

imię i nazwisko dyplomanta

nr albumu: 31284

kierunek studiów: Mechatronika

specjalność: układy mechatroniczne

forma studiów: stacjonarne

**Projekt konstrukcyjny stanowiska dydaktycznego symulującego pracę suwnicy
bramowej.**

praca dyplomowa inżynierska/magisterska

napisana pod kierunkiem:
dr. hab. inż. Bartosz Powalka, prof. ZUT,

Instytut Technologii Mechanicznej

Data wydania tematu pracy:

Data złożenia pracy:

Szczecin, 2018

OŚWIADCZENIE AUTORA PRACY DYPLOMOWEJ

Oświadczam, że praca dyplomowa ~~inżynierska~~/inżynierska/~~magisterska~~ (podać rodzaj pracy) pn.

Projekt konstrukcyjny stanowiska dydaktycznego symulującego pracę suwnicy bramowej.

(temat pracy dyplomowej)

napisana pod kierunkiem:

dr. hab. inż. Bartosz Powalka, prof. ZUT

(tytuł lub stopień naukowy imię i nazwisko opiekuna pracy)

jest w całości moim samodzielnym autorskim opracowaniem sporządzonym przy wykorzystaniu wykazanej w pracy literatury przedmiotu i materiałów źródłowych.

Złożona w dziekanacie Wydziału

Inżynierii Mechanicznej i Mechatroniki

(wydział)

treść mojej pracy dyplomowej w formie elektronicznej jest zgodna z treścią w formie ~~pisemnej~~/pisemnej i graficznej*.

Oświadczam ponadto, że złożona w dziekanacie praca dyplomowa ani jej fragmenty nie były wcześniej przedmiotem procedur procesu dyplomowania związanych z uzyskaniem tytułu zawodowego w uczelniach wyższych.

.....
podpis dyplomanta

Szczecin, dn.

* niepotrzebne skreślić

Streszczenie

Niniejsza praca dyplomowa opisuje konstrukcyjny proces tworzenia modelu suwnicy bramowej, która w przyszłości będzie wykorzystywana w celach dydaktycznych na Wydziale Inżynierii Mechanicznej i Mechatroniki. Praca dyplomowa przedstawia kolejne etapy konstruowania suwnicy bramowej, czyli: zdefiniowanie założeń projektowych, wstępne zaprojektowanie modelu, wykonanie obliczeń wytrzymałościowych, dobranie elementów wykonawczych oraz sterujących, wykonanie algorytmu sterującego i zgodnie z nim zaprogramowanie urządzeń sterujących. Po montażu modelu, zostały wykonane na nim testy mające na celu ustalić, czy suwnica spełnia początkowe założenia projektowe: czy suwnica osiągnie założoną prędkość początkową oraz czy przy obciążeniu suwnicy ugięcie będzie mniejsze niż zdefiniowane przy projektowaniu.

Słowa kluczowe

Suwnica bramowa, model, serwomechanizm, aluminiowy profil konstrukcyjny, Arduino, Pololu Micromaestro.

Abstract

The engineering thesis presents the mechanical design of gantry crane model, later to be used as a didactic tool in the Institute of Mechanical Engineering and Mechatronics. The study describes each stage of designing the model: initial design assumptions, strength calculations, the selection of drivers and executive devices, the creation of control algorithm, according to which the drivers were programmed. Following the assembly of the model are tests which aim to conclude whether the crane met the initial design assumptions or not. These assumptions are: reaching specified average speed over a certain distance and deflection created by applying a load should be lower than the value specified at the beginning of design process.

Keywords

Gantry crane, model, servo, aluminium construction profile, Arduino, Pololu Micromaestro.

Spis treści

Rozdział 1. Wstęp.....	7
Rozdział 2. Założenia projektowe.....	10
Rozdział 3. Dobór konstrukcji.....	12
3.1. Zdefiniowanie zadania do rozpatrzenia.....	12
3.2. Obliczenia ugięcia belki	12
3.3. Obliczenie sił tnących w belce	15
3.4. Otrzymanie wykresów sił tnących	16
3.5. Obliczenie momentów gnących w belce	17
3.6. Otrzymanie wykresów momentów gnących w belce	18
3.7. Obliczenie ugięcia belki	19
3.8. Otrzymanie wykresów ugięcia belki	20
3.9. Wybór aluminiowego profilu konstrukcyjnego	21
3.10. Sprawdzenie poprawności wyników	21
3.11. Analiza statyczna konstrukcji suwnicy bramowej	22
Rozdział 4. Dobór elementów wykonawczych.....	27
4.1. Dobór silników napędowych.....	27
4.2. Obliczanie wartości momentów napędowych.....	29
Rozdział 5. Algorytm sterowania.....	30
5.1. Aplikacja w Matlab	31
5.2. Program w Arduino	31
5.3. Program w Pololu Micromaestro	33
Rozdział 6. Testy.	34
6.1. Pomiar prędkości.....	35
6.2. Uzyskanie wyniku pomiaru prędkości wraz z jego niepewnością	35
6.3. Pomiar Ugięcia	36
6.4. Uzyskanie wyniku pomiaru ugięcia wraz z jego niepewnością	36
Rozdział 7. Zakończenie.	38
Bibliografia.	39
Dodatek A Rysunki Techniczne.....	40
Dodatek B Programy użyte w pracy inżynierskiej.	41
Spis rysunków i spis tabel.....	55

Wprowadzenie

Celem pracy dyplomowej jest skonstruowanie modelu suwnicy bramowej, w celu późniejszego wykorzystania jej do celów dydaktycznych. W pierwszym rozdziale przybliżono ogólny podział suwnic. Drugi rozdział opisuje wszelkie założenia projektowe, które zostały przyjęte przed rozpoczęciem konstruowania modelu. Trzeci rozdział poświęcony jest obliczeniom związanym z doбором profilu konstrukcyjnego wykorzystanego do późniejszego montażu suwnicy. W czwartym rozdziale opisano obliczenia związane z doбором elementów wykonawczych: serwomechanizmów napędzających oś Y i oś X modelu. Piąty rozdział opisuje algorytm sterowania wraz z programem używanym w sterownikach. Szósty rozdział poświęcony jest testom wykonanym w celu potwierdzenia, że model spełnia wcześniej przyjęte założenia projektowe.

Rozdział 1

Wstęp.

Celem poniższej pracy inżynierskiej jest zaprojektowanie i skonstruowanie modelu suwnicy bramowej, która będzie podstawowym elementem w stworzeniu stanowiska badawczego zajmującego się analizą kinematyki i dynamiki suwnic bramowych. W przyszłości będzie możliwe rozbudowanie modelu i wyposażenie go w odpowiednie czujniki, kamery, siłowniki. Umożliwi to coraz bardziej dokładną i głębszą analizę problemów związanych z ruchem suwnic bramowych.

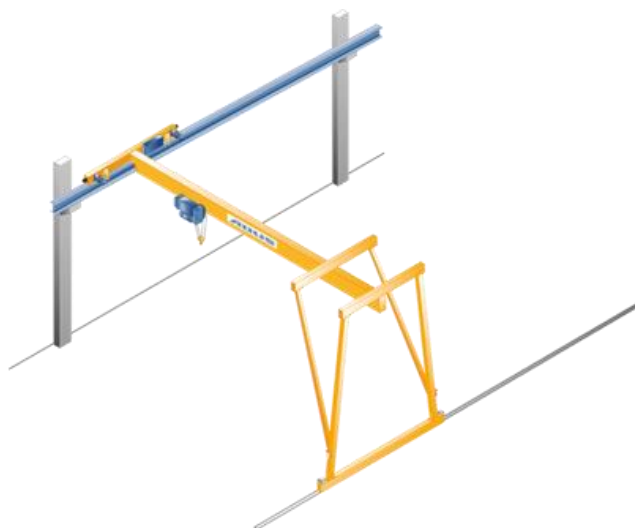
Dźwignica z definicji jest to *‘maszyna robocza pracująca ruchem przerywanym, służąca do przemieszczania na ograniczone odległości różnorodnych ładunków w obrębie danego ośrodka w związku z procesem technologicznym lub usługami przeładunkowymi.’* (Źródło, <https://sjp.pwn.pl/slowniki/d%C5%BAwignica.html>). Podział dźwignic jest bardzo szeroki: suwnice, wciągarki, żurawie itd., aczkolwiek dokładniej rozpatrzę tylko rodzaje suwnic, gdyż jest to tematem mojej pracy.



Rysunek 1.1. Suwnica bramowa. [8]

Suwnica bramowa – jest to rodzaj suwnicy, w którym belkę lub belki dźwigarowe, wraz z zestawem roboczym, opierają się z obu stron na konstrukcji nośnej. Cała konstrukcja znajduje się na zestawie jezdnym, przy czym nie każdy musi być napędzany. Często suwnice poruszają się po wcześniej skonstruowanym torze, najczęściej szynach kolejowych, umożliwia to bardzo szeroki zakres ruchów suwnicy w jednej osi – ogranicza je tylko długość wcześniej skonstruowanego toru. Najczęściej suwnice bramowe mają zastosowanie w przemyśle morskim, ze względu na dużą nośność oraz rozmiar swojego obszaru roboczego.

Zaletą suwnicy bramowej jest możliwość osiągnięcia największych nośności ze wszystkich innych rodzajów suwnic, małe ograniczenia związane z miejscem użytkowania suwnicy, duży asortyment urządzeń roboczych dla takiej suwnicy. Wyraźną wadą jest jednak trudność w wykonaniu oraz długi czas montażu i produkcji.



Rysunek 1.2. Suwnica półbramowa. [2]

Suwnica półbramowa – jest to rodzaj suwnicy, w którym jedna strona belki lub belek dźwigarowych opiera się na konstrukcji – na przykład belce wspornikowej zamocowanej przy ścianie hali technologicznej, podczas gdy drugą stronę podtrzymuje konstrukcja nośna, która porusza się wraz z jej zestawem jezdny po wcześniej przygotowanym torze. Ze względu na swoją konstrukcję, suwnica półbramowa nie jest tak uniwersalna jak suwnica bramowa. Używana jest w magazynach lub halach produkcyjnych, ze względu na potrzebę pomocniczej konstrukcji nośnej, ale dzięki temu jest ona bardziej kompaktowa niż suwnica bramowa. Zaletą tej suwnicy jest możliwość zamontowania jej poniżej zestawu suwnic głównych oraz skrócony montaż takiej suwnicy. Wadą zaś jest wymagana pomocnicza konstrukcja nośna w postaci belki jezdnej.



Rysunek 1.3. Suwnica Pomostowa. [9]

Suwnica pomostowa – jest to rodzaj suwnicy, której belka lub belki dźwigarowe opierają się bezpośrednio na konstrukcji nośnej. Używana często jest w halach produkcyjnych, ponieważ jej niska budowa pozwala na zamontowanie jej w zadanej hali, co umożliwia większą wysokość pracy

suwnicy. Suwnice pomostowe możemy podzielić na: jednoźwigarowe oraz dwuźwigarowe. Ilość belek dźwigarowych wpływa na nośność całej suwnicy oraz na rodzaj wózka jezdnego w danej suwnicy. Zaletą suwnic tego rodzaju jest możliwość montowania ich w małej odległości od sufitu hali technologicznej, duży udźwig oraz większe niż w innych suwnicach prędkości jazdy suwnicy oraz wciągarki.



Rysunek 1.4. Suwnica wysięgnikowa. [10]

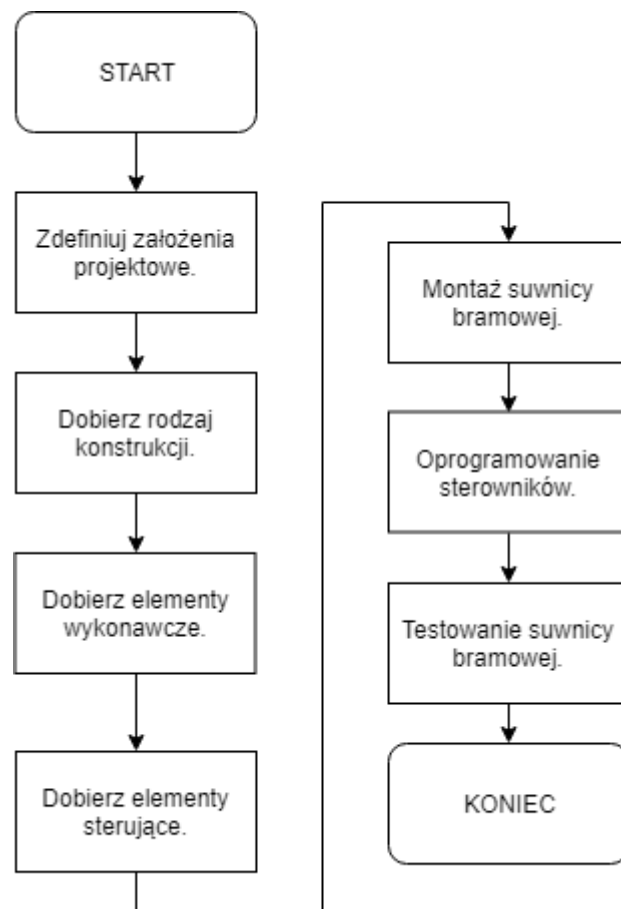
Suwnica konsolowa lub **wysięgnikowa** – jest to rodzaj suwnicy, której konstrukcja nośna znajduje się tylko po jednej stronie suwnicy. Drugi koniec belki dźwigarowej nie posiada żadnego podparcia. Cechą takiej suwnicy jest kompaktowość oraz możliwość umieszczenia takiej suwnicy na zewnątrz hali produkcyjnej bez wymogu budowania dodatkowej konstrukcji wspornikowej. Wadą tej konstrukcji jest mniejsza nośność i rozpiętość suwnicy, przy zachowaniu stosunkowo małych wartości wytrzymałościowych.

