

Projekt robota trójnożnego

Jakub Mazur

6 listopada 2022

Spis treści

1 Wstęp	1
1.1 Historia robotów kroczących	1
1.2 Istniejące konstrukcje trójnożne	2
1.2.1 STrIDER	2
1.2.2 Triped "Martian"	3
1.2.3 Inne konstrukcje	3
2 Model Matematyczny	4
2.1 Noga robota typu RRR	4
2.1.1 Kinematyka Prosta	5
2.1.2 Kinematyka prosta - metoda Denavit Hartenberga [6]	5
2.1.3 Invert kinematics	7
2.2 Cały Robot	8
2.2.1 Matematyka kroku	8
3 Model CAD i druk 3D	9
4 Sekcja Elektroniczna	10
5 Implementacja	10
5.1 Środowisko ROS	10
5.1.1 Schemat implementacji	10
6 Algorytm Chodu	10

1 Wstęp

1.1 Historia robotów kroczących

Pierwsza idea robota kroczącego pojawiła się już pod koniec XV wieku, a narodziła się w głowie nikogo innego jak Leonarda Da Vinci. Od tamtej pory wielu naukowców próbowało tworzyć konstrukcje, które używały nóg zamiast kół. Jednakże, pierwsze faktycznie udane roboty tego typu datuje się dopiero na początek lat 60 ubiegłego wieku. Pojawiły się wtedy pierwsze działające konstrukcje, na przykład robot czteronożny zbudowany przez Josepha Shingleya oraz roboty sześciu i ośmionożne zbudowane przez "Space General Corporation". [1]

Od tamtej pory powstało wiele różnych projektów, które kategoryzuje się w zależności od ilości nóg posiadanych przez robota:

- jednonożne
- dwunożne (Humanoid, chicken-walkers)
- czteronożne (Quadrupedal)
- sześcionożne (Hexapod)

- ośmionożne
- gąsiennicowe

Można tu zaobserwować pewną tendencję spadkową, wraz z upływem czasu widać wzrost udanych konstrukcji o mniejszej ilości nóg. Konstrukcje takie wymagają większej wiedzy naukowców, lepiej dobranych algorytmów ale za to można je skonstruować mniejszym nakładem materiałowym. Stąd naturalne jest dążenie do ograniczania ilości nóg w konstrukcjach robotów kroczących

Pojawia się także inna tendencja w ilości nóg robotów. Prawie wszystkie konstrukcje (poza jedno-nożnymi) opierają się na anatomii zwierząt. Jest to raczej logiczna tendencja, jako że do takich robotów mamy już algorytmy chodu opracowane przez miliony lat ewolucji. Biologiczne "konstrukcje", które nie mają sensu nie przetrwałyby do dziś. [1]

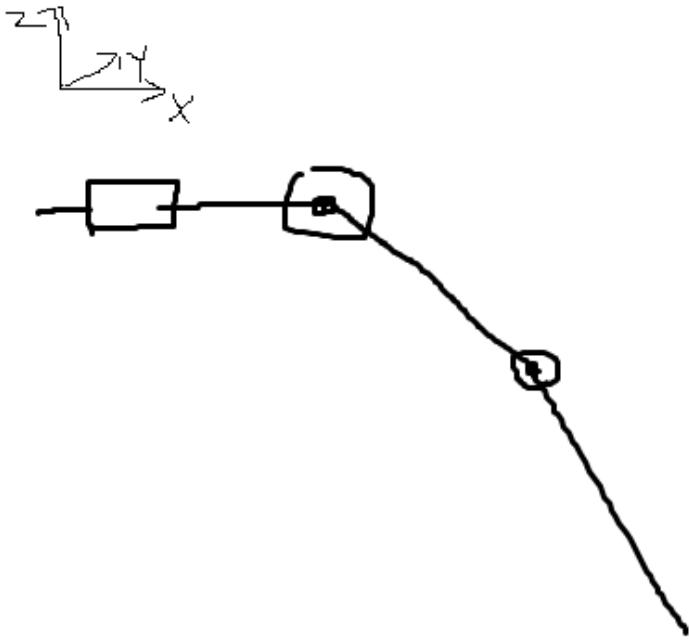
Co natomiast z konstrukcjami robotów trójnożnych? Można się zastanowić czy konstrukcje takie nie powstają ponieważ faktycznie nie mają sensu, czy może dlatego że temat bardziej "klasycznych", prostszych w implementacji, konstrukcji nie został po prostu jeszcze wyczerpany przez naukowców. Jeżeli rozejrzymy się dookoła siebie możemy zaobserwować wiele przedmiotów codziennego użytku które posiadają właśnie trzy nogi, od wszelkich taboretów, przez wieszaki na ubrania po stoły. Są to jednak przedmioty statyczne i dla takich rozwiązań trzy nogi są wymaganym minimum aby dany przedmiot stał stabilnie. Co jednak z konstrukcjami dynamicznymi? Jeżeli robot trójnożny podniesie nogę, straci stabilność, zacznie się przewracać. Czyni to z niego dość ciekawą konstrukcję, gdzie w momencie stania w miejscu zachowuje się bardziej jak roboty o większej ilości nóg, nie przewraca się, a podczas ruchu zachowuje się jak roboty dwunożne, musi odpowiednio szybko odstawić nogę w odpowiednie miejsce aby się nie przewrócić.

