

# KoNaR

Koło naukowe robotyków KoNaR  
Politechnika Wrocławska

---

## Projekt robota mobilnego klasy minisumo "Stefan"

---

Autor:  
Kamil Winnicki

---

Luty 2023

# Spis treści

<b>1 Wstęp</b>	<b>2</b>
<b>2 Kategoria turniejowa Minisumo</b>	<b>2</b>
<b>3 Założenia projektowe</b>	<b>2</b>
<b>4 Elektronika</b>	<b>3</b>
4.1 Płytki Arduino nano . . . . .	3
4.2 Zasilanie . . . . .	4
4.3 Silniki . . . . .	4
4.4 Czujniki podłożą . . . . .	5
4.5 Czujniki odległości . . . . .	5
4.6 Interfejs użytkownika . . . . .	6
4.7 Starter . . . . .	6
<b>5 Mechanika</b>	<b>7</b>
5.1 Rozmieszczenie elektroniki . . . . .	7
5.2 Model 3D . . . . .	7
5.3 Konstrukcja . . . . .	7
5.4 Koła . . . . .	8
5.5 Dociążenie . . . . .	8
5.6 Pierwszy prototyp . . . . .	8
<b>6 Program</b>	<b>9</b>
6.1 Szkielet Programu . . . . .	9
6.2 Główna pętla programu - algorytm walki . . . . .	9
6.3 Fight . . . . .	10
6.4 Search . . . . .	10
6.5 Skręty, obroty i zawroty . . . . .	10
<b>7 Aktualizacje robota</b>	<b>11</b>
7.1 Nowe koła . . . . .	11
7.2 "Technologia wachlarzy" - płachta na byka . . . . .	11
7.3 Dodatkowe czujniki odległości . . . . .	12
7.4 Nowa płytka z elektroniką . . . . .	12
<b>8 Wady konstrukcyjne</b>	<b>12</b>
<b>9 Zawody</b>	<b>13</b>
9.1 RoboChallange Bukareszt 5-6.11.2022 . . . . .	13
9.2 Robo Motion Rzeszów 26.11.2022 . . . . .	13



## 1 Wstęp

Pierwszy pomysł na powstanie robota powstał po zawodach robotycznych w Rybniku w 2022 roku. Stworzenie robota było możliwe dzięki dużej pomocy kolegów z koła naukowego robotyków "KoNaR".

## 2 Kategoria turniejowa Minisumo

Zawody robotów w kategorii sumo wzorowane są na japońskich zapasach sumo. Przeciwnikami są dwa autonomiczne roboty, które próbują wypchnąć rywala poza powierzchnię ograniczonego białą linią czarnego koła ringu (tzw. dohyo). Zazwyczaj walki robotów sumo składają się z 3 kilkuminutowych rund a zwycięzcą jest robot, który dwukrotnie wypchnie przeciwnika poza ring. Zawodnicy nie mają kontroli nad robotami. Jedyną rzeczą jaką mogą wykonać jest uruchomienie pilotem procedury startowej. W czasie zmagań robot musi działać już autonomicznie.