Główną przyczyną wyboru suwnicy bramowej jest jej uniwersalność i najmniejsze ograniczenia dotyczące obszaru roboczego. Umożliwia to używanie modelu w wielu celach i pozwala na badanie różnych zagadnień związanych z kinematyką i dynamiką suwnic bramowych. Wyniki badań otrzymanych na tym modelu umożliwią ich aproksymację na innych rodzajach suwnic.

Rozdział 2

Założenia projektowe.

W celu zaprojektowania modelu suwnicy bramowej postępowano zgodnie z zaprezentowanym poniżej algorytmem.



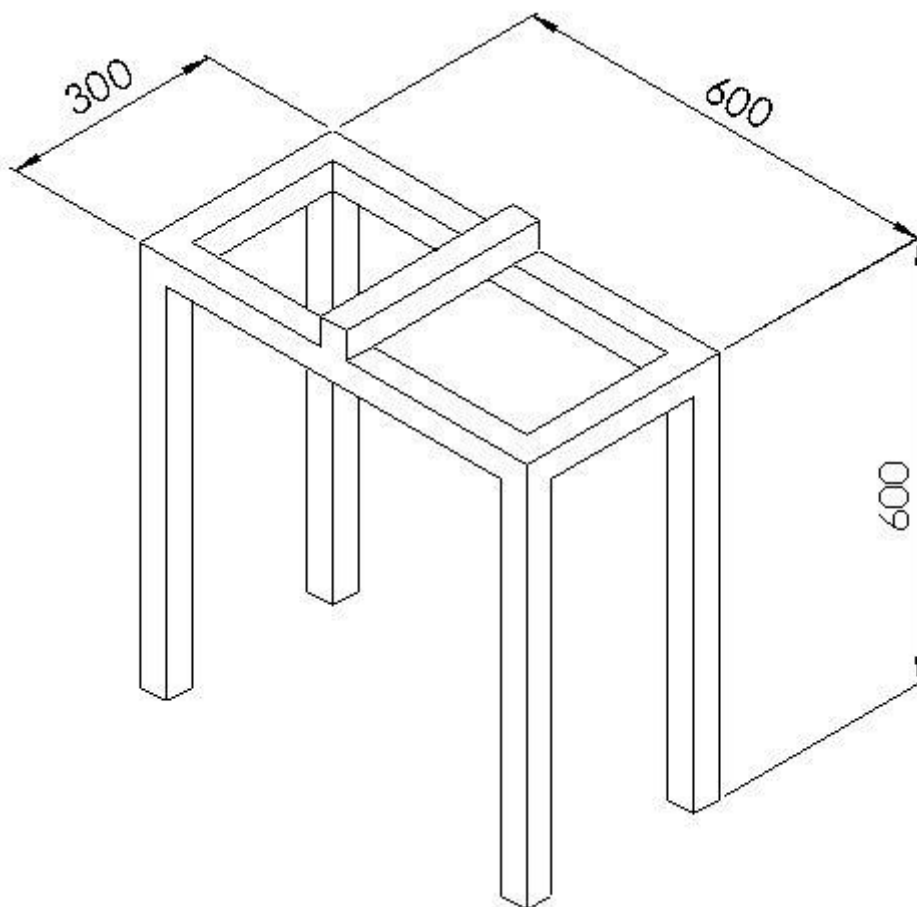
Rysunek 2.1. Algorytm konstruowania modelu suwnicy bramowej.

Pierwszym krokiem w celu zaprojektowania modelu suwnicy bramowej było zdefiniowanie pewnych założeń projektowych, które powinny zostać osiągnięte wraz z zakończeniem całego projektu. Głównymi założeniami były: wymiary modelu, prędkość jaką suwnica ma osiągać przy pracy oraz maksymalny udźwig modelu.

Proces projektowania suwnicy jest procesem iteracyjnym, w którym każdy krok jest powtarzany wielokrotnie, w celu wyeliminowania błędów, albo poprawienia parametrów suwnicy.

Każdy z powyższych kroków może być wykonywany jednocześnie w celu skrócenia czasu projektowania i konstruowania całego modelu.

Wymiary gabarytowe suwnicy powinny być zbliżone tych zaprezentowanych w schemacie poniżej:



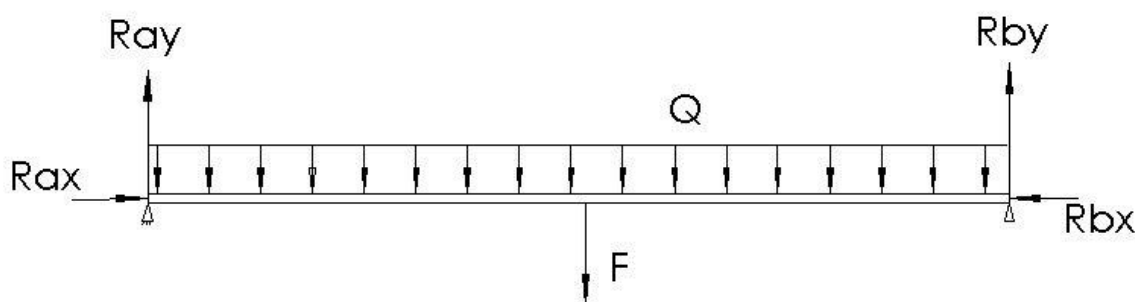
Rysunek 2.2. Założone wymiary gabarytowe modelu.

Wstępnie przyjęto, że suwnica powinna być w stanie osiągnąć prędkość $5 \left[\frac{m}{min} \right]$. Wartość ta została przyjęta, ponieważ suwnice bramowe mogą osiągać prędkości do $0,5 \left[\frac{m}{s} \right]$, a model jest około 10 razy mniejszy niż prawdziwa suwnica bramowa. Suwnica powinna osiągać prędkość maksymalną w czasie mniejszym niż $0,5[s]$. Maksymalny ciężar, zamontowanym na modelu powinien być równy 10 [kg], zwiększenie ciężaru dopuszczanego na modelu nie zwiększyłoby znacząco jego funkcjonalności, a jedynie utrudniło obsługiwanie go operatorowi.

Dobór konstrukcji.

3.1. Zdefiniowane zadania do rozpatrzenia.

Przy doborze konstrukcji pierwszym krokiem było ustalenie materiału, który powinien zostać użyty do budowy modelu. W celu ustalenia najlepszego przekroju profilu obliczona została belka o długości 500 mm z siłą przyłożoną na środku jej długości oraz oparta na dwóch podporach. Schemat poniżej przedstawia wcześniej opisany przypadek belki.



Rysunek 3.1. Schemat przedstawiający belkę użytą do ustalenia profilu zastosowanego do konstrukcji modelu.

Powyższa belka, służy zdeterminowaniu, który z dwóch dostępnych profili na uczelni zostanie użyty do konstrukcji suwnicy bramowej – aluminiowy profil konstrukcyjny 40x40 o szerokości rowków 8mm lub kwadratowa rura aluminiowa o wymiarach 40x40x2. Głównym kryterium doboru profilu jest ugięcie belki.

3.2. Obliczenie ugięcia belki.

W celu obliczenia ugięcia belki należało wyznaczyć reakcje w podporach belki, przy czym można zauważyć że w powyższym uproszczonym modelu belki nie występują siły w osi X więc reakcje obu podpór w tej osi są równe zero. Z równań statyki można wyznaczyć równania sił na osi Y oraz momentów sił względem punktu znajdującego się w podporze, co pozwoli nam obliczyć reakcje występujące w obu podporach.

Równanie sił w osi Y:

$$\sum F_x = 0 \quad (3.1)$$

$$\sum F_y = 0 \quad (3.2)$$

Gdzie:

- F_y – siły działające w osi Y. [N]

Podstawiając siły do wzoru (3.1) otrzymujemy równanie sumy sił na osi X:

$$R_{ax} - R_{bx} = 0 \quad (3.3)$$

Gdzie:

- R_{ax} – reakcja w podporze A w osi X. [N]
- R_{bx} – reakcja w podporze B w osi X. [N]

Przy uproszczonym schemacie belki siły w osi X się zerują, ze względu na to że nie występuje żadna inna siła w poziomej.

Podstawiając siły działające w osi Y do równania (1.2) otrzymujemy:

$$R_{ay} + R_{by} - F - Q \cdot l = 0 \quad (3.4)$$

$$Q = \rho \cdot V \quad (3.5)$$

,gdzie:

- R_{ay} – reakcja podpory A w osi Y [N]
- R_{by} – reakcja podpory B w osi Y [N]
- F – siła skupiona, która równa się ciężarowi zawieszonemu na suwnicy [N]
- Q – siłą ciągłą; masa całej belki [N/m]
- ρ – gęstość aluminium = $2720 \left[\frac{kg}{m^3} \right]$
- V – objętość belki [m^3]

Przekształcając wzór (3.4) w celu wyznaczenia reakcji w podporze A otrzymujemy:

$$R_{ay} = -R_{by} + F + Q \cdot l \quad (3.6)$$

Do wyznaczenia R_{ay} potrzebne jest znalezienie wartości reakcji w podporze B, w tym celu skorzystamy z sumy momentów sił względem punktu A.

$$\sum M_A = 0 \quad (3.7)$$

Podstawiając wszelkie momenty względem punktu A otrzymujemy:

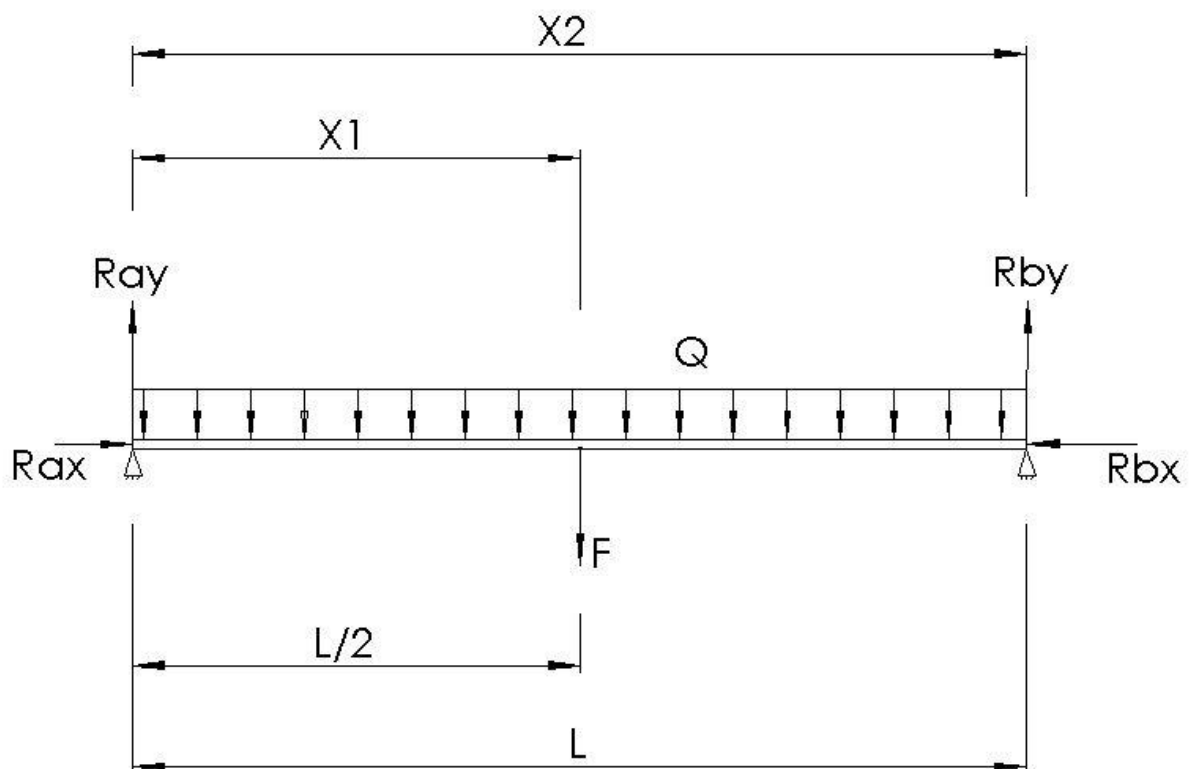
$$-R_{by} \cdot l + F \cdot \frac{l}{2} + Q \cdot \frac{l^2}{2} = 0 \quad (3.8)$$

Przekształcając równanie (3.7) w celu wyznaczenia reakcji w podporze B w osi Y otrzymujemy:

$$R_{by} = \frac{F \cdot \frac{l}{2} + Q \cdot \frac{l^2}{2}}{l} = \frac{F + Q \cdot l}{2} \quad (3.9)$$

3.3. Obliczenie sił tnących w belce.

Następnie, zostały wyznaczone siły tnące i momenty gnące działające na powyższą belkę.



Rysunek 3.2. Schemat belki z podziałem na dwa przedziały.

Równanie sił tnących działających wyznaczane od lewej strony belki w pierwszym przedziale:

$$T = R_{ay} - Q \cdot x_1 \quad (3.10)$$

Gdzie:

- T – siły tnące.
- x_1 – odległość od punktu A z pierwszego przedziału.