1.2 Istniejące konstrukcje trójnożne

Konstrukcje trójnożne pojawiały się w dziełach science fiction już od dawna, od "The War of the Worlds" z 1898 aż po "Mroczne Widmo" z 2001 i regularnie pojawiają się naukowcy, którzy próbują udowodnić że nie trzeba ogarnicać tego typu robotów do dzieł z obszaru science fiction.

1.2.1 STrIDER

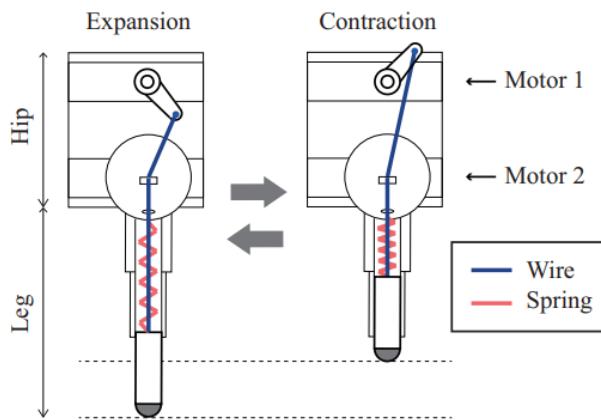
Został zbudowany w 2007 na Uniwersytecie Stanowym w Virginii. Jego celem były eksperymenty z algorytmami chodu i doclewo, prowadzenie obserwacji. Miał to umożliwić długie nogi, które znacznie podwyższały konstrukcję i sprawiały że górną platformą była idealna do instalowania wszelkiego rodzaju urządzeń typu kamery. Kinematykę robota można określić jako $3 - SRRR$, a kinematykę jednej nogi jako RRR . Przy czym "pierwsze" R oznacza obrót dookoła osi x a dwa kolejne R oznaczają obrót wokół osi y (zgodnie z oznaczeniami na rysunku 1) [2]



Rysunek 1: Uproszczony model matematyczny nogi robota STriDER

1.2.2 Triped "Martian"

Jest to robot zbudowany przez Yoichi Masudę na uniwersytecie w Osace w 2017 roku. Kinematyka tego robota opiera się na mechanizmie SLIP (Spring-Loaded-Inverted Pendulum). SLIP ma w pewien sposób symulować sposób poruszania się stosowany przez ludzi i zwierzęta. Sprężyna wewnętrzna nogi robota jest naciągana, co powoduje skracanie się nogi, a zwalnianie sprężyny z powrotem wydłuża człon. Dodatkowo dodane jest serwo, które może obracać nogę dookoła osi y . Czyni to z nogi robota mechanizm o kinematyce typu RL . Został także dodany czujnik naprężenia, który jest w stanie zmierzyć siłę naciągu nici kompresującej sprężynę. [3]



Rysunek 2: Model nogi robota Triped Martian źródło: [3]

1.2.3 Inne konstrukcje

W internecie można także znaleźć kilka różnych, niezbyt dobrze udokumentowanych konstrukcji zakończonych mniejszym lub większym sukcesem:

