_k_1, dq_k_1, Tk_1, Ta_1, Tb_1, dt_1, V_1);
q_p_2 = 0;          % pozycja początkowa przegubu nr 1 [rad]
dq_p_2 = 0.0;       % prędkość początkowa przegubu nr 1 [rad]
q_k_2 = 1/4 * pi;   % pozycja końcowa przegubu nr 1 [rad/s]
dq_k_2 = 0.0;       % prędkość końcowa przegubu nr 1 [rad/s]
Tk_2 = 10;          % Czas trwania ruchu przegubu [s]
Ta_2 = 4;           % Czas trwania fazy przyspieszania [s]
Tb_2 = 4;           % Czas trwania fazy hamowania [s]
dt_2 = 0.01;        % Krok czasowy [s]
V_2 = 0.05;          % Prędkość przegubu w fazie drugiej [rad/s]
[q_2,dq_2,ddq_2, t_2] = Trajectory_Generation(solved, q_p_2, dq_p_2, ...
      q_k_2, dq_k_2, Tk_2, Ta_2, Tb_2, dt_2, V_2);
q_1 = double(q_1); dq_1 = double(dq_1); ddq_1 = double(ddq_1); q_2 =
double(q_2); dq_2 = double(dq_2); ddq_2 = double(ddq_2);
m1 = 2.5;
m2 = 1.5;
```

```
L1 = 0.6;
L2 = 0.5;
[u] = dynamics_model_earth(q_1, q_2, dq_1, dq_2, ddq_1, ddq_2, m1, m2, L1,
L2);
u_1 = u(1, :);
u_2 = u(2, :);
figure;
subplot(2, 2, 1); plot(t_1, q_1, t_1(1), q_p_1, 'o', Tk_1, q_k_1, 'o')
legend('Pozycja', 'Pozycja początkowa', 'Pozycja końcowa',
"Location","best"); title('Pozycja przegub 1'); xlabel('t [s]');
ylabel('Pozycja kątowa [rad]'); grid on; axis auto
subplot(2, 2, 2); plot(t_1, u_1);
legend('Moment przegub 1', "Location","best"); title('Moment przegub 1')
xlabel('t [s]'); ylabel('Moment'); grid on; axis auto
subplot(2, 2, 3);; plot(t_2, q_2, t_2(1), q_p_2, 'o', Tk_2, q_k_2, 'o')
legend('Pozycja', 'Pozycja początkowa', 'Pozycja końcowa',
"Location","best"); title('Pozycja przegub 2'); xlabel('t [s]')
ylabel('Pozycja kątowa [rad]'); grid on; axis auto;
subplot(2, 2, 4); plot(t_2, u_2); legend('Moment przegub 2',
"Location","best"); title('Moment przegub 2'); xlabel('t [s]');
ylabel('Moment'); grid on; axis auto

p1 = 0; % p1  % manipulator mounting point in x axis
p2 = 0; % p2  % manipulator mounting point in y axis
m0 = 35; % m0  % the mass of the satellite
m1 = 2.5; % m1  % the mass of the first kinematic pair
m2 = 1.5; % m2  % the mass of the second kinematic pair
L1 = 0.6; % L1  % the length of the first link
L2 = 0.5; % L2  % the length of the second link
a1 = L1/2; % a1  % center of mass of the first link L1/2
a2 = L2/2; % a2  % center of mass of the second link L2/2
I0 = 1.5; % I0  % Satellite moment of inertia
I1 = (1/12) * m1 * L1^2; % I1  % Moment of inertia of the first kinematic
pair 1/12 mL^2
I2 = (1/12) * m2 * L2^2; % I2  % Moment of inertia of the second kinematic
pair

for index = 1:1:3
    m0 = [35, 70, 300];
    I0 = [1.5, 30, 100];
    text = "Parametry masowe: masa satelity = " + num2str(m0(index)) + "kg,
moment bezwładności satelity = " + num2str(I0(index)) + "kg*m^2";
    [M_matrix, C_matrix] = dynamics_matrix(p1, p2, m0(index), m1, m2, L1,
L2, a1, a2, I0(index), I1, I2);
    F_x = zeros(1, length(u));
    F_y = zeros(1, length(u));
    T_0 = zeros(1, length(u));
    Q = [F_x; F_y; T_0; u];
```

```
    h = 0.01;
    q0 = [0; 0; 0; 0; 0];
    dq0 = [0; 0; 0; 0; 0];
    [q_space, dq_space] = ode4(Q, q0, dq0, C_matrix, M_matrix, h);

