Politechnika Warszawska

Raport Praca Domowa 1

Dynamika układów wieloczłonowych

inż. Weronika Biszczat 303296

Tomasz Niedziałkowski 313576

Maksymilian Łuc 313811

Prowadzący: mgr inż. Paweł Maciąg

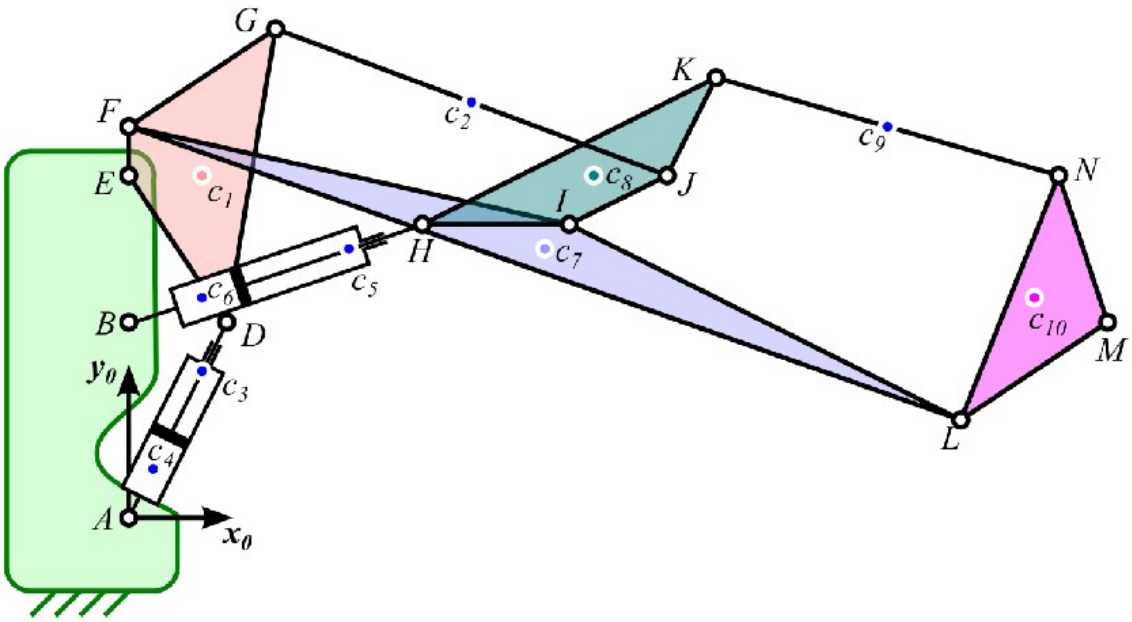
Data oddania: **15 maja 2023**

Spis treści

Spis rysunków

1 Wstęp

Zadanie zawarte w pracy domowej 1 polegało na przeprowadzeniu analizy kinematycznej mechanizmu przedstawionego na rysunku 1.



Rysunek 1: Schemat kinematyczny mechanizmu

W tym celu należało:

- zbudować model w programie Adams i zasymulować pracę mechanizmu,
- przygotować skrypt MATLAB do przeprowadzenia analizy kinematycznej mechanizmu,
- wykonane prace opisać w zbiorczym raporcie.

2 Program Matlab

2.1 Wymagania programu

Skrypt w programie MATLAB ma spełniać następujące wymagania:

- wymiary mechanizmu należy przyjąć zgodnie z danymi podanymi na rysunku 1,
- lokalne układy odniesienia należy umieścić w środkach mas członów (punkty c_i),
- założyć, że zależność długości siłownika k $l_k(t)$ ma postać:

$$l_k(t) = l_{0k} + a_k sin(\omega_k t + \phi_k)$$

  gdzie stałe l_k, a_k, ω_k i ϕ_k które należy dobrać samodzielnie,

- należy napisać program, który na żądanie obliczy przebiegi położeń, prędkości i przyspieszeń liniowych dowolnego punktu mechanizmu, a także prędkości i przyspieszenia kątowe dowolnego członu,
- przedział czasu, krok tablicowania obliczeń oraz wymaganą dokładność obliczeń należy przyjąć samodzielnie,
- program powinien wykrywać osobliwość macierzy Jacobiego i sygnalizować ją użytkownikowi,
- w programie należy umieścić komentarze informujące o sposobie jego obsługi i wyjaśniające wykonywane operacje.