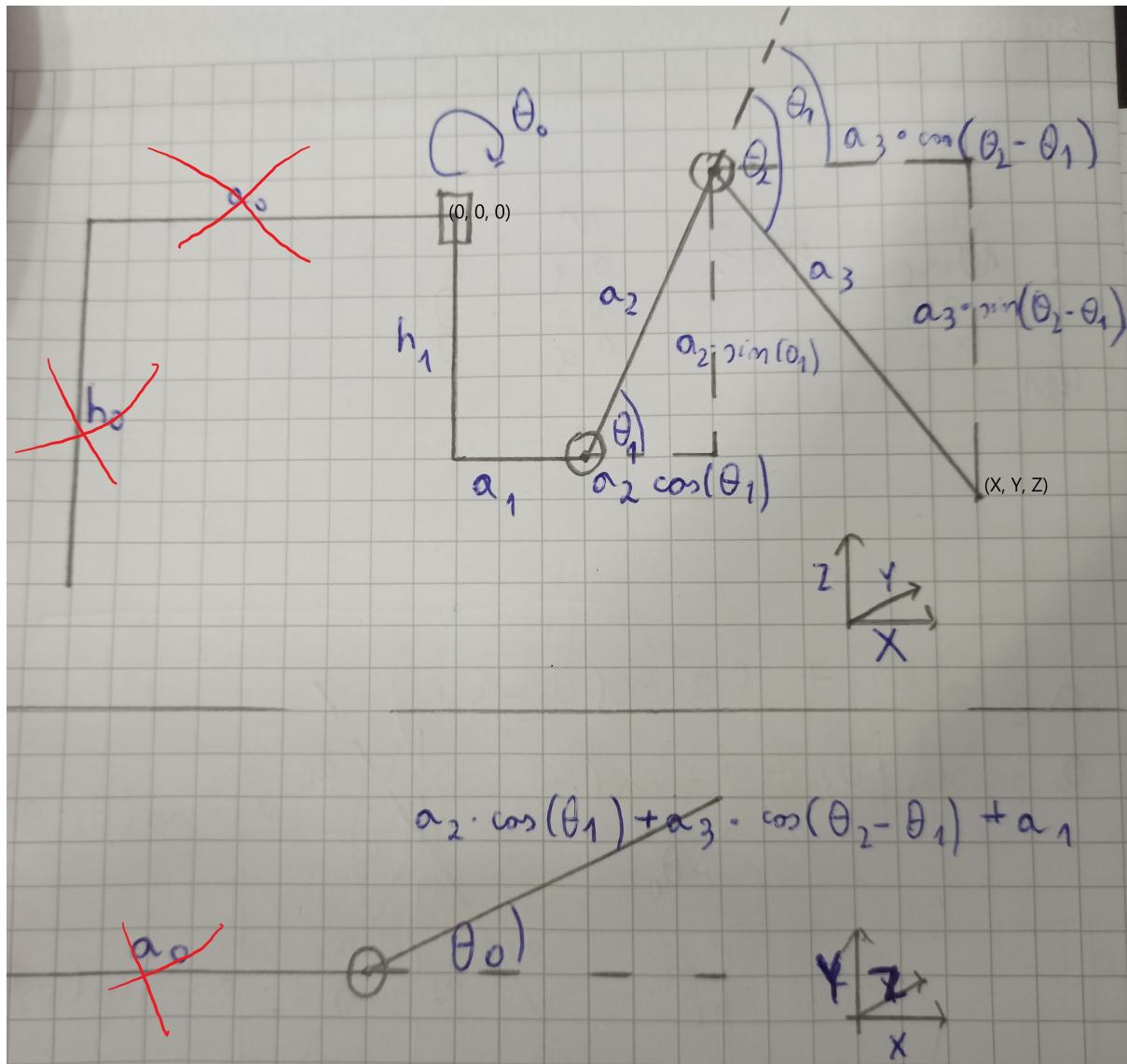
- Makerfaire 3-legged walking robot
- missel tripod robot

Niestety konstrukcje te nie posiadają żadnej dokumentacji technicznej i nie da się dokładnie określić zasady ich działania. Nie można nawet mieć pewności że konstrukcje te faktycznie istnieją a nie są oszustwem lub inną formą naganania rzeczywistości, na co mogłaby wskazywać jakość filmików i uboga ilość materiałów.

2 Model Matematyczny

2.1 Noga robota typu RRR

Noga ma trzy stopnie swobody. Wszystkie są typu obrotowego, co czyni z niej konstrukcję typu *RRR*. Przy czym dwa elementy rotacyjne obracają się dookoła osi poziomej (*X*) (podobnie jak w przypadku robota STrIDER), a jedna dookoła osi pionowej (*Z*). Są to też te same osi obrotu co w przypadku ramienia robotycznego typu antromorficznego (zwanego także "angular" bądź "jointed"). Co za tym idzie, cały model matematyczny jest w zasadzie identyczny jak w przypadku manipulatora tego typu.



Rysunek 3: Model Matematyczny

2.1.1 Kinematyka Prosta

Obliczanie kinematyki prostej polega na uzyskaniu równań końówki ramienia robotycznego we współrzędnych kartezjańskich względem współrzędnych konfiguracyjnych. Inaczej mówiąc, wejściem algorytmu jest zbiór współrzędnych konfiguracyjnych, a na wyjściu otrzymamy współrzędne kartezjańskie.[4]

W przypadku tego konkretnego manipulatora, mamy do czynienia z trójwymiarowym układem współrzędnych kartezjańskich i trzema stopniami swobody. Da to liniowo niezależny układ trzech równań, w którym parametrami będą kąty na jakich mają ustawić się serwomechanizmy a wynikiem wektor współrzędnych kartezjańskich.