Równanie sił tnących w drugim przedziale belki:

$$T = R_{ay} - Q \cdot x_2 - F \quad (3.11)$$

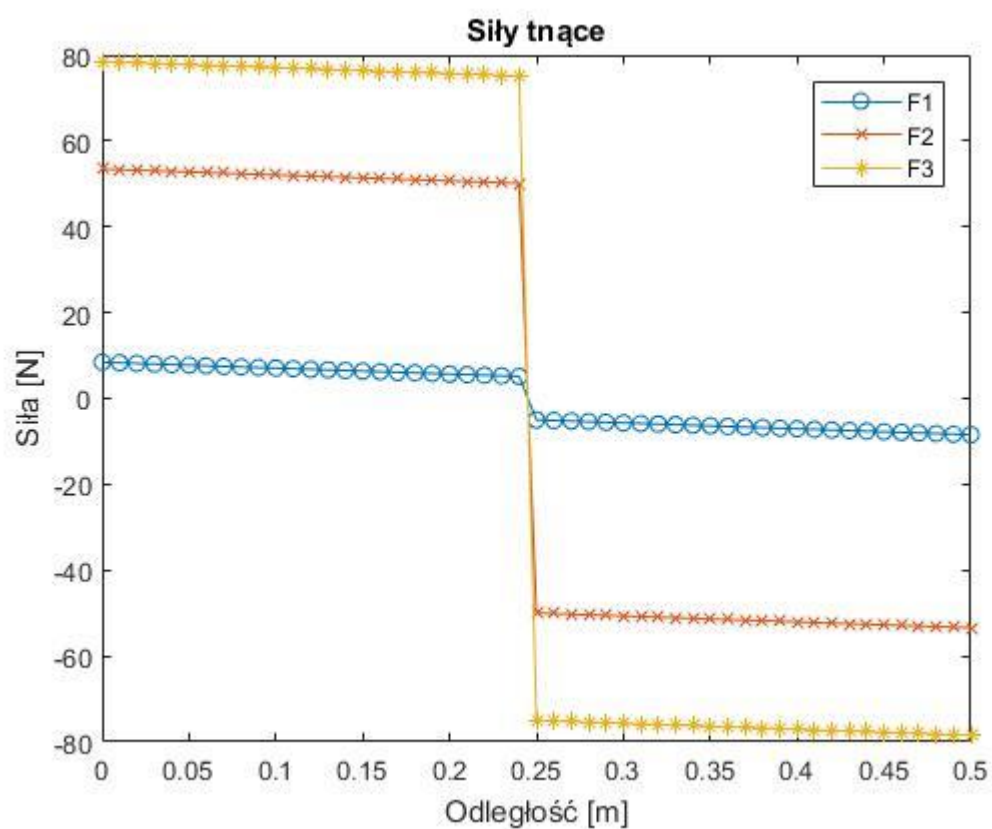
Gdzie:

- x_2 – odległość od punktu A z drugiego przedziału.

3.4. Otrzymanie wykresów sił tnących.

Wyznaczając wykresy sił tnących dla profilu konstrukcyjnego użyto następujących wartości:

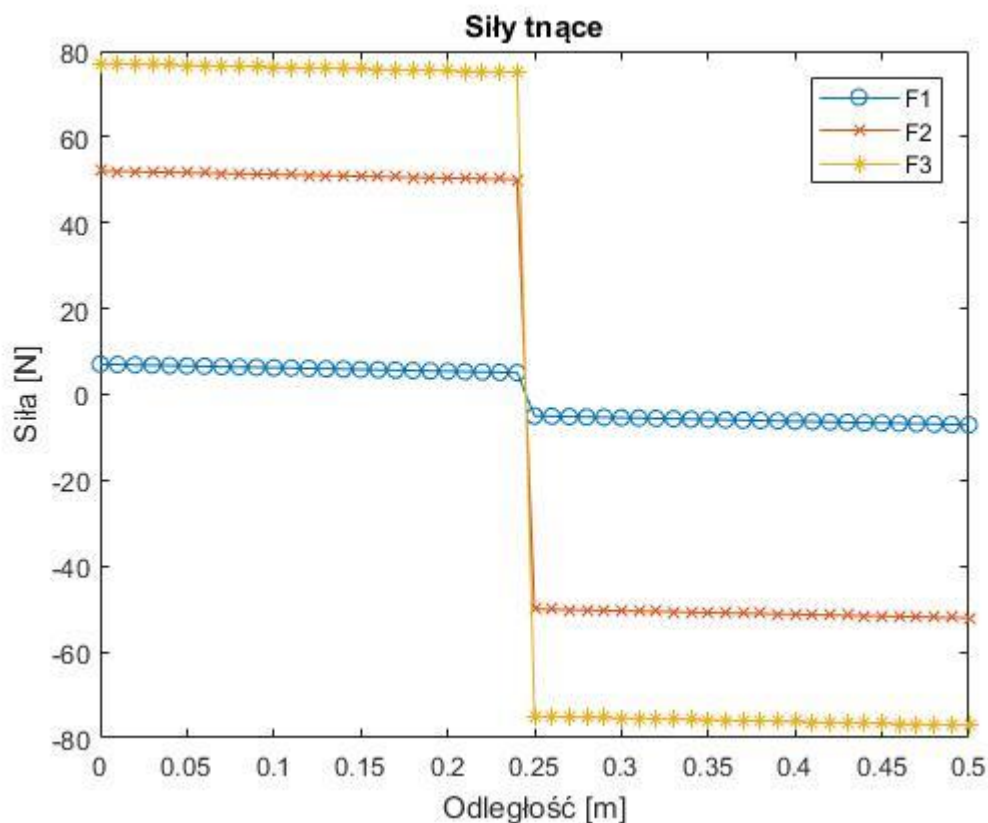
- $Q = 13,52 \frac{N}{m}$
- $F_1 = 1 \text{ kg}$
- $F_2 = 10 \text{ kg}$
- $F_3 = 15 \text{ kg}$



Rysunek 3.3. Wykres sił tnących dla aluminiowego profilu konstrukcyjnego 40x40.

Do uzyskania wykresu sił tnących dla rury kwadratowej 40x40x2 użyto następujących wartości

- $Q = 8.1117 \frac{N}{m}$
- $F_1 = 1 \text{ kg}$
- $F_2 = 10 \text{ kg}$
- $F_3 = 15 \text{ kg}$



Rysunek 3.4. Wykres przedstawiający siły tnące dla aluminiowej rury kwadratowej 40x40x2.

3.5. Obliczenie momentów gnących w belce.

Aby wyznaczyć ugięcie belki należy wyznaczyć momenty gnące występujące w belce. Równanie momentów gnących z pierwszego przedziału belki jest następujące:

$$M_g = R_{ay} \cdot x_1 - \frac{Q \cdot x_1^2}{2} \quad (3.12)$$

Gdzie:

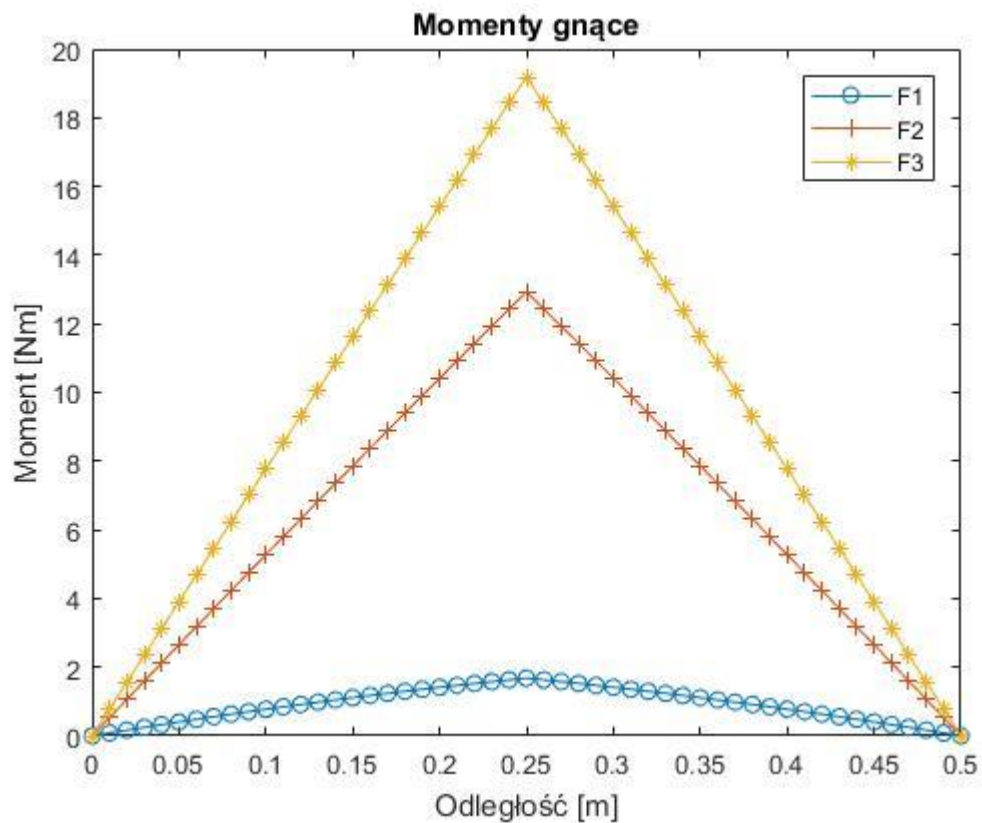
- M_g – momenty gnące w belce.

Równanie momentów gnących w drugim przedziale belki jest następujące:

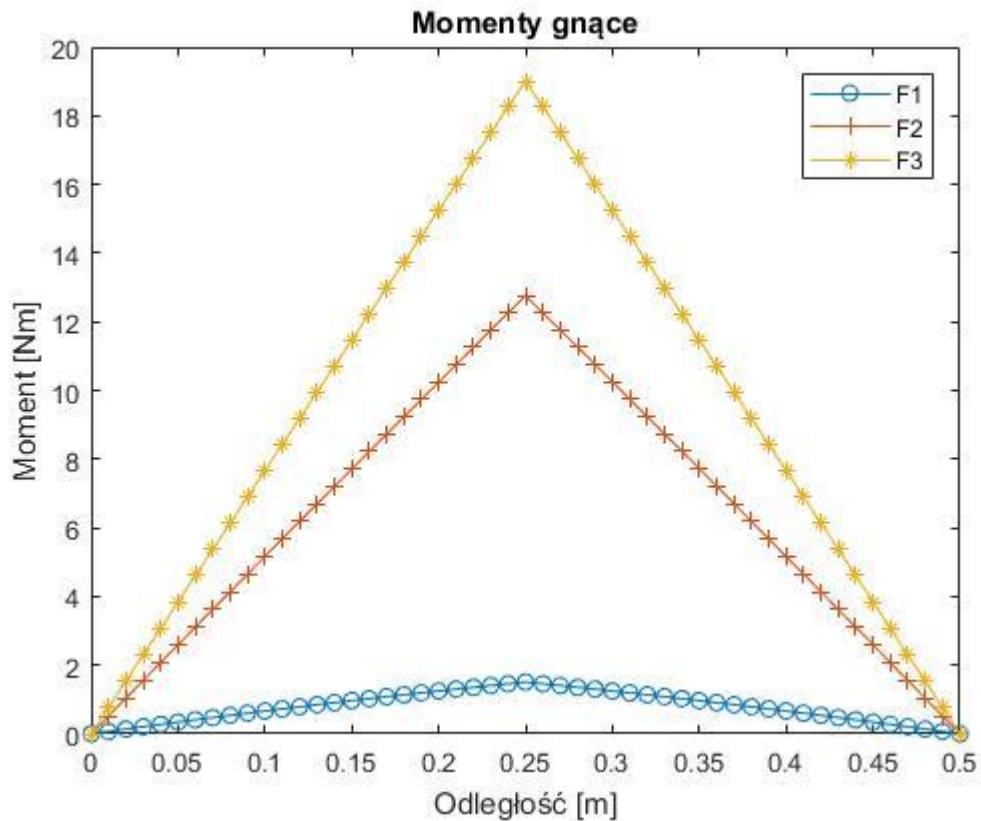
$$M_g = R_{ay} \cdot x_2 - \frac{Q \cdot x_2^2}{2} - F \cdot \left(x_2 - \frac{l}{2}\right) \quad (3.13)$$

3.6. Otrzymanie wykresów momentów gnących.

Do wyznaczenia wykresów momentów gnących dla profilu konstrukcyjnego i rury kwadratowej, użyte zostały te same wartości zmiennych użytych w wyznaczaniu wykresów sił tnących.



Rysunek 3.5. Wykres momentów gnących dla profilu konstrukcyjnego.



Rysunek 3.6. Wykres momentów gnących dla aluminiowej rury kwadratowej 40x40x2.

3.7. Obliczenie ugięcia belki.

W celu wyznaczenia ugięcia belki stosujemy zależność:

$$\frac{d^2 y}{dx^2} = \frac{M_g}{E \cdot J} \quad (3.14)$$

Gdzie:

- E – moduł Younga dla aluminium.
- J – moment bezwładności belki w osi zginania.

Podwójnie całkując powyższe równanie zgodnie z założeniami metody Clebscha otrzymujemy:

$$y(x) = \frac{1}{E \cdot J} \left(\int \int M_g(x) dx + Cx + D \right) \quad (3.15)$$

Gdzie:

- C – stała pierwszego całkowania.
- D – stała drugiego całkowania.