2.2 Struktura programu

Głownym skryptem, który łączy wszystkie podprogramy jest ten o nazwie: *Main.m*. Wywołuje on kolejne podprogamy:

- *LoadData.m* - odpowiedzialny za wczytanie danych zawartych w plikach csv
- *Solve.m* - odpowiedzialny za wykonanie wszystkich niezbędnych obliczeń
- *PointInfo.m* - wykreślający otrzymane wyniki

2.3 Main.m

Program *Main.m*, jak wspomniano, łączy ze sobą wszystkie funkcjonalności wymienionych podprogramów. Zadania programu prezentują się następująco:

- stworzenie wektora opisującego przedział czasowy, dla którego mają zostać wykonane obliczenia
- wywołanie podprogramu *LoadData.m* i zapisanie danych w tablicy
- wywołanie podprogramu *Solve.m* i zapisanie wyników (położenia, prędkości, przyspieszenia punktów) w tablicy
- wywołanie podprogramu *PointInfo.m*

2.4 LoadData.m

Skrypt LoadData.m wczytuje pięc plików CSV zawierających dane opisujące geometrię układu (w nawiasach podano nazwy kolumn w odpowiednich plikach):

- points.csv – zawiera współrzędne charakterystycznych punktów mechanizmu w układzie globalnym (X,Y),
- objects.csv – zawiera współrzędne środków mas członów w układzie globalnym (X,Y),
- bonds.csv – w tym pliku znajdują się informacje o parach obrotowych, pomiędzy członami. Kolejno zapisywane jest indeks członu pierwszego, indeks członu drugiego i indeks punktu w którym ma powstać para (ObjA,ObjB,Point),
- constrains.csv – informacje o parach obrotowych połączonych z podstawą. Dane zapisane są w sposób analogiczny do pliku bonds.csv (ObjA,ObjB,Point),
- drives.csv – opisuje siłowniki, a więc pary postępowe wraz z więzami kierującymi. Zapisujemy w nim kolejno: środek członu pierwszego, środek członu drugiego, punkt początkowy siłownika, punkt końcowy siłownika oraz kolejno stałe l, a, ω, ϕ opisujące wzór na długość siłownika.

2.5 Solve.m

Program pobiera nastepujące argumenty:

- wektor t – przedział czasowy wykonywania obliczeń,
- points – współrzędne więzów mechanizmu,
- macierz objects - zawierającą położenia środków ciężkości członów mechanizmu,
- bonds – macierz opisującą pary kinematyczne człon-człon,
- contrains – macierz opisującą pary kinematyczne człon-podłoże,
- drives – macierz opisującą napędzane pary postępowe.

Program na początku tworzy wskaźniki na funkcje odpowiedzialne za kolejno:

- FIK – wyznaczenie macierzy Φ^K,
- FID – wyznaczenie macierzy Φ^D,
- FId – wyznaczenie macierzy Jakobiego,
- FIt – wyznaczenie macierzy pochodnej równań więzów względem czasu,
- Gamma – wyznaczenie macierzy Γ.

Następnie definiowany jest wektor przybliżenia początkowego, przy użyciu danych zawartych w plikach csv. Kolejnym krokiem jest zdefiniowanie macierzy, do których zapisywane będą obliczone położenia, prędkości i przyspieszenia. Ostatnim krokiem w działaniu programu jest wyznaczenie poszukiwanych wartości dla każdej chwili czasowej zdefiniowanego przedziału czasowego. Obliczenia te wykonywane są w pętli przechodzącej do kolejnych chwil. Należy teraz zagłębić się w budowę poszczególnych elementów programu właśnie wymienionych.