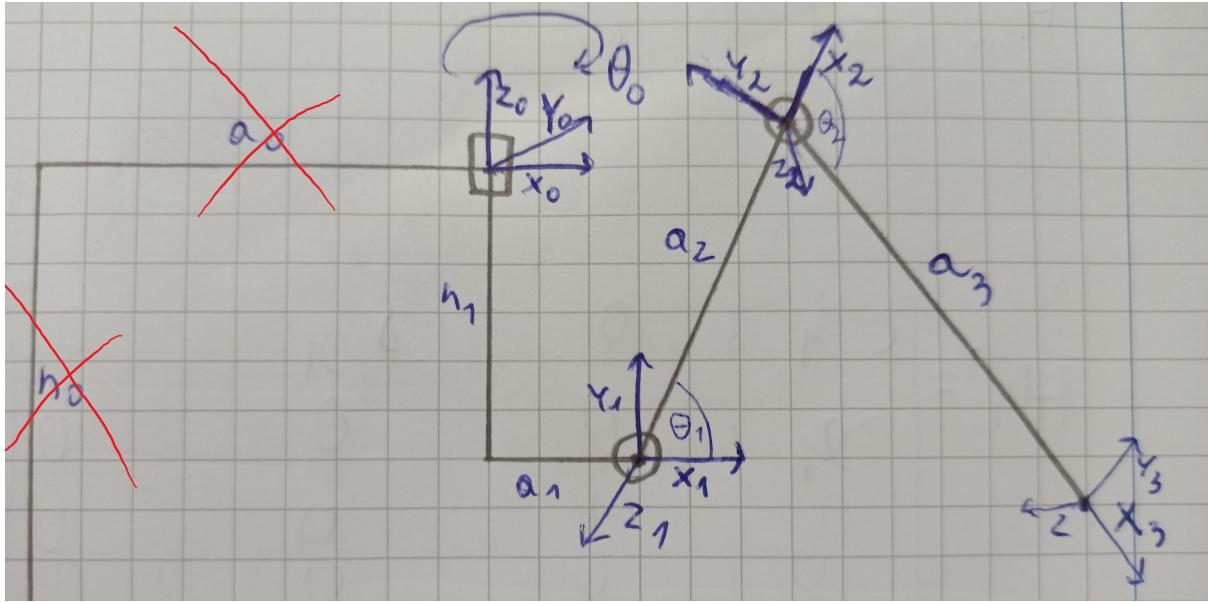
Najprostszą metodą liczenia kinematyki prostej jest rozrysowanie modelu matematycznego i geometryczne wyprowadzenie potrzebnych równań. Model taki dla tego manipulatora został przedstawiony na rysunku 3. Przyjęty początek układu współrzędnych został oznaczony jako $(0, 0, 0)$, a punkt którego współrzędne kartezjańskie są poszukiwane został oznaczony jako (X, Y, Z) . h_1 i a_{1-3} to stałe długości poszczególnych członów nogi. Natomiast θ_{0-2} to właśnie pozycje serwomechanizmów - parametry algorytmu. Na ich podstawie zostaną obliczone współrzędne końówki manipulatora w systemie kartezjańskim. Same obliczenia geometryczne są już w tym momencie dość trywialne, wystarczy do każdego członu a_{1-3} przenieść jego długość na osi X , Y lub Z za pomocą trygonometrii (sinus lub cosinus) i zsumować odpowiednie długości. Da to układ równań 1.

$$\begin{aligned} a_{temp} &= a_2 \cos \theta_1 + a_3 \cos (\theta_2 - \theta_1) + a_1 \\ Y &= a_{temp} \cdot \sin \theta_0 \\ X &= a_{temp} \cdot \cos \theta_0 \\ Z &= a_2 \sin \theta_1 - a_3 \sin (\theta_2 - \theta_1) \end{aligned} \tag{1}$$

2.1.2 Kinematyka prosta - metoda Denavita Hartenberga [6]

Alternatywą dla zwykłych obliczeń geometrycznych jest metoda Denavita Hartenberga. Polega ona na przedstawieniu całkowitego przekształcenia jako iloczynu przekształceń jednorodnych kolejnych członów. Pojedyncze przekształcenie jednorodne składa się natomiast z 6 przekształceń prostych (3 dla rotacji i 3 dla przesunięć). Wymnożenie tych przekształceń prostych da przekształcenie jednorodne. [5]

Metoda ta jest w szczególności użyteczna dla bardzo skomplikowanych manipulatorów, gdzie stopni swobody jest znacznie więcej niż ilość współrzędnych kartezjańskich.