Podstawiając do równania (3.15) momenty gnące w obu przedziałach otrzymujemy następujące równania:

$$y(x_1) = -\frac{R_{ay} \cdot x_1^3}{6} + \frac{Q \cdot x_1^4}{24} + C \cdot x_1 + D \quad (3.16)$$

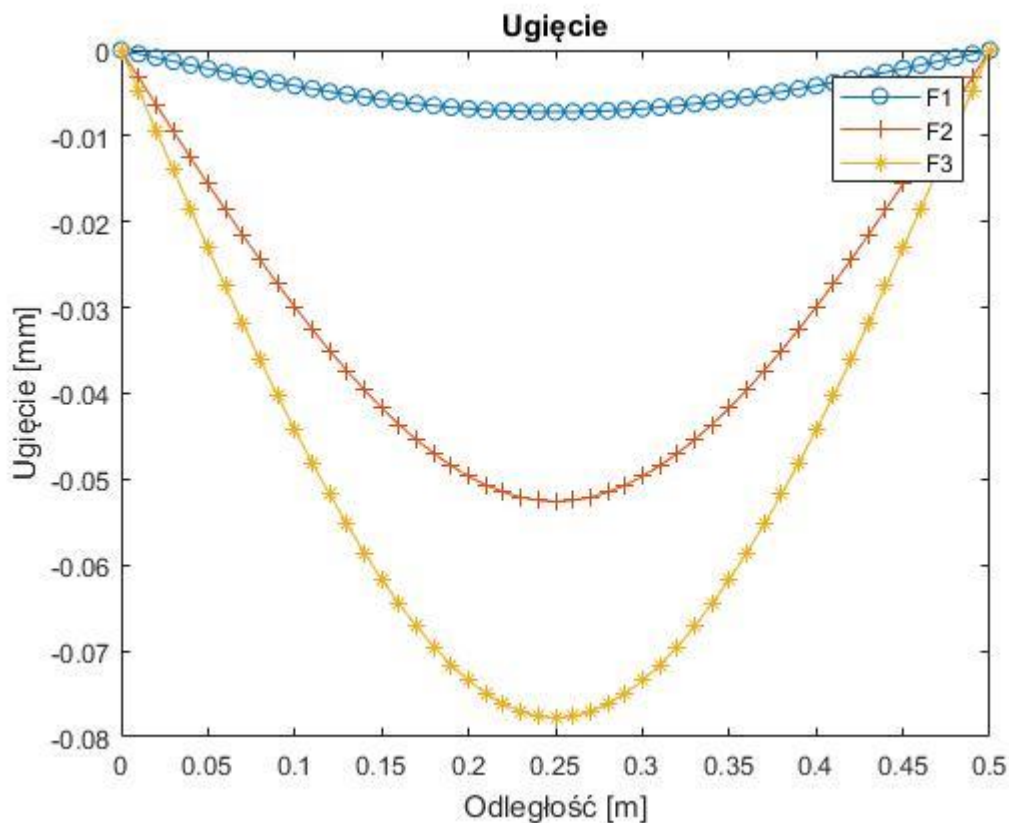
$$y(x_2) = -\frac{R_{ay} \cdot x_2^3}{6} + \frac{Q \cdot x_2^4}{24} + \frac{F \cdot (x_2 - \frac{l}{2})^3}{6} + C \cdot x_2 + D \quad (3.17)$$

Wzór (1.15) opisuje ugięcie belki w pierwszym przedziale, a wzór (3.16) w drugim przedziale belki.

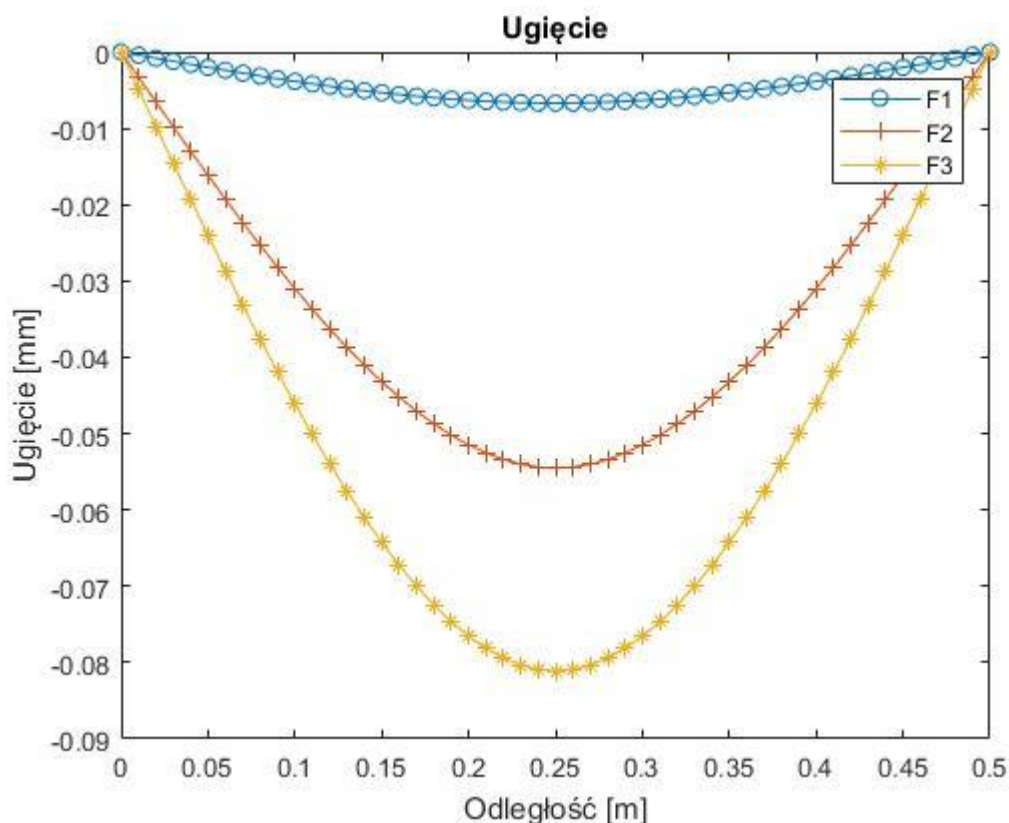
3.8. Otrzymanie wykresów ugięcia belki.

Przy obliczaniu wartości użytych do stworzenia wykresu obu profili użyte zostały następujące wartości:

- $E = 70 \cdot 10^9 [Pa]$
- $I_{konstrukcyjny} = 7,5783 \cdot 10^{-8} [m^4]$
- $I_{kwadrat} = 7 \cdot 10^{-8} [m^4]$



Rysunek 3.7. Wykres ugięcia belki dla aluminiowego profilu konstrukcyjnego 40x40.



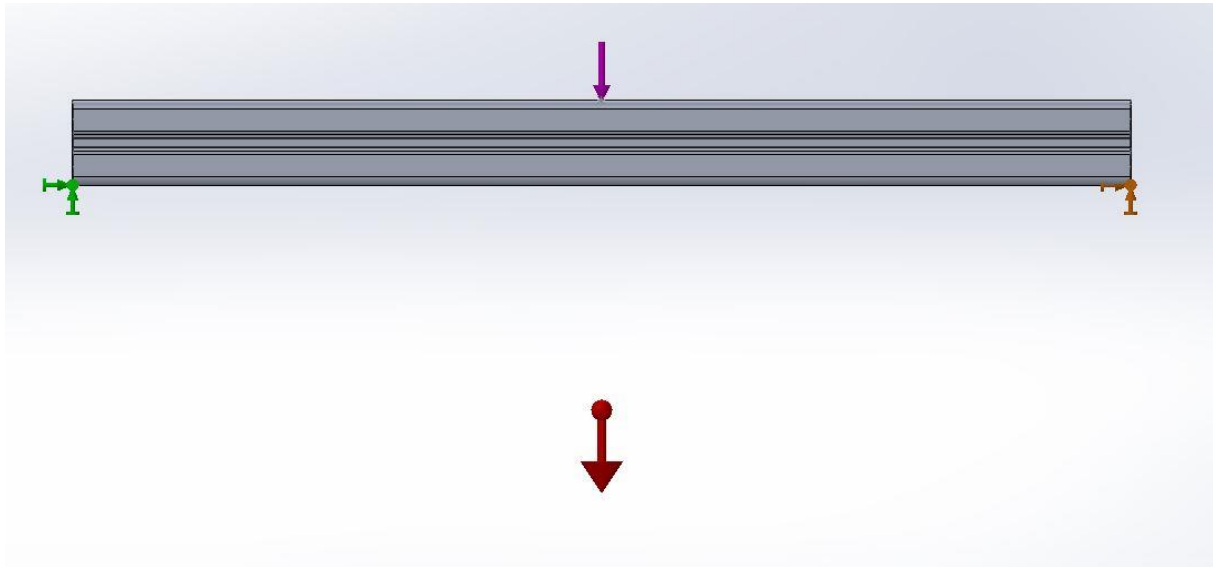
Rysunek 3.8. Wykres ugięcia belki dla aluminiowej rury kwadratowej 40x40x2.

3.9. Wybór aluminiowego profilu konstrukcyjnego.

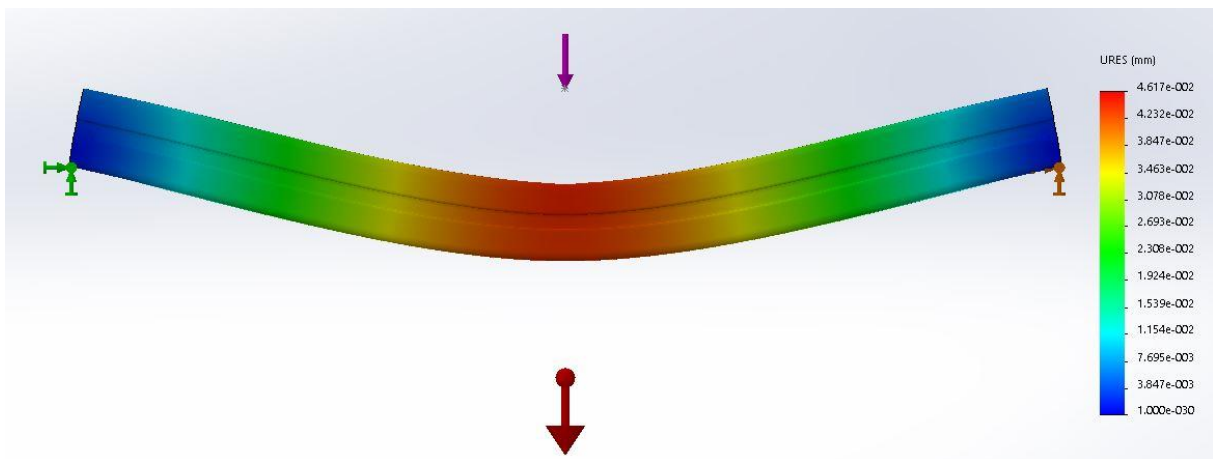
Przy wyborze profilu, z którego powinien być wykonany model suwnicy bramowej jednym czynnikiem było ugięcie belki, które powinno być mniejsze dla profilu konstrukcyjnego. Dodatkowym czynnikiem jest łatwość pracy z profilem konstrukcyjnym, duża ilość producentów profili i akcesoriów do profili pozwala na szybsze i łatwiejsze dobranie niezbędnych elementów oraz na zmniejszenie ogólnych kosztów związanych z budową modelu suwnicy bramowej.

3.10. Sprawdzenie poprawności wyników.

Aby upewnić się, że wyniki uzyskane analitycznie mogą odpowiadać rzeczywistości wykonana została analiza statyczna belki w programie SOLIDWORKS, wraz z jej przemieszczeniem. Obciążona została ciężarem 15 kg.



Rysunek 3.9. Belka użyta do analizy statycznej w programie SOLIDWORKS.

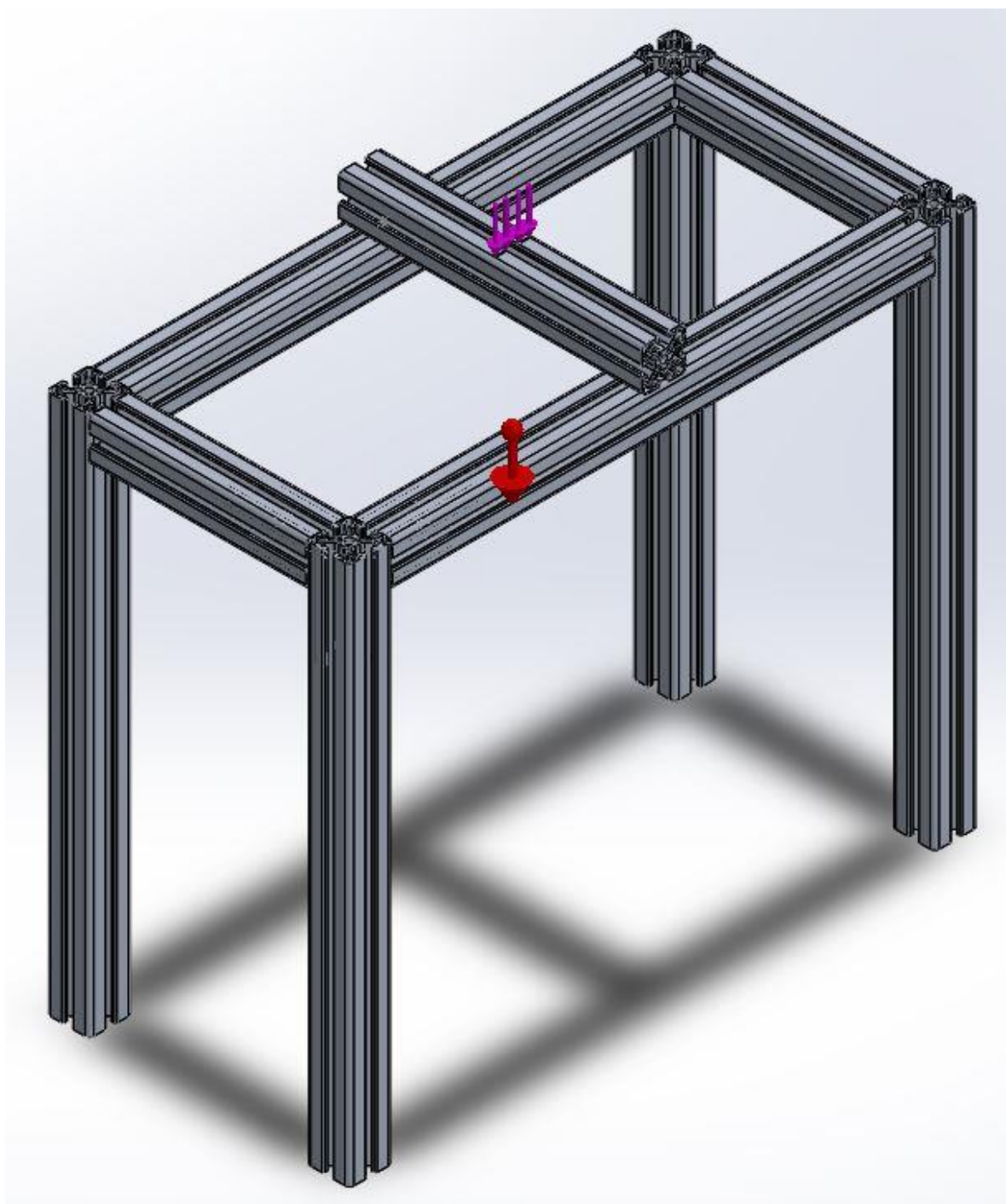


Rysunek 3.10. Ugięcie belki z programu SOLIDWORKS

Porównując maksymalną wartość przemieszczenia uzyskaną metodą elementów skończonych: $4,617 \cdot 10^{-2}$ a maksymalną wartością ugięcia dla profilu konstrukcyjnego $7,779 \cdot 10^{-2}$, można założyć, że błąd między obliczeniami analitycznymi, a metodą elementów skończonych jest pomijalnie mały.