2.5.1 FIK

Zadaniem funkcji jest wyznaczenie macierzy Φ^K. Badany układ składa się z par obrotowych i par postępowych, które nalezy opisać. W tym projekcie zostało przyjęte, że w funkcji FIK zostaną opisane tylko pary obrotowe, natomiast pary postępowe i ich macierze Φ^K oraz Φ^D zostaną wyznaczone w funkcji FID. Może być to nieco mylące, że elementy macierzy Φ^K wyznaczane są w funkcji o nazwie FID, jednak chcieliśmy zachować wszystkie elementy dotyczące par postępowych w jednej funkcji. Takie rozwiązanie nie sprawia problemow, gdyż na koniec macierze Φ^K oraz Φ^D są łaczone we wspólną macierz Φ.

Tak, jak w każdej funkcji na początek tworzona jest macierz przechowująca zwracany wynik. Następnie iterujemy po kolejnych parach opisanych w pliku bonds i zapisujemy ich równania do odpowiednich komórek macierzy Φ^K według wzóru 2.18 ze skryptu zajęć wykładowych. Wektory r wyznaczane są za pomocą funkcji getXY, która jako argument przyjmuje wektor położeń oraz indeks interesującego nas członu, który to z kolei zapisany jest w pliku bonds. Macierz rotacji wyznaczana jest za pomocą funkcji RMatrix, która działa na podobnej zasadzie co poprzednio wspomniana funkcja. Różnicą jest to, że funkcja RMatrix zwraca macierz rotacji wykorzystując kąt orientacji lokalnego układu współrzędnych. Jako ostatnie wyznaczane są wektory prowadzące ze środów lokalnych układów współrzędnych do punktu połączenia członów. Jest to realizowane za pomocą prostego odejmowania wektorów, gdyż pozycje punktu połączenia, jak i lokalnego układu współrzędnych, wywoływane są przy użyciu indeksów identyfikujących te punkty.

Kolejna pętla wykonuje te same operacje, tylko na parach kinematycznych człon-podłoże. W tym przypadku wektor s_b jest zerowy, dlatego obliczane równanie jest krótsze.

2.5.2 FID

Następnie wyznaczane są elementy macierzy Φ^K opisujące parę postępową oraz macierz Φ^D. Macierz przechowująca wyniki jest odpowiednio większa. Każdy element opisujący pojedynczy więz w macierzy Φ^K miał wymiar 2x1, tutaj element opisujący pojedynczy więz ma wymiar 3x1, gdyż łączona jest tutaj macierz Φ^K z Φ^D. Stosowane są tutaj również inne wzory tj. 2.19 oraz 2.28, ponieważ jest to para postępowa.

2.5.3 FId

Ta funkcja wyznacza macierz jakobiego. przyjmuje standardowe argumenty, takie jak poprzednie funkcje. Jej wymiary to 3 * liczba członów x 3 * liczba członów. Tutaj również, analizowane są w osobnych pętlach pary obrotowe człon-człon, pary obrotowe człon-podłoże oraz pary postępowe. Macierz dzieli się na część kinematyczną i dynamiczną. Część kinematyczna opisująca pary obrotowe wykorzystuje wzory od 2.31 do 2.34, z uwzględnieniem zerowego wektora s_b w parach obrotowych człon-podłoże. Po uzupełnieniu informacji i członach obrotowych, opisywane są człony postępowe. Część Φ^K wykorzystuje wzory od 2.43 do 2.46 oraz 2.49 do 2.52. Elementy macierzy jakobiego uzupełniane są w kolejności w jakiej były wypisywane w tym akapicie.

2.5.4 FIt

W tym punkcie wyznaczana jest pochodna po czasie macierzy Φ. Jej składowa Φ^K nie jest zależna od czasu, dlatego odpowiadające jej pola mają wartość 0. Do obliczenia pochodnej funkcji opisują-

cych wysunięcie tłoków, wykorzystywana jest funkcja df. Struktura funkcji jest znana. Z pliku csv pobierane są tylko wartości liczbowe, dlatego pochodne zostały wyznaczone ręcznie i uzupełniane są o wartości liczbowe.