Rysunek 4: Model Denavit Hartenberg

Z praktycznego punktu widzenia obliczenia należy zacząć od stworzenia specjalnego rysunku (Rys. 4) z zaznaczonymi kolejnymi obrotami lokalnych układów współrzędnych, a następnie zebrać odpowiednie transformacje do tabelki (Tab. 1)

Tabela 1: Tabela z parametrami DH

Joint i	θ_i	α_i	r_i	d_i
1	θ_0	$\frac{\pi}{2}$	a_1	h_1
2	θ_1	0	a_2	0
3	θ_2	0	a_3	0

Gdzie:

θ_i - angle from x_{n-1} to x_n around z_{n-1}

α_i - angle from z_{n-1} to z_n around x_n

r_i - distance between the origin of the $n-1$ frame and the origin of the n frame along the x_n direction.

d_i - distance from x_{n-1} to x_n along the z_{n-1} direction

Następnie macierze transformacji jednorodnej (pomiędzy ramkami $n-1$ i n) oblicza się zgodnie ze wzorem 2. Wzór ten jest właśnie obliczony poprzez wymnożenie wzorów na wyżej wspomniane 6 przekształceń prostych. Ale w zasadzie wzór ten sprowadza się do 2 zasadniczych elementów. R jest macierzą Rotacji o rozmiarze 3×3 a T to macierz Transformacji 1×3 .

$$H_n^{n-1} = \left[\begin{array}{c|c} R & T \\ \hline 0 & 0 \\ 0 & 0 \end{array} \right] = \left[\begin{array}{cccc} \cos \theta_i & -\sin \theta_i \cdot \cos \alpha_i & \sin \theta_i \cdot \sin \alpha_i & a_i \cdot \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cdot \cos \alpha_i & -\cos \theta_i \cdot \sin \alpha_i & a_i \cdot \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{array} \right] [7] \quad (2)$$

Następnie należy podstawić wartości dla każdego z 3 punktów i otrzymane macierze wymnożyć zgodnie ze wzorem 3.

$$H_n^{n-1} = H_1^0 \cdot H_2^1 \cdot H_3^2 =$$

$$\begin{bmatrix} c\theta_0 \cdot (c\theta_1 \cdot c\theta_2 - s\theta_1 \cdot s\theta_2) & -c\theta_0 \cdot (c\theta_1 \cdot s\theta_2 + s\theta_1 \cdot c\theta_2) & s\theta_0 & c\theta_0 \cdot (a_1 + a_2 \cdot c\theta_1 + a_3 \cdot (c\theta_1 \cdot c\theta_2 - s\theta_1 \cdot s\theta_2)) \\ s\theta_0 \cdot (c\theta_1 \cdot c\theta_2 - s\theta_1 \cdot s\theta_2) & -s\theta_0 \cdot (c\theta_1 \cdot s\theta_2 + s\theta_1 \cdot c\theta_2) & -c\theta_0 & s\theta_0 \cdot (a_1 + a_2 \cdot c\theta_1 + a_3 \cdot (c\theta_1 \cdot c\theta_2 - s\theta_1 \cdot s\theta_2)) \\ c\theta_1 \cdot s\theta_2 + s\theta_1 \cdot c\theta_2 & c\theta_1 \cdot c\theta_2 - s\theta_1 \cdot s\theta_2 & 0 & h_1 + a_2 \cdot s\theta_1 + a_3 \cdot (c\theta_1 \cdot s\theta_2 + s\theta_1 \cdot c\theta_2) \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} c\theta_0 \cdot c(\theta_1 + \theta_2) & -c\theta_0 \cdot s(\theta_1 + \theta_2) & s\theta_0 & c\theta_0 \cdot (a_1 + a_2 \cdot c\theta_1 + a_3 \cdot c(\theta_1 + \theta_2)) \\ s\theta_0 \cdot c(\theta_1 + \theta_2) & -s\theta_0 \cdot s(\theta_1 + \theta_2) & -c\theta_0 & s\theta_0 \cdot (a_1 + a_2 \cdot c\theta_1 + a_3 \cdot c(\theta_1 + \theta_2)) \\ c\theta_1 \cdot s\theta_2 + s\theta_1 \cdot c\theta_2 & c(\theta_1 + \theta_2) & 0 & h_1 + a_2 \cdot s\theta_1 + a_3 \cdot s(\theta_1 + \theta_2) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

Następnie, aby otrzymać właściwe równania, należy z równania 3 wziąć część odpowiedzialną za transformację. Efektem jest wzór 4, czyli finalna wersja równań kinametyki prostej obliczonej metodą Denavita Hartenberga

$$\begin{aligned} X &= c\theta_0 \cdot (a_1 + a_2 \cdot c\theta_1 + a_3 \cdot c(\theta_1 + \theta_2)) \\ Y &= s\theta_0 \cdot (a_1 + a_2 \cdot c\theta_1 + a_3 \cdot c(\theta_1 + \theta_2)) \\ Z &= h_1 + a_2 \cdot s\theta_1 + a_3 \cdot s(\theta_1 + \theta_2) \end{aligned} \quad (4)$$

TODO - znak się nie zgadza

2.1.3 Invert kinematics

Kinematyka odwrotna to TODOTODOTODO

Zwykle odwrotną kinematykę można obliczyć poprzez rozwiązywanie równań kinametyki prostej. Jest to w tym przypadku także teoretycznie możliwe, jako że mamy do czynienia z trzema niewiadomymi (θ_{0-2}) i układem trzech równań nr. 1 lub 4. (Równanie na a_{temp} jest jedynie pomocnicze, trzeba je traktować jak część równań Y i X).