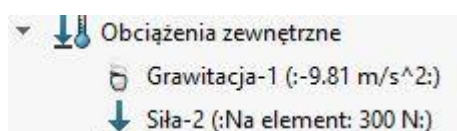
3.11. Analiza statyczna konstrukcji suwnicy bramowej

Ugięcie całej suwnicy zostało policzone metodą elementów skończonych w programie SOLIDWORKS. Model użyty do obliczeń wygląda następująco:



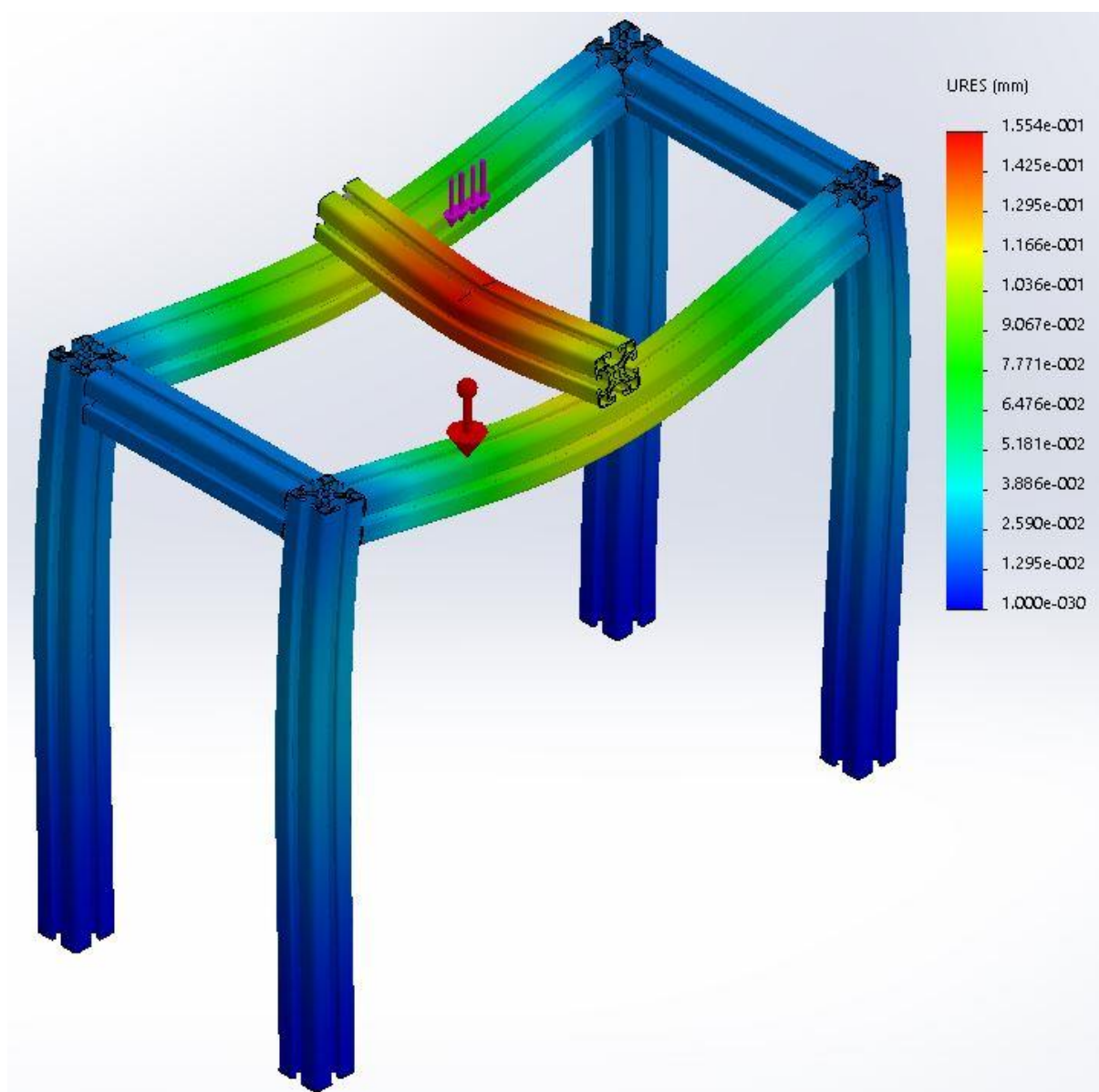
Rysunek 3.11. Model użyty do obliczeń metodą elementów skończonych.

Obciążenia zewnętrzne użyte do obliczeń:



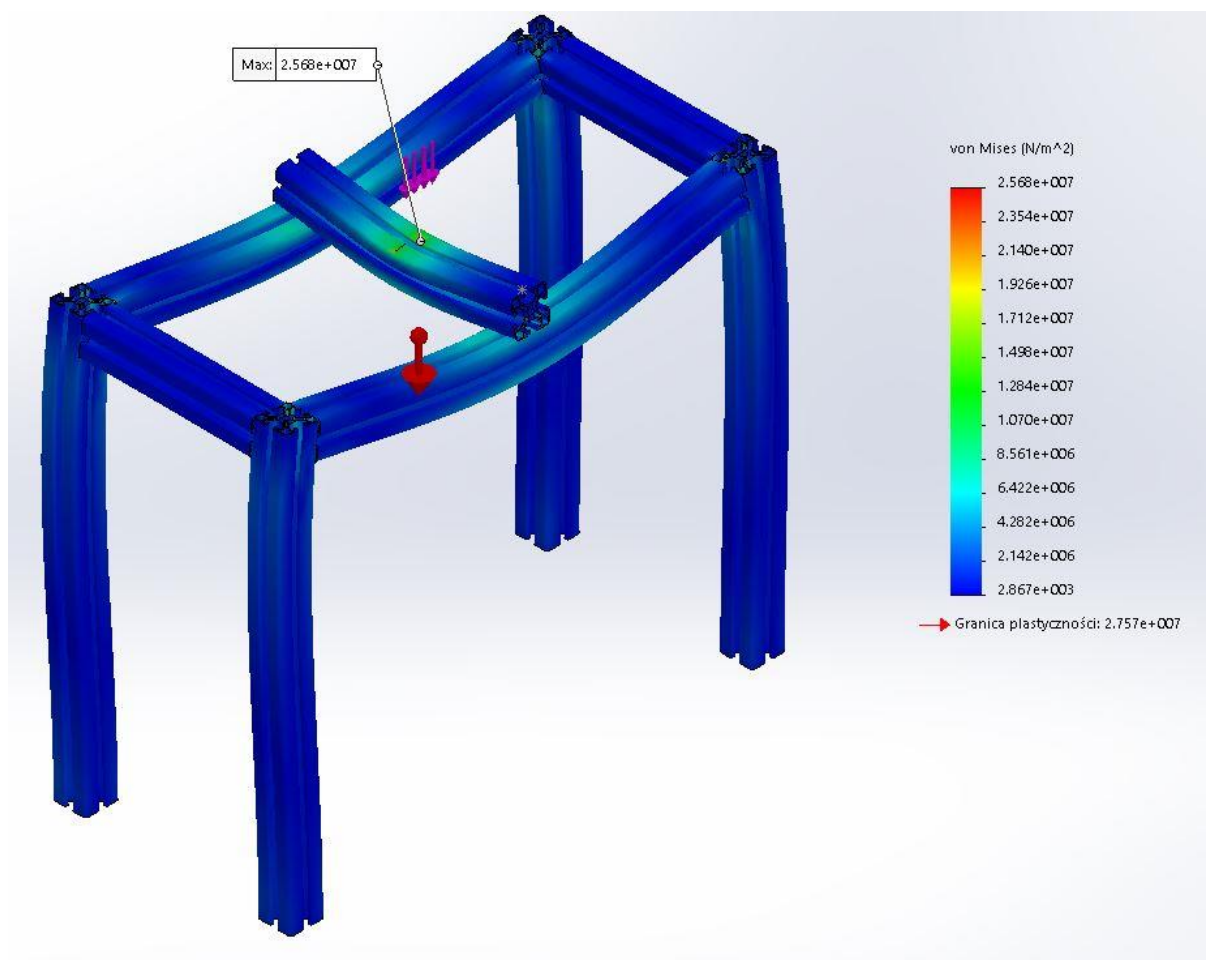
Rysunek 3.12. Obciążenia zewnętrzne użyte do obliczeń w programie SOLIDWORKS.

Podczas użytkowania suwnicy założone zostało że maksymalny ciężar, który powinien zostać zawieszony na suwnicy powinien wynosić 10 kg, ale do obliczeń wykorzystujemy ciężar 30 kg. Ze względu na to, że w przyszłości na suwnicy powinien zostać zamontowany system anulowania drgań, którego masy nie jesteśmy w stanie przewidzieć.



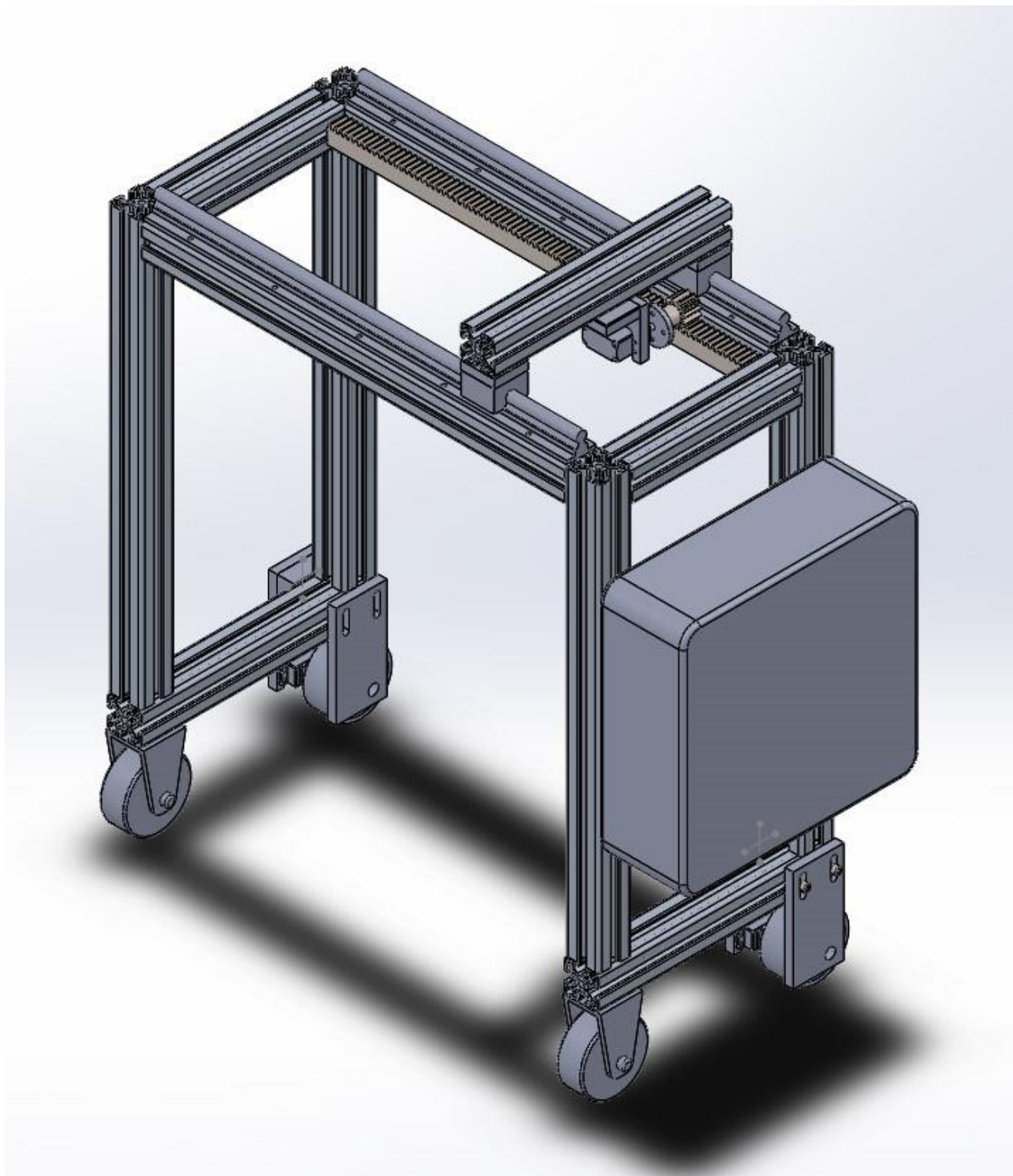
Rysunek 3.13. Ugięcie ramy suwnicy.

Naprężenia w całej suwnicy są następujące:



Rysunek 3.14. Naprężenia w ramie modelu.

Po wykonaniu obliczeń wytrzymałościowych zaprojektowany został model CAD suwnicy bramowej wraz z elementami wykonawczymi oraz uchwytami niezbędnymi do zamocowania ich na ramie modelu.



Rysunek 3.15. Model CAD suwnicy bramowej.