2.5.5 Gamma

Macierz Γ potrzebna do wyznaczenia przyspieszenia wyznaczana jest w kolejności zgodnej z kolejnością z poprzednich funkcji (pary obrotowe człon-człon, pary obrotowe człon-podłoże, pary postępowe). Wykorzystywane wzory to 2.42, 2.59 oraz 2.71. Wzór 2.71 uwzględnia przesunięcie tłoka zadaną funkcją.

2.5.6 NR

Funkcja zawiera proces wyznaczania wektora położeń za pomocą metody Newtona-Rhapsona. Definiowana jest maksymalna liczba iteracji oraz precyzja obliczeń. Wczytywane są wektory parametrów początkowych oraz wyznaczana jest macierz Φ. Następnie obliczenia wektora położeń w kolejnym kroku wyznaczane są do momentu, kiedy nie zostanie osiągnięta oczekiwana dokładność lub zostanie wykonana maksymalna liczba iteracji. Funkcja jest analogiczna to tej, zawartej w materiałach wykładowych. Po każdym wyznaczeniu wektora położeń, sprawdzana jest niezależność więzów/ osobliwość jakobianu. Jeżeli zostanie ona wykryta to program jest zatrzymywany. Sprawdzanie polega na obliczeniu wyznacznika macierzy jakobianu.

2.5.7 Pozostałe funkcje

Program został wyposażony w pomocnicze funkcje, odpowiedzialne za m.in. wyznaczenie wersora prostopadłego lub stycznego między wybranymi punktami. Funkcje tę są krótkie, dlatego w tym raporcie ich omówienie zostanie pominięte. Stosowne komentarze można znaleść w kodzie źródłowym.

2.6 PointInfo.m

Program pobiera nastepujące argumenty:

- wektor t - przedział czasowy wykonywania obliczeń
- macierz q - zawierającą położenia środków ciężkości członów mechanizmu
- macierz q_prim - zawierającą prędkości środków ciężkości członów mechanizmu
- macierz q_wtor - zawierającą przyspieszenia środków ciężkości członów mechanizmu
- macierz objects - zawierającą położenia środków ciężkości członów mechanizmu
- obj - numer członu, którego położenie ma zostać wyświetlone lub do którego należy inny punkt, który ma zostać wyświetlony
- points - współrzędne więzów mechanizmu
- point - punkt którego położenie itd. ma zostać wyświetlone

Program bierze pod uwagę liczbę argumentów zapewnionych przy wywoływaniu fukncji. Kiedy liczba argumentów wynosić będzie 8, wyświetlone zostaną informacje dotyczące wybranego punktu, który należy do obiektu, który został również podany jako argument. Jeżeli liczba argumentów będzie wynosić 6, to program wyswietli informacje dotyczące środka ciężkości wybranego obiektu. Tworzone są tablice, uzupełniane o położenia, prędkości oraz przyspieszenia wybranego punktu w każej chwili czasowej, zawartej w zdefiniowanym przedziale t.
Program został wysposażony w funkcje odpowiedzialne za pobranie z macierzy q, q_prim oraz q_wtor danych dotyczących wybranych punktów. Komentarze opisujące każdy element kodu znajadują się w odpowiednich plikach załączonych do raportu.

3 Model Adams

Przyjęte zostały następujące funkcje długości siłownika.