Jest to jak najbardziej wykonalne w przypadku θ_0 , można to zrobić łącząc wzór na X i Y , dzieląc go obustronnie, zamieniając $\frac{\sin}{\cos}$ na tan i wyciągając θ_0 na lewą stronę. Daje to wzór 5

$$\theta_0 = \arctan \frac{Y}{X} \quad (5)$$

Większy problem pojawia się jednak w przypadku obliczania θ_1 i θ_2 , ponieważ wartości te da się obliczyć z wyżej wspomnianego równania, ale nie da się wyznaczyć na te wartości równań w postaci jawnej. A bez postaci jawnej poprawna implementacja tych równań będzie znacznie utrudniona.

Można natomiast zastosować pewnego rodzaju uproszczenie. Jeżeli wyznaczanie równań θ_1 i θ_2 sprawdzi się do problemu dwuwymiarowego, obliczenia stają się w zasadzie identyczne jak w przypadku obliczeń kinametyki odwrotnej dla robota typu SCARA, co było już robione wielokrotnie.

Aby obliczyć kąt θ_2 można powrócić do rysunku 3 i zastosować na trójkącie utworzonym z a_2 i a_3 twierdzenie cosinusów połączone z twierdzeniem pitagorasa. Daje to wzór 6.

$$(x - a_1)^2 + (z + h_1)^2 = a_2^2 + a_3^2 - 2 \cdot a_2 \cdot a_3 \cdot \cos(180^\circ - \theta_2) \quad (6)$$

Następnie na podstawie równania 6 aby ostatecznie otrzymać θ_2 należy zastosować wzór $\cos(180^\circ - \theta) = -\cos \theta$ i za pomocą prostych przekształceń wyciągnąć z równania θ_2 . Daje to wzór 7

$$\theta_2 = \arccos \left(\frac{(x - a_1)^2 + (z + h_1)^2 - a_2^2 - a_3^2}{2a_2a_3} \right) [9] \quad (7)$$

Natomiast obliczenia dla θ_1 TODO

$$\theta_1 = \arctan \frac{z + h_1}{x - a_1} + \arcsin \left(\frac{a_3 \sin \theta_2}{\sqrt{(x - a_1)^2 + (z + h_1)^2}} \right) [9] \quad (8)$$

Ostatecznie zbierając równania 5, 8, 7 otrzymujemy układ równań który stanowi kinematykę odwrotną opisywanej nogi robotycznej.

TODO - układ równań

TODO - ograniczenia.

Natomiast trzeba cały czas pamiętać że jest to tylko uproszczenie, które będzie owocować pewnymi drobnymi błędami. Trzeba zauważyc, że zmiana współrzędnej Y nie wpływa na wartości kątów θ_1 i θ_2 , tylko na θ_0 . Oznacza to, że wraz ze zmianą kąta θ_0 , na potrzeby uproszczenia obliczeń, obraca się także o ten kąt oś X układu współrzędnych. W praktyce oznacza to, że odsunięcie nogi od punktu $(0, 0, 0)$ zawsze będzie rzutowane na cylinder o promieniu równym odsunięciu o takim samym X i Z , ale $Y = 0$. Problem jest zaprezentowany na rysunku XXX. Jednakże błąd powinien być pomijalnie mały a równania powinny umożliwić skuteczną implementację algorytmu chodu.

2.2 Cały Robot

Zwykle konstruując roboty wzorujemy się na zwierzętach występujących w naturze. W tym przypadku nie mamy jednak tego luksusu, dlatego na początek należy przyjąć jakieś najbardziej intuicyjne założenie. Dlatego też uznałem że najlepszym rozwiążaniem będzie rozmieścić nogi na jednej płaszczyźnie w równych odstępach - co 120 deg.