## 3 Założenia projektowe

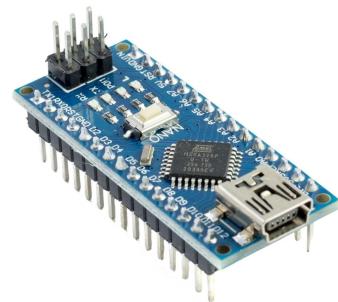
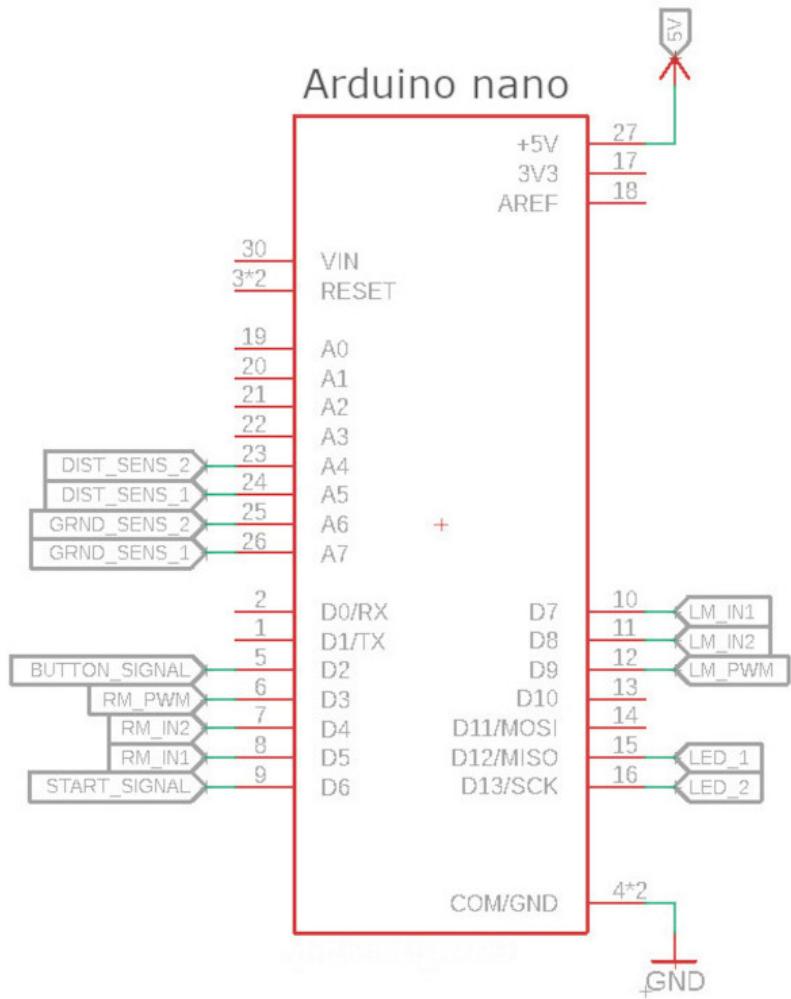
- Spełnienie fizycznych ograniczeń regulaminu kategorii minisumo (rozmiar max 100x100 mm oraz maksymalna waga 500g)
- W pełni autonomiczne działanie robota (poruszanie się, wykrywanie i wypchanie przeciwnika, kontrola krawędzi ringu)
- Sterowanie oparte o mikrokontroler firmy Atmel, Atmega328p-au na płytce Arduino nano
- Dwa silniki sterowane różnicowo
- Dwa czujniki wykrywające przeciwnika
- Dwa czujniki wykrycia linii

## 4 Elektronika

Elektronika została wykonana na płytce uniwersalnej, jednostronnej.

### 4.1 Płytki Arduino nano

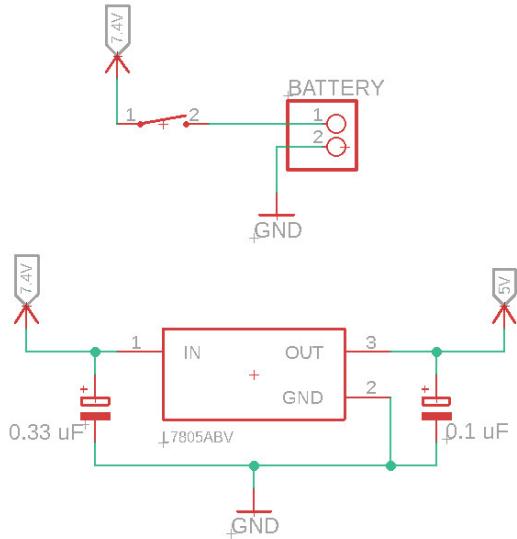
Wybór płytki Arduino nano jako głównego sterowania został dokonany ze względu na uproszczenie wykonania elektroniki oraz znajomość środowiska Arduino.