Długość siłownika $l_1(t)$ łączącego punkty A i D:

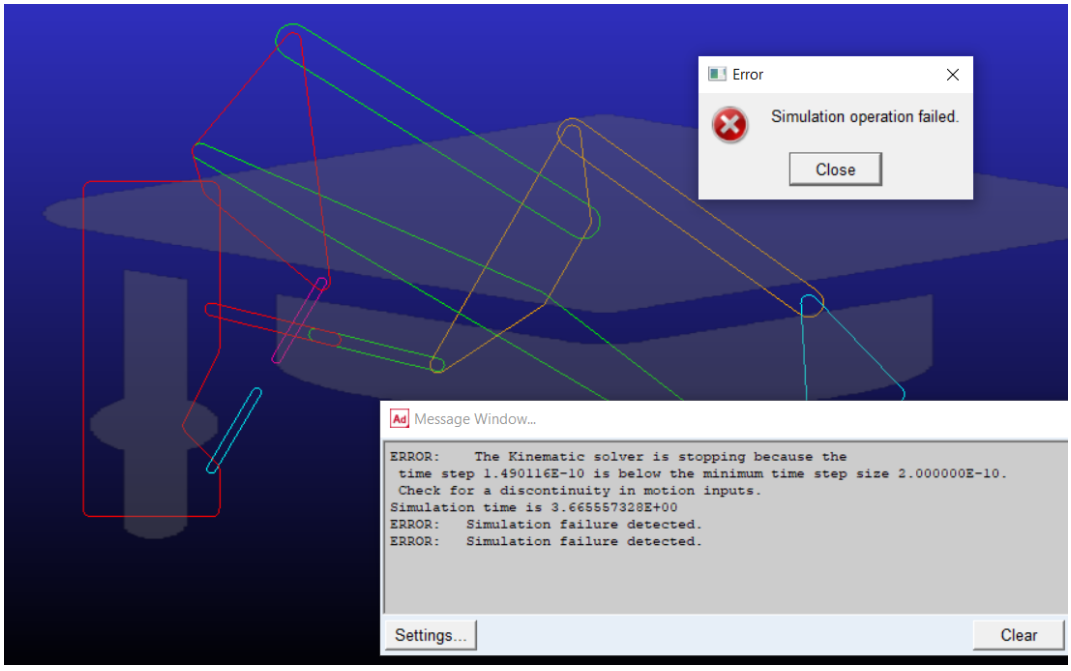
$$l_1 = l_{01} + a_1 sin(\omega_1 t + \phi_1) = 0.4472 + 0.05 \cdot sin(0.5 \cdot t + 0.0)$$

Długość siłownika $l_2(t)$ łączącego punkty B i H:

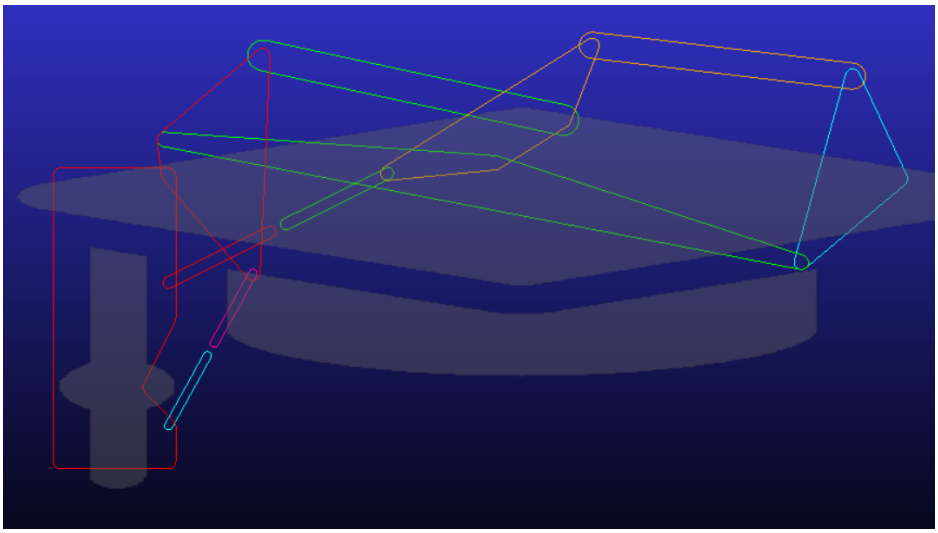
$$l_2 = l_{02} + a_2 sin(\omega_2 t + \phi_2) = 0.6324 + 0.05 \cdot sin(1.0 \cdot t + 0.0)$$

Nietypowe długości spoczynkowe siłowników (l_{01} i l_{02}) wynikają z przyjętego założenia, że początkowa konfiguracja mechanizmu będzie dokładnie taka, jak na rysunku 1.