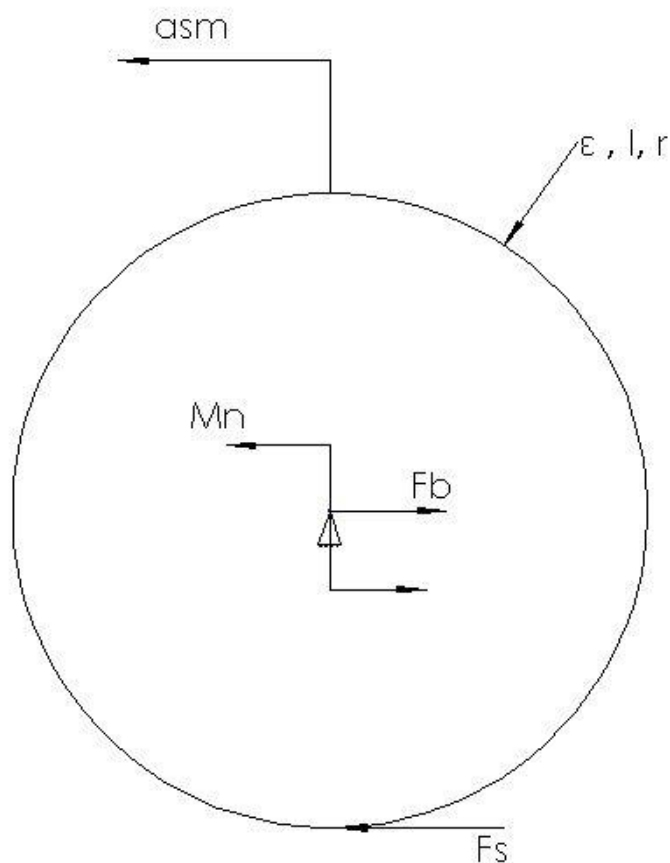
Rozdział 4

Dobór elementów wykonawczych.

Kolejnym krokiem przy projektowaniu suwnicy był dobór silników napędowych dla dwóch osi jezdnych. Oś Y będzie poruszać się za pomocą dwóch serwomechanizmów zamocowanych do kół jezdnych, zaś oś X będzie zrealizowana za pomocą listwy zębatej i koła zębatego.

4.1. Dobór silników napędowych.

W celu dobrania silników dysponujących odpowiednim momentem napędowym, obliczamy moment potrzebny do napędzenia koła napędowego, które zostało przedstawione na poniższym rysunku.



Rysunek 4.1. Schemat koła napędowego użytego do obliczenia momentu napędowego.

Gdzie:

- I – moment bezwładności koła napędowego,
- r – promień koła napędowego.
- M_n – moment napędowy silnika,
- F_b – siła bezwładności całej suwnicy bramowej,
- F_s – siła Styczna,
- a_{sm} – przyspieszenie liniowe.
- ε – przyspieszenie kątowe.

Wyznaczając M_n należało wyprowadzić dwa równania – sumę sił na osi X oraz sumę momentów względem środka koła napędowego. Suma sił na osi X:

$$F_s = F_b \quad (4.1)$$

$$F_s = (m_1 + m_2) \cdot a_{sm} \quad (4.2)$$

Gdzie:

- m_1 – masa suwnicy bramowej,
- m_2 – masa koła napędowego.

Suma momentów względem środka koła napędowego:

$$M_n - F_s \cdot r = I \cdot \varepsilon \quad (4.3)$$

$$\varepsilon = \frac{a_{sm}}{r} \quad (4.4)$$

$$I = \frac{m_2 \cdot r^2}{2} \quad (4.5)$$

$$M_n = I \cdot \varepsilon - F_s \cdot r \quad (4.6)$$

4.2. Obliczenie wartości momentów napędowych.

Moment napędowy dla osi napędowej został policzony przy przyjęciu następujących wartości:

- $r = 80 \text{ [mm]}$
- $a_{sm} = 1 \text{ [\frac{m}{s^2}]}$
- $m_1 = 30 \text{ [kg]}$
- $m_2 = 0,01 \text{ [kg]}$

Do obliczenia momentu potrzebnego do napędzenia suwnicy bramowej, potrzebne jest przyspieszenie - a_{sm} . Przy ustalaniu założeń konstrukcyjnych ważniejsze było ustalenie i osiągnięcie zadanej prędkości, niż otrzymanie zadanego przyspieszenia, dlatego zostało ono przyjęte w taki sposób aby czas, w którym suwnica rozpędzi się do maksymalnej prędkości był stosunkowo mały – przy przyspieszeniu $1 \text{ [\frac{m}{s^2}]}$ ten czas wynosi $0,25 \text{ [s]}$.

Po wykonaniu obliczeń z powyższymi wartościami wypadkowy moment napędowy serwomechanizmu powinien wynosić $M_n = 2.4 \text{ [Nm]}$. Znając potrzebny moment napędowy do uzyskania zakładanego przyspieszenia zostały dobrane dwa takie same serwomechanizmy PowerHD LF-20MG-360 Standard, których zadaniem będzie napędzenie suwnicy bramowej.



Rysunek 4.2. Serwomechanizm użyty w suwnicy bramowej.

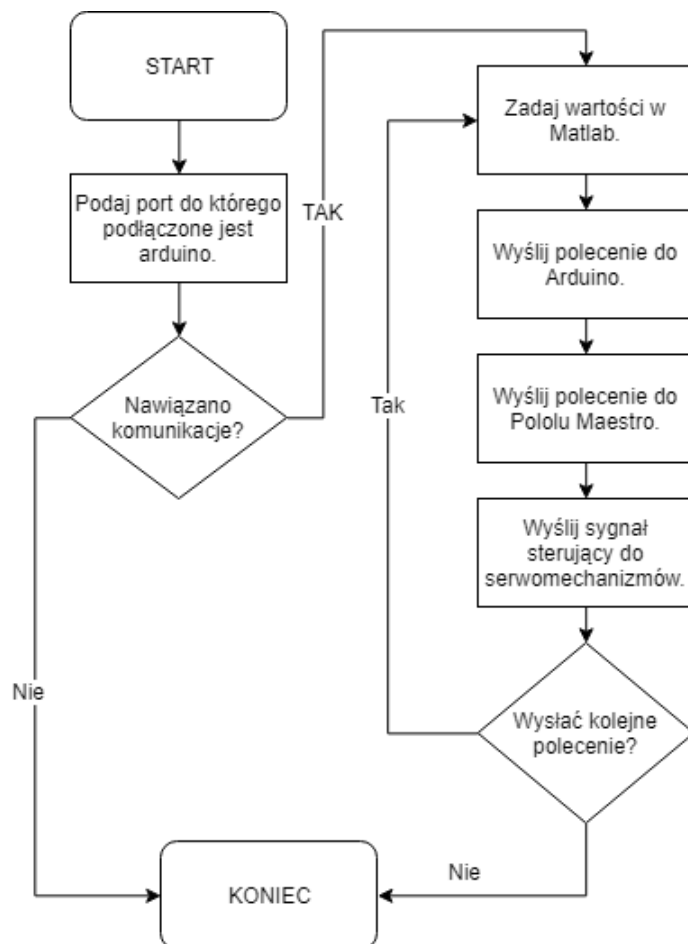
Źródło: <https://alexnlld.com/wp-content/uploads/2016/03/f29731a5-87a5-4455-b537-0da53012a029.jpg>
07.04.2018

Ze względu na wykonywane próby działania serwomechanizmu, i poprawnego działania przy sterowaniu nim, ten sam serwomechanizm został użyty do realizacji ruchu liniowego dla osi Y.

Rozdział 5

Algorytm sterowania.

Do sterowania modelem wykorzystywany jest program napisany w Matlabie, który po porcie szeregowym COM będzie komunikował się z Arduino. Arduino interpretuje polecenia wysyłane portem szeregowym i na ich podstawie może wykonywać szereg czynności. Pozwala to w przyszłości na rozbudowanie systemu o dodatkowe komponenty. Głównym zadaniem Arduino jest wysyłanie informacji do sterownika PWM Pololu MicroMaestro poleceń. Arduino jest pośrednikiem w tej komunikacji z tego względu, że komunikacja między Matlabem a Pololu Micromaestro nie jest wspierana żadnymi gotowymi bibliotekami.



Rysunek 5.1. Główny algorytm sterowania.

Po uruchomieniu programu użytkownik zostaje proszony o podanie portu COM, do którego podłączone jest Arduino. Po poprawnym wpisaniu portu i nawiązaniu połączenia program przejdzie do okienka odpowiedzialnego za sterowanie modelem.

5.1. Aplikacja w Matlab.

Suwnica bramowa będzie sterowana za pomocą aplikacji napisanej w Matlabie, będzie ona łączyła się z arduino za pomocą interfejsu szeregowego. Interfejs aplikacji prezentuje się następująco:



Rysunek 5.2. Interfejs aplikacji sterującej.

Aplikacją można sterować suwnicą w dwóch osiach, zadając prędkość albo określając pewne przemieszczenie. Klikając przycisk „Steruj X” lub „Steruj Y” wysyłamy polecenie do Arduino, z zadanymi parametrami. Polecenie powinno zostać zinterpretowane przez mikrokontroler i powinien on wysterować Pololu MicroMaestro użyte do sterowania serwomechanizmów sygnałem PWM. Przyciski „Stop X” i „Stop Y” wysyłają polecenie zatrzymania serwomechanizmów.

5.2. Program Arduino.

Arduino w suwnicy ma za zadanie wysyłać polecenia do sterownika serw, oraz odbierać polecenia wysyłane przez port szeregowy z programu Matlab. Samo Arduino powinno nasłuchiwać komunikatów wysyłanych portem szeregowym, a następnie interpretować je. W tym podrozdziale przedstawiony będzie kod użyty w Arduino UNO wraz z opisem działania.

Biblioteki użyte w Arduino:

- PololuMaestro.h – biblioteka odpowiedzialna za polecenia użyte w komunikacji z Pololu Micromaestro

- `SoftwareSerial.h` – biblioteka, która umożliwia obsługę wirtualnego portu szeregowego którym będzie możliwa komunikacja ze sterownikiem.

Przy uruchomieniu Arduino uruchamiane zostają dwa porty szeregowe, jeden do komunikacji z Matlabem a drugi do nawiązania połączenia z Pololu MicroMaestro.

- `Serial.begin(9600);`
- `maestroSerial.begin(9600);`

W pętli na Arduino wykonywane są dwie funkcje:

`receiveEndMarker()` – funkcja ta zapisuje wartości odebrane z portu szeregowego,

`ExecuteCommand()` – ta funkcja ma za zadanie wykonać różne polecenia w zależności od tego jakie polecenie zostało zapisane przez funkcję `receiveEndMarker()`.

Funkcja `receiveEndMarker()` odczytuje nowe informacje pojawiające się na porcie szeregowym i zapisuje ciąg znaków w 4 różne zmienne. Pierwsza nazwana `Data1` odpowiada za rodzaj polecenia które ma zostać wykonane, np.: polecenie „servo” ma za zadanie wysterować serwomechanizm z parametrami zapisanymi w trzech następnym zmiennych. Każda kolejna zmienna oddzielona jest znakiem „:”, a koniec polecenia oznaczony jest znakiem „,”.

Funkcja `ExecuteCommand()` wykonuje polecenie z listy komend opisanych w funkcji `CommandList()`, a następnie czyści macierz, w której zapisane były zmienne użyte do wykonywania poleceń.

`CommandList()` zawiera listę czynności które powinny zostać wykonane w zależności od tego jaka wartość znajduje się w pierwszej zmiennej. Przy wartości „servo” steruje serwomechanizm, używając drugiej zmiennej jako port na sterowniku, trzeciej i czwartej zaś jako kolejno wartość prędkości i przyspieszenia. Wartość „a” jest używana do potwierdzenia nawiązania komunikacji między Matlabem a Arduino.

Dla bezpieczeństwa suwnica bramowa jest wyposażona w dwa czujniki krańcowe, które mają za zadanie ograniczyć obszar ruchu w osi Y. Ruch w osi X nie jest niczym ograniczony ze względu na brak konstrukcji po której suwnica powinna się poruszać.

5.3. Program w Pololu MicroMaestro.

Do sterowania serwomechanizmami używany jest sygnał PWM, który wysyłany jest dzięki Pololu MicroMaestro. Sterownik Pololu używany jest też do obsługi dwóch przełączników krańcowych, które mają za zadanie ograniczyć ruch w osi Y modelu. Po przełączeniu przełącznika krańcowego, serwomechanizm powinien przemieścić oś do miejsca, w którym przełącznik nie jest załączony. Następnie umożliwi użytkownikowi ponowne sterowanie modelem.

Rozdział 6

Testy.

Poniższy rozdział jest w całości poświęcony testom, które mają za zadanie zweryfikować czy suwnica po zmontowaniu spełnia założenia projektowe. Wykonane zostały dwa testy, za których pomocą ustalono czy suwnica spełnia dwa najważniejsze założenia projektowe: czy osiąga ona założoną prędkość oraz czy ugięcie suwnicy przy obciążeniu nie jest większe niż to, które zostało założone.



Rysunek 6.1. Złożony model suwnicy bramowej.

6.1. Pomiar prędkości.

Pierwszym testem jest wykonanie pomiaru czasu w jakim suwnica pokona odległość 2 [m] i na podstawie tego pomiaru wyliczona jest średnia prędkość suwnicy:

$$V = \frac{s}{t} \quad (6.1)$$

Gdzie:

- s – droga pokonana przez suwnicę,
- t – czas, w którym suwnica pokonała drogę,
- V – średnia prędkość suwnicy na danym odcinku drogi.

Do wykonania pomiaru użyty został stoper cyfrowy, oraz taśma miernicza w celu wyznaczenia długości drogi, na której przemieszczać będzie się model.

6.2. Uzyskanie wyniku prędkości wraz z jego niepewnością.

Odległość, którą suwnica pokonuje podczas testów jest równa 2 [m], zmierzony czas przejazdu jest podany w tabeli poniżej.

Tabela 6.1. Tabela wyników pomiarów prędkości.

Numer pomiaru	Czas przejazdu [s]	Numer pomiaru	Czas przejazdu [s]
1	18,03	6	17.58
2	17.93	7	18.16
3	18.22	8	17.96
4	17.51	9	18.21
5	18.32	10	18.02

Średnią uzyskiwaną prędkość suwnicy otrzymujemy ze wzoru:

$$\bar{t} = \frac{1}{n} \sum_{i=1}^n t_i \quad (6.2)$$

Gdzie:

- V – średnia prędkość suwnicy.
- n – liczba prób.