Rysunek 2: Płytki Arduino na-no

Rysunek 1: Schemat podłączeń do płytka

## 4.2 Zasilanie



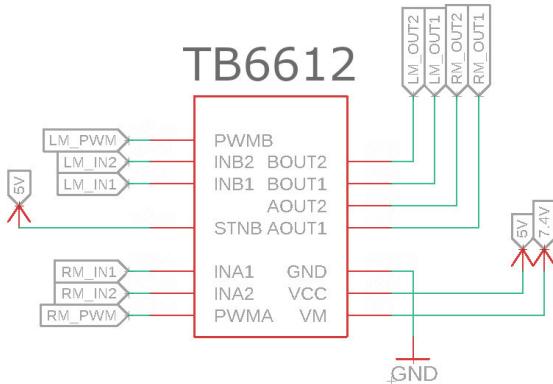
Zasilanie zostało oparte o akumulator litowo - polimerowy 2S 7.4V, o pojemności 800mAh.

Ponieważ mikrokontroler atmega328p-au wymaga zasilania między 2.8V a 5V konieczne jest obniżenie napięcia. Został użyty do tego stabilizator liniowy LM7805. Płytką Arduino nano zawiera wbudowany stabilizator jednak został zastosowany zewnętrzny ze względu na jego niską wydajność prądową. W skrajnych przypadkach spowodowało to przegrzanie stabilizatora, co z kolei mogłoby uszkodzić procesor.

Dodatkowo w celu filtracji zasilania zostały zastosowane dwa kondensatory elektrolityczne.

## 4.3 Silniki

Silniki wybrane do robota to N20-BT17 micro (zamienniki Pololu micro-series), o przekładni 50:1. Jako sterownik silników użyty został TB6612FNG jako gotowy moduł.



Rysunek 3: Schemat podłączenia sterownika



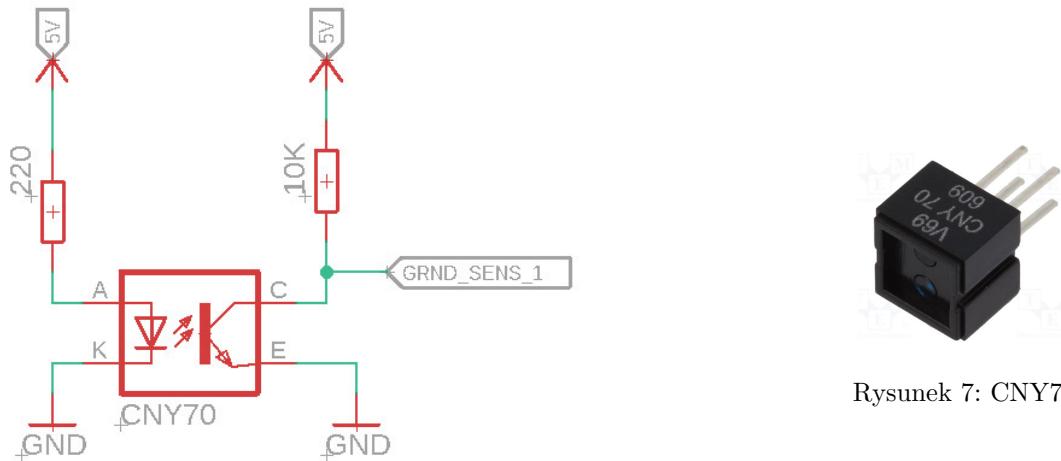
Rysunek 4: Silnik



Rysunek 5: Sterownik TB6612FNG

#### 4.4 Czujniki podłoża

Wykrywanie krawędzi ringu konieczne jest do uniknięcia wypadnięcia robota z ringu. Do tego celu zastosowane zostały czujniki odbiciowe CNY70. Ich działanie opiera się o diode nadawczą oraz fotorezystor. Im światła odbitego jest więcej tym napięcie na wyjściu jest niższe. Krawędź ringu na zawodach pokryta jest białą farbą, kolor ten odbija światło bardziej niż czarne dlatego taki czujnik idealnie się do tego nadaje. W robocie zastosowano dwa takie czujniki, umieszczone z przodu.



Rysunek 7: CNY70

Rysunek 6: Schemat podłączenia czujnika

#### 4.5 Czujniki odległości

W celu wykrywania przeciwnika konieczne było zastosowanie czujników odległości. Wybór padł na analogowe czujniki optyczne SHARP GP2Y0A41SK0F, ponieważ w trakcie budowania robota miały one jeszcze przystępna cenę, są łatwe w obsłudze oraz szybkie w odczytach.