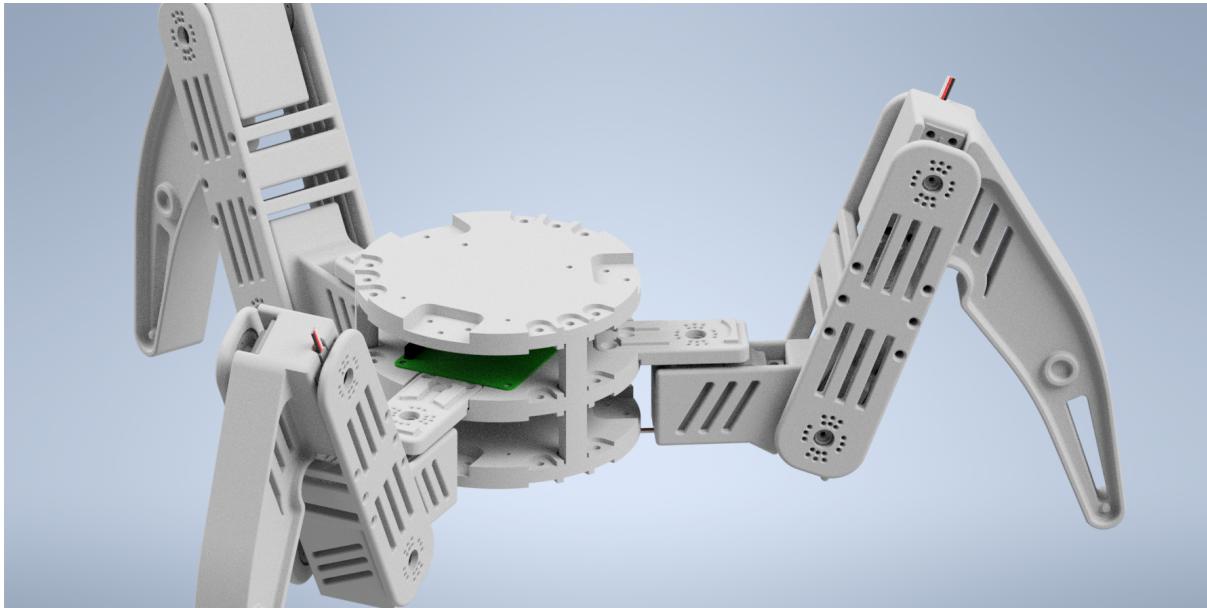
W poprzednim podrozdziale przyjąłem osie układu relatywne do ułożenia początkowego nogi robota. Oznacza to że robot trójnożny będzie miał 3 niezależne osie x i 3 niezależne osie y , tylko oś z zgadza się między kolejnymi nogami. Idzie za tym konieczność stworzenia pewnej metody "obracania" wszystkich tych osi x i y do jednego zunifikowanego układu współrzędnych. Pozwoli to obliczyć kinematykę całego robota.

2.2.1 Matematyka kroku

Aby móc w prosty sposób napisać później algorytm chodu i móc się skupić na kolejności przestawiania nóg przez robota, długości i wysokości kroku i innych tego typu aspektach, należy teraz dobrze ten krok sparametryzować. Na podstawie moich poprzednich doświadczeń z podobnymi projektami, uważam że najbardziej intuicyjne będzie sparametryzowanie kroku do:

- wartości kąta pomiędzy kierunkiem "do przodu" kroku a osią x danej nogi
- długości kroku
- wysokości kroku

3 Model CAD i druk 3D



Rysunek 5: Model Złożeniowy

Jednym z głównych założeń projektu była możliwość wydrukowania całej konstrukcji w 3D. Miało to na celu znaczne obniżenie kosztów produkcji, ale przede wszystkim umożliwić znacznie szybsze przeróbki. Jest to o tyle istotne, że prowadzone będą badania z algorytmami chodu - przypadku większości algorytmów wydłużenie pewnych elementów nóg może zmniejszyć wymagane prędkości ruchu serw. Eksperymenty takie mogą wymusić liczne przedruki poszczególnych członów nóg robota.

Drugim założeniem projektowym stojącym za konstrukcją taką jak na rysunku 5 jest modularność. Bardzo podobną modularność oferuje robot TurtleBot 3, który także był inspiracją stojącą za konstrukcją. Zarówno Turtlebot jak i konstrukcja budowana w ramach tej pracy składają się z wielu identycznych warstw, które zawierają liczne otwory montażowe umożliwiające przykręcenie w zasadzie dowolnego elementu.

Zostały także dokładnie zwymiarowane otwory montażowe znajdujące się na orczykach do zakupionych serw i przeniesione na poszczególne elementy nóg. Do montażu wspomnianych orczyków zakupione zostały śruby M1.6, co stanowi swojego rodzaju drobny eksperyment. Zwykle orczyki montuje się za pomocą kleju lub wkrętów dostarczanych wraz z serwomechanizmem. Są to jednak metody przynajmniej częściowo destrukcyjne - nie umożliwiają szybkiego demontażu i wymiany elementów, co jest bardzo ważnym elementem tego projektu. Zastosowanie śrub z nakrętkami rozwiązuje ten problem - jednak pojawia się pytanie czy nie generuje to innych problemów? (TODO - tutaj opisać problemy i później we wnioskach odpowiedź czy generuje czy nie czy od razu tutaj polecieć że było git?)