Niepewność tego pomiaru, ze względu na to że jest to pomiar pośredni, otrzymujemy ze wzorów:

$$u(V) = \sqrt{\left(\frac{\partial V}{\partial t}\right)^2 u^2(t)} \quad (6.3)$$

$$\frac{\partial V}{\partial t} = \frac{s}{t^2} \quad (6.4)$$

$$u(t) = \sqrt{\frac{1}{n(n-1)} \sum_{i=1}^n (t_i - \bar{t})^2} \quad (6.5)$$

Gdzie:

- $u(V)$ – niepewność pomiarowa przy pomiarze prędkości,
- $u(t)$ – niepewność pomiarowa przy pomiarze czasu.

Podstawiając dane z tabeli 6.1 do wzorów (6.2) i (6.5) prędkość średnia i jej odchylenie wynosi:

$$V = 6,67 \pm 0,0371 \left[\frac{m}{min} \right]$$

6.3. Pomiar ugięcia.

Drugim testem było sprawdzenie ugięcia dla suwnicy, w tym celu użyto wskaźnika zegarowego do zarejestrowania przemieszczenia się belki po umieszczeniu na niej ciężaru $10[kg]$. Wynik ten nie powinien przekraczać $0.1[mm]$. W celu zmniejszenia niepewności pomiarowej zostało wykonane 10 pomiarów ugięcia belki. Wyniki z pomiarów zapisane są w tabeli poniżej:

Tabela 6.2. Tabela wyników uzyskanych podczas pomiarów ugięcia belki

Numer pomiaru	Przemieszczenie [mm]	Numer pomiaru	Przemieszczenie [mm]
1	0.05	6	0.05
2	0.06	7	0.06
3	0.05	8	0.08
4	0.08	9	0.06
5	0.07	10	0.06

Do wykonania pomiaru ugięcia został użyty czujnik zegarowy, który mierzy ugięcie belki nośnej suwnicy.

6.4. Uzyskanie wyniku ugięcia wraz z jego niepewnością.

W celu otrzymania wyniku ugięcia, które występuje przy obciążeniu belki masą $10[kg]$ obliczono średnią uzyskaną z 10 prób, następnie policzono niepewność standardową typu A tego wyniku.

$$\bar{Y} = \frac{1}{n} \sum_{i=1}^n Y_i \quad (6.6)$$

Gdzie:

- Y – ugięcie $[mm]$

Niepewność pomiarowa typu A została policzona według wzoru:

$$u(Y) = \sqrt{\frac{1}{n(n-1)} \sum_{i=1}^n (Y_i - \bar{Y})^2} \quad (6.7)$$

Gdzie:

- $u(Y)$ – niepewność pomiarowa przy pomiarze ugięcia.

Podstawiając wartości z tabeli 6.2 do wzorów (6.6) i (6.7) ugięcie wraz z jego odchyleniem wynosi:

$$Y = 0,062 \pm 0,011 [mm]$$

Zakończenie.

Celem pracy inżynierskiej było zaprojektowanie i skonstruowanie modelu suwnicy bramowej, która w przyszłości ma być rozszerzona o układ anulujący drgania podczas pracy suwnicy. Po wykonaniu testów model suwnicy spełnia najważniejsze założenia projektowe, czyli osiąga prędkość średnią równą 6,67, i ugięcie nie jest większe niż 0,1 [mm]. Model został zaprojektowany w taki sposób aby w przyszłości była możliwość rozszerzenia go. Podstawowym rozszerzeniem będzie układ anulujący drgania, kolejnym ulepszeniem modelu jest komunikacja bezprzewodowa z Arduino UNO, oraz wyposażenie modelu w Akumulator, co pozwoliłoby na całkowicie bezprzewodową pracę suwnicy – coś co teraz było nie możliwe ze względu na ograniczone koszty przy projektowaniu. Wraz z rozwojem modelu możliwe jest wymienienie serwomechanizmów, na taki rodzaj napędu który będzie w stanie spełnić inne założenia projektowe.

Bibliografia.

- [1] Abus Crane Systems Polska, <https://www.abuscranes.pl/>, w dn. 10.09.2017.
- [2] Abus suwnica półbramowa,
http://www.abuscranes.pl/var/storage/images/krane/laufkrane/halbportalkrane/33751-114-pol-PL/suwnice-polbramowe_productImage.png 20.03.2018
- [3] BMH Semi-gantry Crane <http://www.ytcrane.com/semi-gantry-crane.html#contact-page> w dn. 05.09.2017.
- [4] Career Trend,,: *Types of gantry cranes* <https://careertrend.com/info-8495093-types-gantry-cranes.html> , w dn. 10.09.2017.
- [5] EMH <https://emhcranes.com/free-standing-bridge-crane-system/>, w dn. 12.09.2017.
- [6] Halliday D., Resnick R., Walker J.: *Podstawy fizyki Tom I*. Wydawnictwo Naukowe PWN, str. 375-396 Warszawa 2015.
- [7] Rialex Crane Systems,,: *Suwnice* <https://www.rialex.com.pl/pl/produkty/urzadzenia-transportu-bliskiego/suwnice>, w dn. 10.09.2017.
- [8] Rialex suwnica bramowa, [https://www.rialex.com.pl/media/file/SuwnicaBramowa\(1\).png](https://www.rialex.com.pl/media/file/SuwnicaBramowa(1).png) 20.03.2018
- [9] Rialex suwnica pomostowa,
https://www.rialex.com.pl/media/Images/Animacje/Suwnice_new/L_Suwnica_dwubelkowa.png 20.03.2018
- [10] Rialex suwnica wysięgnikowa,
https://www.rialex.com.pl/media/Images/Animacje/Suwnice_new/L_Suwnica_konsolowa.png 20.03.2018
- [11] The American Society of Mechanical Engineers,,: *The PAECO Container Crane*,
<https://www.asme.org/wwwasmeorg/media/ResourceFiles/AboutASME/Who%20We%20Are/Engineering%20History/Landmarks/85-paceco-container-crane.pdf>, w dn. 11.09.2017.
- [12] Total Crane <http://www.totalcrane.com/types-of-overhead-cranes.html>, w dn. 05.09.2017.
- [13] Urząd Dozoru Technicznego,
https://www.udt.gov.pl/index.php?option=com_content&view=article&id=493:suwnice-wciagniki-wciagarki-zuraw-stacjonarny-nazwa-definicja-i-podleganie&catid=160&Itemid=555, w dn. 05.09.2017.
- [14] Wolny S., Siemienieć A.: *Wytrzymałość materiałów część I*, Uczelniane wydawnictwo Naukowo-Dydaktyczne AGH, str. 165-312., Kraków 2002.

Dodatek A

Rysunki techniczne

W tym załączniku znajdują się rysunki techniczne części użytych do konstrukcji suwnicy bramowej, które zostały wykonane na hali technologicznej.

Dodatek B

Programy użyte w pracy inżynierskiej.

W tym dodatku znajdują się wszystkie programy napisane i użyte w modelu suwnicy bramowej.

Program 1 Program użyty w Arduino UNO

```
#include <PololuMaestro.h> //Załadowanie biblioteki Pololu

#include <SoftwareSerial.h> //Załadowanie biblioteki odpowiedzialnej za
obsługę wirtualnego portu szeregowego

SoftwareSerial maestroSerial(10, 11); //Definicja wirtualnego portu
szeregowego 10 RX 11 TX

MicroMaestro maestro(maestroSerial); //Definicja sterownika - Pololu micro
maestro

boolean incData = false; //Zmienna symbolizująca nowe nadchodzące
informacje

String Data[4]; //Zmienna przechowująca części przychodzącego strumienia
danych

String Data1; //Pierwsza część macierzy Data definiuje co ma zostać
wykonane

float Data2,Data3,Data4; //Parametry używane do obsługi poleceń

int i = 0;

char a = 'b' ;

void setup() {

    //Uruchomienie obu portów szeregowych - głównego portu Arduino oraz
wirtualnego portu Pololu

    Serial.begin(9600);

    maestroSerial.begin(9600);

}
```

```

void loop() {
    //Wywołanie dwóch głównych funkcji w pętli
    receiveEndMarker(); //Odbiór danych i podział ich na części składowe
    ExecuteCommand(); //Wykonywanie polecenia w zależności od odebranych
    danych.
}

void receiveEndMarker(){
    static char EndMarker = ','; //Znacznik końca polecenia
    static char Divider = ':'; //Znacznik dzielący polecenie
    char receive; //Zmienna w którą zapisywany jest bajt odebrany
    String Databuf1,Databuf2,Databuf3,Databuf4; //Bufer zmiennych - tylko do
    chwilowego przechowania danych służący do zmiany ich typu

    while (Serial.available() >0 && incData == false){
        receive = Serial.read();
        if(receive == Divider){
            i++;
        }
        if(receive != EndMarker && receive != Divider){
            Data[i] += receive;
        }
        if(receive == EndMarker){
            incData = true;
            i=0;
        }
        //Zmiana typu danych na taki, który może zostać użyty w przyszłości
        Databuf1 = Data[0];
        Databuf2 = Data[1];
        Databuf3 = Data[2];
        Databuf4 = Data[3];
        Data1 = Databuf1;
        Data2 = Databuf2.toFloat();
    }
}

```



```

    Data3 = Databuf3.toFloat();

    Data4 = Databuf4.toFloat();

}

}

void ExecuteCommand() {

    if(incData == true){

        CommandList(); //Funkcja ,w której są zapisane wszelkie polecenia do
        wykonania przez arduino, które są wywoływane z Matlab.

        for(int k =0;k<=3;k++){

            //Wyczyszczenie macierzy Data

            Data[k]="\0";

        }

    }

    incData = false;

}

void CommandList(){

    if(Data1 == "servo"){

        //Po wpisaniu do konsoli: 'servo:Data2:Data3:Data4,' sterownik
        wysterowuje serwomechanizm,

        //Data2 jest tutaj kanałem sterownika, Data3 prędkością obrotową, Data4
        przyspieszeniem.

        maestro.setAcceleration((int) Data2, (int)Data4);

        maestro.setTarget((int) Data2,(int) Data3);

    }

    if(Data1 == "test"){

        //polecenie testowe, ma za zadanie wyświetlić wartości użyte w
        poleceniu.

        Serial.println(Data1);

        Serial.println((int)Data2);

        Serial.println((int)Data3);

```

```

        Serial.println((int)Data4);
    }
    if(Data1 == "a"){
        Serial.println('a');
    }
}

```

Program 2 Program aplikacji do sterowania modelem w Matlab

```

classdef Sterowanie < matlab.apps.AppBase
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                matlab.ui.Figure
        KanalPanel               matlab.ui.container.Panel
        ButtonGroup              matlab.ui.container.ButtonGroup
        PredkoscButton           matlab.ui.control.RadioButton
        PrzemieszczenieButton    matlab.ui.control.RadioButton
        PrzyspieszenieLabel      matlab.ui.control.Label
        Acc                      matlab.ui.control.NumericEditField
        Parameter1               matlab.ui.control.NumericEditField
        SterujButton             matlab.ui.control.Button
        Parameter1Label          matlab.ui.control.Label
        StopButton               matlab.ui.control.Button
        Kana2Panel               matlab.ui.container.Panel
        ButtonGroup2             matlab.ui.container.ButtonGroup
        PredkoscButton2          matlab.ui.control.RadioButton
        PozycjaButton2           matlab.ui.control.RadioButton
        Przyspieszenie2Label     matlab.ui.control.Label
        Acc2                     matlab.ui.control.NumericEditField
        Parameter2               matlab.ui.control.NumericEditField
        SterujButton2            matlab.ui.control.Button
        Parameter2Label          matlab.ui.control.Label
    end
end

```

```

        Stop2Button          matlab.ui.control.Button
    end

    properties (Access = private)

        s;
    end

    methods (Access = private)

        function [Port] = PortOnBoot(app)

            Tytul = 'Port';
            Wiadomosc = 'Podaj port COM arduino, np.: COM3';
            Port = inputdlg(Wiadomosc,Tytul);
            Port = char(Port);
        end

        function [] = SetupSerial(app,Port)

            delete(instrfind)
            app.s = serial(Port);
            fopen(app.s);
            pause(2)
            a = 'b';
            fwrite(app.s,'a,');
            while(a ~= 'a')
                a = fread(app.s,1,'uchar');
            end
            if a == 'a'
                mbox= helpdlg('Komunikacja nawiązana');
                uiwait(mbox);
            else
                mbox = errordlg('BRAK KOMUNIKACJI','Błąd komunikacji');
            end
        end
    end
end

```

```

        uiwait(mbox)

    end

end

function [] = SendCmd(app,Command)

    %Funkcja ma wysłać polecenie na port szeregowy.

    disp(Command);

    fwrite(app.s,char(Command));

end

function [Send] = BuildCmd(app,Input)

    %Funkcja budująca polecenie, które zostanie wysłane do Arduino,
    Input jest wektorem o 3 argumentach.

    Msg = 'servo:%d:%.2f:%.2f,';

    Send = compose(Msg,Input);

end

function [Data] = GatherData(app,Channel)

    %Funkcja zbiera dane z pól numerycznych, jako wejście przyjmuje
    kanał na który ma zostać wysłane późniejsze polecenie.

    GatherDataMatrix = [app.Parameter1.Value, app.Acc.Value;

        app.Parameter2.Value, app.Acc2.Value];

    Data = [Channel, GatherDataMatrix(Channel+1,1),
        GatherDataMatrix(Channel+1,2)];

end

function [Data] = GetDir(app,Channel,Pos)

    if Pos == 0

        Data = [Channel 5972 0];
    end
end