Długości siłowników są ograniczone zależnościami geometrycznymi, po przekroczeniu dozwolonych przedziałów układ staje się nieliniowy i niemożliwe jest uzyskanie zbieżności wyników. Przykłady napotkanych błędów są widoczne na rysunkach 2,3 oraz 4.



Rysunek 2: Błąd symulacji dla wartości współczynnika a=0.1.



Rysunek 3: Błąd symulacji dla wartości współczynnika a=0.2.

Editor - C:\Users\moskit\Desktop\duw zapasowe\duw_pd1-main\pd1\DUW\PD1\Main.m

Main.m | PointInfo.m | Solve.m | LoadData.m

```
1   clc; clear;
2
3   %określenie przedziału czasowego dla którego mają zostać wykonane
4   %obliczenia
5   t= linspace(0,15,501);
6
7   %wczytanie danych dotyczących konstrukcji modelu z plików csv
8   [points,objects,bonds,contrains,drives] = LoadData();
9
10  %obliczenie położeń, prędkości i przyspieszeń środków ciężkości członów
11  [q,q_prim,q_wtor] = Solve(t,points,objects,bonds,contrains,drives);
12
13  %wyświetlenie wykresów położeń, prędkości oraz przyspieszeń wybranego
14  %punktu wybranego członu lub środka ciężkości
15  [XY, V, A] = PointInfo(t,q,q_prim,q_wtor,objects,8,points,8);
16  %PointInfo(t,q,q_prim,q_wtor,objects,1)
```

Command Window

```
BŁĄD: Po 10000 iteracjach Newtona-Raphsona nie uzyskano zbieżności
BŁĄD: Po 10000 iteracjach Newtona-Raphsona nie uzyskano zbieżności
BŁĄD: Po 10000 iteracjach Newtona-Raphsona nie uzyskano zbieżności
```

Rysunek 4: Błąd obliczeń programu.

W przypadku siłownika rozpiętego pomiędzy punktami A i D, obliczenie maksymalnego wydłużenia jest możliwe w prosty sposób korzystając z podstawowej zależności na długości boków trójkąta. Znając stałą odległość punktów |AE|=0.7 oraz |ED|=0.36 otrzymujemy minimalną długość odległości punktów |AD|=0.34. Spoczynkowa odległość wynosi |AD|=0.45. Odejmując od wartości spoczynkowej wartość minimalną otrzymujemy możliwą wartość współczynnika wynoszącą a=0.11.

Obliczenie metodą analityczną długości drugiego siłownika jest niestety bardziej skomplikowane, dlatego najlepszą metodą na wyznaczenie wartości współczynnika jest sposób doświadczalny. W celu w podprogramie LoadData.m należy w pliku drives.csv dla siłownika pierwszego wpisać wartość współczynnika równą 0.0. Sprawia to, że siłownik ten jest unieruchomiony i pozwala na sprawdzenie jakie wartości współczynnika dla siłownika drugiego są zgodne z geometrią. W ten sposób sprawdzono kolejne wartości począwszy od 0.2. Największą możliwą do uzyskania wartością współczynnika dla siłownika drugiego było a=0.05. Wartość tą wykorzystano przy prezentacji wyników.

4 Podsumowanie

W pracy domowej 1 udało się zamodelować zadany mechanizm z wykorzystaniem programu Adams, a także zrealizować skrypt obliczający kinematykę mechanizmu dla tak danego, 2-wymiarowego układu. Wyniki otrzymane z oprogramowania komercyjnego pokrywają się z tymi z przygotowanego skryptu.