    Figure; sgtitle(text); subplot(5, 1, 1); plot(t_1, q_space(4, :))
    legend("Manipulator na satelicie", "Location","best")
    title('Pozycja kątowa przegubu nr 1 manipulatora na satelicie')
    xlabel('t [s]'); ylabel('Pozycja [rad]'); grid on; axis auto
    subplot(5, 1, 2); plot(t_1, q_space(5, :));
    legend("Manipulator na satelicie", "Location","best")
    title('Pozycja kątowa przegubu nr 2 manipulatora na satelicie')
    xlabel('t [s]'); ylabel('Pozycja [rad]'); grid on; axis auto
    subplot(5, 1, 3); plot(t_1, q_space(1, :));
    title('Pozycja satelity na osi X');
    xlabel('t [s]'); ylabel('Pozycja [m]'); grid on; axis auto
    subplot(5, 1, 4); plot(t_1, q_space(2, :));
    title('Pozycja satelity na osi Y');
    xlabel('t [s]'); ylabel('Pozycja [m]'); grid on; axis auto
    subplot(5, 1, 5); plot(t_1, q_space(3, :));
    title('Orientacja satelity');
    xlabel('t [s]'); ylabel('Orientacja [rad]'); grid on; axis auto

    %%% Pozycja i orientacja końcówki manipulatora na satelicie
    [Pee_space, Psiee_space] = direct_2DoF_space([q_space(4, :);
 q_space(5, :)]', q_space(1, :)', q_space(2, :)', q_space(3, :)', L1, L2);

    figure; sgtitle(text);
    subplot(2, 1, 1); plot(Pee_space(:, 1), Pee_space(:, 2))
    legend("Manipulator na satelicie", "Location","best")
    title('Pozycja końcówki roboczej manipulatora na satelicie')
    xlabel('Pozycja x [m]'); ylabel('Pozycja y [m]');grid on; axis auto

    subplot(2, 1, 2); plot(t_1, Psiee_space)
    legend("Manipulator na satelicie", "Location","best")
    title('Orientacja końcówki roboczej manipulatora na satelicie')
    xlabel('t [s]'); ylabel('Orientacja [rad]'); grid on; axis auto

    %%% Porównanie do ziemskiego
    q0 = [0; 0];
    dq0 = [0; 0];

    u = double(u);
    [q_earth, dq_earth] = ode4_earth(u, q0, dq0, L1, L2, m1, m2);
    [Pee_earth, Psiee_earth, Vee_earth, Omee_earth] =
direct_2DoF_earth(q_earth', dq_earth', L1, L2);

    figure; sgtitle(text)
```

```
    subplot(2, 2, 1);
    hold on; plot(t_1, q_space(4, :)); plot(t_1, q_earth(1, :)); hold off
    legend("Manipulator na satelicie", 'Manipulator ziemski',
"Location","best")
    title('Porównanie pozycja kątowych przegubu nr 1')
    xlabel('t [s]'); ylabel('Pozycja kątowa[rad]');grid on; axis auto
    subplot(2, 2, 2)
    hold on; plot(t_1, q_space(5, :)); plot(t_1, q_earth(2, :)); hold off
    legend("Manipulator na satelicie", 'Manipulator ziemski',
"Location","best")
    title('Porównanie pozycja kątowych przegubu nr 2')
    xlabel('t [s]'); ylabel('Pozycja kątowa[rad]'); grid on; axis auto

    subplot(2, 2, 3)
    hold on; plot(t_1, Psiee_space); plot(t_1, Psiee_earth); hold off
    legend("Manipulator na satelicie",'Manipulator ziemski',
"Location","best")
    title('Porównanie orientacji końcówki roboczej manipulatora')
    xlabel('t [s]'); ylabel('Orientacja [rad]'); grid on; axis auto

    subplot(2, 2, 4)
    hold on; plot(Pee_space(:, 1), Pee_space(:, 2)); plot(Pee_earth(:, 1),
Pee_earth(:, 2)); hold off
    legend("Manipulator na satelicie", 'Manipulator ziemski',
"Location","best")
    title('Porównanie pozycji końcówki roboczej manipulatora')
    xlabel('Pozycja x [m]'); ylabel('Pozycja y [m]'); grid on; axis auto
end
```

10. Analizę otrzymanych wyników oraz wnioski końcowe.

Podczas tego ćwiczenia kluczowym elementem było porównanie wyników uzyskanych dla manipulatora ziemskiego z manipulatorem umieszczonym na satelicie. Przeprowadzenie symulacji dynamiki prostej umożliwiło obserwację zachowania całego układu dla różnych parametrów masowych satelity Wyznaczenie sił działających na satelitę z zasady zachowania pędu jest kluczowym elementem podczas projektowania układu ze sterowaną bazą. Znajomość tych sił umożliwia dobór sterowania silnikami pozycjonującymi bazę tak aby pozostała ona nieruchoma.

Warto również zwrócić uwagę na różnicę w równaniach Lagrange’a drugiego rodzaju dla manipulatora ziemskiego oraz umieszczonego na satelicie. Wyznaczanie macierzy M oraz C bez pomocy metod numerycznych byłoby z pewnością zadaniem bardzo czasochłonnym, a czasami wręcz niemożliwym.