```

```

elseif Pos >0
    Data = [Channel 8000 0];
else
    Data = [Channel 4000 0];
end

end

function [] = SendVel(app,Channel)
    Data = GatherData(app,Channel);
    Data(2) = ConvVelocity(app,Data(2));
    Data(3) = ConvAcceleration(app,Data(3));
    Send = BuildCmd(app,Data);
    SendCmd(app,Send);
end

function [] = SendVelY(app,Channel)
    Data = GatherData(app,Channel);
    Data(2) = ConvVelocity(app,Data(2)*2);
    Data(3) = ConvAcceleration(app,Data(3));
    Send = BuildCmd(app,Data);
    SendCmd(app,Send);
end

function [] = SendPos(app,Channel)
    Data = GatherData(app,Channel);
    Pos = Data(3);
    Pos = ConvPosition(app,Data(2),Pos);
    Data(3) = 0;
    Data(2) = ConvVelocity(app,Data(2));
    Send = BuildCmd(app,Data);

```

```

        SendCmd(app, Send);

        disp(Pos);

        pause(Pos);

        SendStop(app, Channel);

end

function [] = SendPosY(app, Channel)

    Data = GatherData(app, Channel);

    Pos = Data(3);

    Pos = ConvPositionY(app, Data(2), Pos);

    Data(3) = 0;

    Data(2) = ConvVelocity(app, Data(2)*3);

    Send = BuildCmd(app, Data);

    SendCmd(app, Send);

    disp(Pos);

    pause(Pos);

    SendStop(app, Channel);

end

function [] = SendStop(app, Channel)

    Data = GetDir(app, Channel, 0);

    Send = BuildCmd(app, Data);

    SendCmd(app, Send);

end

function [Vel] = ConvVelocity(app, VelIn)

    Vel = 6000+(VelIn/12)*2000;

end

function [Accel] = ConvAcceleration(app, AccIn)

    Accel = 255*AccIn/100;

```

```

end

function [Pos] = ConvPosition(app, VelIn, PosIn)
    Pos = PosIn/(13*VelIn);
    Pos = abs(Pos);
end

function [Pos] = ConvPositionY(app, VelIn, PosIn)
    Pos = PosIn/(15.5*VelIn);
    Pos = abs(Pos);
end

end

methods (Access = private)
    % Code that executes after component creation
    function startupFcn(app)
        Port = PortOnBoot(app);
        SetupSerial(app, Port);
        clc;
    end

    % Selection changed function: ButtonGroup2
    function ButtonGroup2SelectionChanged(app, event)
        if app.PredkoscButton2.Value == 1
            app.Przyspieszenie2Label.Text = 'Przyspieszenie [%]';
            app.Acc2.Limits = [0 100];
            app.Acc2.Value = 0;
            app.Parameter2.Value = 0;
        else
            app.Przyspieszenie2Label.Text = 'Przemieszczenie [mm]';
            app.Acc2.Limits = [-500 500];
            app.Acc2.Value = 0;
            app.Parameter2.Value = 0;
        end
    end
end

```

```

        end

    end

    % Selection changed function: ButtonGroup
    function ButtonGroupSelectionChanged(app, event)

        if app.PredkoscButton.Value == 1

            app.PrzyspieszenieLabel.Text = 'Przyspieszenie [%]';
            app.Acc.Limits = [-100 100];
            app.Acc.Value = 0;
            app.Parameter1.Value = 0;

        else

            app.PrzyspieszenieLabel.Text = 'Przemieszczenie [mm]';
            app.Acc.Limits = [10 1000];
            app.Acc.Value = 1000;
            app.Parameter1.Value = 0;

        end

    end

    % Button pushed function: SterujButton2
    function SterujButton2Pushed(app, event)

        if app.PredkoscButton2.Value == 1

            SendVelY(app,1);

        elseif app.PozycjaButton2.Value == 1

            SendPosY(app,1);

        end

    end

    % Button pushed function: SterujButton
    function SterujButtonPushed(app, event)

        if app.PredkoscButton.Value == 1

            SendVel(app,0);

        elseif app.PrzemieszczenieButton.Value == 1

            SendPos(app,0);

        end

    end

```



```

end

% Button pushed function: Stop2Button
function Stop2ButtonPushed(app, event)
    SendStop(app,1);
end

% Button pushed function: StopButton
function StopButtonPushed(app, event)
    SendStop(app,0);
end

% Close request function: UIFigure
function UIFigureCloseRequest(app, event)
    delete(app);
    delete(instrfind);
end

end

% App initialization and construction
methods (Access = private)

    % Create UIFigure and components
    function createComponents(app)

        % Create UIFigure
        app.UIFigure = uifigure;
        app.UIFigure.Position = [100 100 307 352];
        app.UIFigure.Name = 'UI Figure';
        app.UIFigure.CloseRequestFcn = createCallbackFcn(app,
@UIFigureCloseRequest, true);

        % Create KanalPanel
        app.KanalPanel = uipanel(app.UIFigure);
        app.KanalPanel.TitlePosition = 'centertop';
        app.KanalPanel.Title = 'Kanal 1';
        app.KanalPanel.Position = [11 183 287 160];

        % Create ButtonGroup

```

```

app.ButtonGroup = uibuttongroup(app.KanalPanel);
app.ButtonGroup.SelectionChangedFcn = createCallbackFcn(app,
@ButtonGroupSelectionChanged, true);
app.ButtonGroup.Position = [5 72 130 61];
% Create PredkoscButton
app.PredkoscButton = uiradiobutton(app.ButtonGroup);
app.PredkoscButton.Text = 'Prędkość';
app.PredkoscButton.Position = [11 34 73 15];
app.PredkoscButton.Value = true;
% Create PrzemieszczenieButton
app.PrzemieszczenieButton = uiradiobutton(app.ButtonGroup);
app.PrzemieszczenieButton.Text = 'Przemieszczenie';
app.PrzemieszczenieButton.Position = [11 12 115 15];
% Create PrzyspieszenieLabel
app.PrzyspieszenieLabel = uilabel(app.KanalPanel);
app.PrzyspieszenieLabel.HorizontalAlignment = 'center';
app.PrzyspieszenieLabel.Position = [137 70 150 15];
app.PrzyspieszenieLabel.Text = 'Przyspieszenie [%]';
% Create Acc
app.Acc = uieditfield(app.KanalPanel, 'numeric');
app.Acc.Limits = [0 100];
app.Acc.ValueDisplayFormat = '%.2f';
app.Acc.HorizontalAlignment = 'center';
app.Acc.Position = [150 41 127 22];
% Create Parameter1
app.Parameter1 = uieditfield(app.KanalPanel, 'numeric');
app.Parameter1.Limits = [-12 12];
app.Parameter1.ValueDisplayFormat = '%.2f';
app.Parameter1.HorizontalAlignment = 'center';
app.Parameter1.Position = [151 91 126 22];
% Create SterujButton

```

```

app.SterujButton = uibutton(app.KanalPanel, 'push');
app.SterujButton.ButtonPushedFcn = createCallbackFcn(app,
@SterujButtonPushed, true);

app.SterujButton.Position = [151 8 126 24];
app.SterujButton.Text = 'Steruj X';

% Create Parameter1Label
app.Parameter1Label = uilabel(app.KanalPanel);
app.Parameter1Label.HorizontalAlignment = 'center';
app.Parameter1Label.Position = [164 118 98 15];
app.Parameter1Label.Text = 'Prędkość [m/min]';

% Create StopButton
app.StopButton = uibutton(app.KanalPanel, 'push');
app.StopButton.ButtonPushedFcn = createCallbackFcn(app,
@stopButtonPushed, true);

app.StopButton.BackgroundColor = [1 0 0];
app.StopButton.FontSize = 20;
app.StopButton.FontColor = [1 1 1];
app.StopButton.Position = [8 8 124 55];
app.StopButton.Text = 'Stop X';

% Create Kana2Panel
app.Kana2Panel = uipanel(app.UIFigure);
app.Kana2Panel.TitlePosition = 'centertop';
app.Kana2Panel.Title = 'Kanał 2';
app.Kana2Panel.Position = [11 13 287 160];

% Create ButtonGroup2
app.ButtonGroup2 = uibuttongroup(app.Kana2Panel);
app.ButtonGroup2.SelectionChangedFcn = createCallbackFcn(app,
@ButtonGroup2SelectionChanged, true);

app.ButtonGroup2.Position = [5 72 130 61];

% Create PredkoscButton2
app.PredkoscButton2 = uiradiobutton(app.ButtonGroup2);
app.PredkoscButton2.Text = 'Prędkość';

```

```

app.PredkoscButton2.Position = [11 34 73 15];
app.PredkoscButton2.Value = true;
% Create PozycjaButton2
app.PozycjaButton2 = uiradiobutton(app.ButtonGroup2);
app.PozycjaButton2.Text = 'Przemieszczenie';
app.PozycjaButton2.Position = [11 12 115 15];
% Create Przyspieszenie2Label
app.Przyspieszenie2Label = uilabel(app.Kana2Panel);
app.Przyspieszenie2Label.HorizontalAlignment = 'center';
app.Przyspieszenie2Label.Position = [134 70 153 15];
app.Przyspieszenie2Label.Text = 'Przyspieszenie [%]';
% Create Acc2
app.Acc2 = uieditfield(app.Kana2Panel, 'numeric');
app.Acc2.Limits = [0 100];
app.Acc2.ValueDisplayFormat = '%.2f';
app.Acc2.HorizontalAlignment = 'center';
app.Acc2.Position = [152 41 126 22];
% Create Parameter2
app.Parameter2 = uieditfield(app.Kana2Panel, 'numeric');
app.Parameter2.Limits = [-4 4];
app.Parameter2.ValueDisplayFormat = '%.2f';
app.Parameter2.HorizontalAlignment = 'center';
app.Parameter2.Position = [152 91 126 22];
% Create SterujButton2
app.SterujButton2 = uibutton(app.Kana2Panel, 'push');
app.SterujButton2.ButtonPushedFcn = createCallbackFcn(app,
@SterujButton2Pushed, true);
app.SterujButton2.Position = [150 10 126 24];
app.SterujButton2.Text = 'Steruj Y';
% Create Parameter2Label
app.Parameter2Label = uilabel(app.Kana2Panel);

```

```

    app.Parameter2Label.HorizontalAlignment = 'center';
    app.Parameter2Label.Position = [163 118 98 15];
    app.Parameter2Label.Text = 'Prędkość [m/min]';
    % Create Stop2Button
    app.Stop2Button = uibutton(app.Kana2Panel, 'push');
    app.Stop2Button.ButtonPushedFcn = createCallbackFcn(app,
@Stop2ButtonPushed, true);
    app.Stop2Button.BackgroundColor = [1 0 0];
    app.Stop2Button.FontSize = 20;
    app.Stop2Button.FontColor = [1 1 1];
    app.Stop2Button.Position = [8 10 123 53];
    app.Stop2Button.Text = 'Stop Y';
end
end
methods (Access = public)
    % Construct app
    function app = Sterowanie
        % Create and configure components
        createComponents(app)
        % Register the app with App Designer
        registerApp(app, app.UIFigure)
        % Execute the startup function
        runStartupFcn(app, @startupFcn)
        if nargin == 0
            clear app
        end
    end
end
% Code that executes before app deletion
function delete(app)
    % Delete UIFigure when app is deleted
    delete(app.UIFigure)
end

```

```
        end
    end
end
```

Program 3 Program użyty w Pololu Micromaestro

```
begin
    button_a if sequence_a endif
    button_b if sequence_b endif

repeat

sub button_a
    4 get_position 500 GREATER_THAN
    return

sub button_b
    5 get_position 500 GREATER_THAN
    return

sub sequence_a
    4000 1 servo 700 delay
    6000 1 servo
    return

sub sequence_b
    8000 1 servo 700 delay
    6000 1 servo
    Return
```

Spis rysunków:

Rysunek 1.1. Suwnica bramowa. [8]	7
Rysunek 1.2. Suwnica półbramowa. [2]	8
Rysunek 1.3. Suwnica Pomostowa. [9].....	8
Rysunek 1.4. Suwnica wysięgnikowa. [10]	9
Rysunek 2.1. Algorytm konstruowania modelu suwnicy bramowej.	10
Rysunek 2.2. Założone wymiary gabarytowe modelu.	11
Rysunek 3.1. Schemat przedstawiający belkę użytą do ustalenia profilu zastosowanego do konstrukcji modelu.....	12
Rysunek 3.2. Schemat belki z podziałem na dwa przedziały.	14
Rysunek 3.3. Wykres sił tnących dla aluminiowego profilu konstrukcyjnego 40x40.....	16
Rysunek 3.4. Wykres przedstawiający siły tnące dla aluminiowej rury kwadratowej 40x40x2.	17
Rysunek 3.5. Wykres momentów gnących dla profilu konstrukcyjnego.....	18
Rysunek 3.6. Wykres momentów gnących dla aluminiowej rury kwadratowej 40x40x2.....	19
Rysunek 3.7. Wykres ugięcia belki dla aluminiowego profilu konstrukcyjnego 40x40.	20
Rysunek 3.8. Wykres ugięcia belki dla aluminiowej rury kwadratowej 40x40x2.....	21
Rysunek 3.9. Belka użyta do analizy statycznej w programie SOLIDWORKS.....	22
Rysunek 3.10. Ugięcie belki z programu SOLIDWORKS.....	22
Rysunek 3.11. Model użyty do obliczeń metodą elementów skończonych.....	23
Rysunek 3.12. Obciążenia zewnętrzne użyte do obliczeń w programie SOLIDWORKS.....	23
Rysunek 3.13. Ugięcie ramy suwnicy.....	24
Rysunek 3.14. Naprężenia w ramie modelu.....	25
Rysunek 3.15. Model CAD suwnicy bramowej.....	26
Rysunek 4.1. Schemat koła napędowego użytego do obliczenia momentu napędowego.	27
Rysunek 4.2. Serwomechanizm użyty w suwnicy bramowej.	29
Rysunek 5.1. Główny algorytm sterowania.	30
Rysunek 5.2. Interfejs aplikacji sterującej.	31
Rysunek 6.1. Złożony model suwnicy bramowej.	33

Spis tabel:

Tabela 6.1. Tabela wyników pomiarów prędkości.....	34
Tabela 6.2. Tabela wyników uzyskanych podczas pomiarów ugięcia belki.....	35