Oryginalny projekt zawiera także pewną wadę konstrukcyjną - jest to bardzo cienki element łączący nogę z tułowiem (element nogi zero). Istnieje duże ryzyko gięcia a może nawet łamania się wyżej wymienionego elementu. Aby zmniejszyć to ryzyko, wspomniany element został miejscami pogrubiony i zakupione zostały (TODO - trzeba kupić) metalowe orczyki. Najtrwalszym rozwiązaniem oczywiście byłoby stworzenie dodatkowego elementu który byłby przytwierdzony do dolnej części obudowy i posiadałby oś obrotu z pierwszym elementem nogi - "podtrzymywałby" ten element od dołu.

Całość konstrukcji została zaprojektowana w programie Autodesk Inventor. Program ten został wybrany tylko i wyłącznie ze względu na fakt, że był autorowi projektu dość dobrze znany. [TODO - opisać problem że szkic się "rozjeżdżały" podczas modyfikowania szkiców "poziomy" niżej czy nie?]

Modele zostały wydrukowane na drukarce Zortrax M200. Została ona wybrana ze względu na jej dostępność na uczelni. Dodatkowo, aby przygotować pliki pod druk, należało model przetworzyć programem Z-Suite. W programie większość ustawień pozostawiana była bez zmian, jedynie 2 istotne ustawienia zostały dostosowane do projektu. Zostało ustawione minimalne niezerowe wypełnienie, około 10% i warstwy, wierzchnia i spodnia, zostały ustawione na najgrubszą możliwą opcję. Ustawienia te znalezione zostały eksperymentalnie - wydają się najlepiej balansować między wytrzymałością a czasem druku i ilością zużytego materiału. Dodatkowo, przed ostatecznym wydrukiem, wewnątrz programu inventor, zwiększone zostały o około 0.3 – 0.4mm wszystkie otwory montażowe, ponieważ modele "puchną" i dziury te po wydruku były znacznie mniejsze niż na tworzonych szkicach.

4 Sekcja Elektroniczna

Projekt elektroniczny na potrzeby tego projektu został ograniczony do minimum. Zasilanie składa się z akumulatora typu LiPo i minimum 2 przetwornic. Jako mózg urządzenia zastosowany został minikomputer Raspberry Pi 4B i to do jego zasilenia potrzebna jest jedna z przetwornic. Zgodnie z dokumentacją Raspberry zasilanie jest ze źródła o napięciu 5V i prądzie przynajmniej 3A.[10] Dlatego została zainstalowana przetwornica TODO.

Druga przetwornica ma za zadanie zasilić serwomechanizmy. Serwomechanizmy wybrane do tego projektu to Feetech FT5715M (sztuk 3) i PowerHD LF-20MG (sztuk 6). W czasie zakupu serwa te wypadały najlepiej spośród wszystkich dostępnych pod kątem prędkości ruchu do ceny. Serwa firmy Feetech są zasilane napięciem z zakresu 4.8 do 6V a power HD napięciem 4.8 do 6.6V. Dlatego jako wspólne napięcie zasilania ustalone została wartość 6V. Jako że zwykle serwo pobiera do około 1A prądu to do ich zasilenia potrzeba przetwornicę o wyjściu 6V i minimum 9A [11]. Natomiast należy pamiętać że dobieranie przetwornicy "na styk" przy czymś takim jak zasilanie serwomechanizmów może spowodować później problemy przy większych obciążeniach. Dlatego, aby uwzględnić pewien zapas prądowy, wybrana została przetwornica TODO

5 Implementacja

5.1 Środowisko ROS

5.1.1 Schemat implementacji

6 Algorytm Chodu

Literatura

- [1] Manuel F. Silva, J. A. Tenreiro Machado (2006) A Historical Perspective of Legged Robots
- [2] Forward and Inverse Displacement Analysis of a Novel Three Legged Mobile Robot Based on the Kinematics of In Parallel Manipulators
- [3] Yoichi Masuda, Masato Ishikawa (2017) Simplified Triped Robot for Analysis of Three-Dimensional Gait Generation
- [4] Richard P. Paul (1981) Robot Manipulators: Mathematics, Programming, and Control : the Computer Control of Robot Manipulators
- [5] Wojciech Paszke Kinematyka prosta: reprezentacja Denavita-Hartenberga
- [6] How to Find Denavit-Hartenberg Parameter Tables, blogpost by Automatic Addison
- [7] Homogeneous Transformation Matrices Using Denavit-Hartenberg, blogpost by Automatic Addison
- [8] Matlab Denavit Hartenberg calculations
- [9] Adam Labuda, Janusz Pomirski, Andrzej Rak (2009) Model manipulatora o dwóch stopniach swobody

- [10] Raspberry Pi 4 Power Requirements: Everything You Need to Know
- [11] Electrical characteristics of servos and introduction to the servo control interface
- [12] Alexander Wallen Kiessling, Niclas Maatta (2020) Anthropomorphic Robot Arm