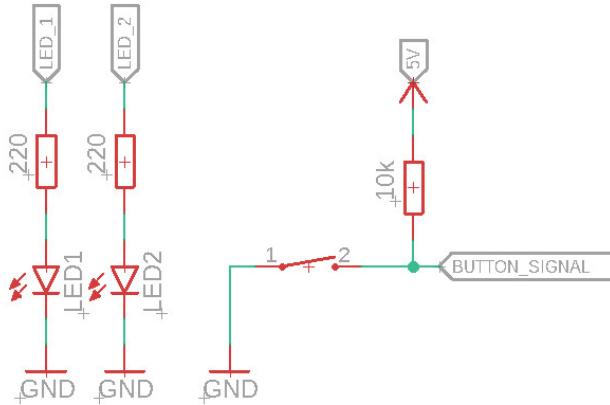


Rysunek 9: Sharp GP2Y0A41SK0F

Rysunek 8: Schemat podłączenia czujników

## 4.6 Interfejs użytkownika

”Interfejs użytkownika” zawiera dwie diody informujące o stanie w jakim znajduje się robot oraz przycisk do wyboru kierunku lub ręcznego startu.



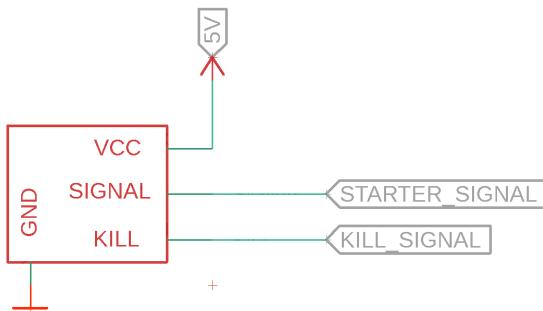
Rysunek 10: Schemat interfejsu

## 4.7 Starter

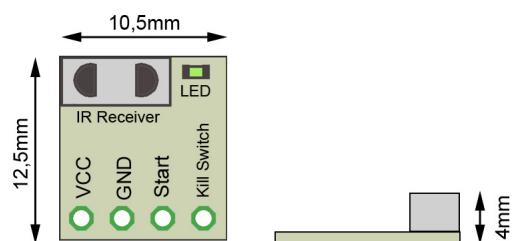
W celu zdalnego wystartowania robotów na zawodach stosuje się starter. Pozwala on na jednoczesne wystartowanie dwóch robotów w tym samym momencie co eliminuje zbyt wcześnie start jednego z robotów. Na zawodach na całym świecie przyjęto ogólny standard starterów (wiedeński). Taki moduł startowy posiada odbiornik fali podczerwonej o częstotliwości 38kHz oraz diode informującą w jakim stanie znajduje się starter. Mrugnięcie dwa razy oznacza, że starter został połączony z pilotem, brak świecenia - czekanie na START, ciągłe świecenie - oznacza START, ciągłe mruganie - stan STOP.

Moduł startowy zasilany może być napięciem od 3.3V do 5V. Posiada dwa pin'y sygnałowe: SIGNAL - podający stan wysoki w trakcie stanu START, oraz stan niski w trybie czekania i STOP. Kolejny pin to KILL, jednak jest on obowiązkowy tylko w kategorii MegaSumo, dlatego w tym robocie nie został połączony.

Więcej informacji na temat startera można znaleźć na stronie oraz GitHubie twórcy.



Rysunek 11: Schemat podłączenia startera



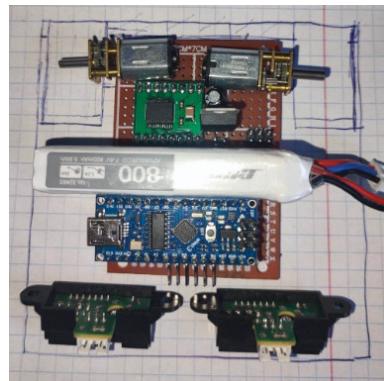
Rysunek 12: Starter

## 5 Mechanika

Głównymi założeniami przy projektowaniu mechaniki było stworzenie robota możliwie jak najniższego oraz posiadającego obudowę z blachy.

### 5.1 Rozmieszczenie elektroniki

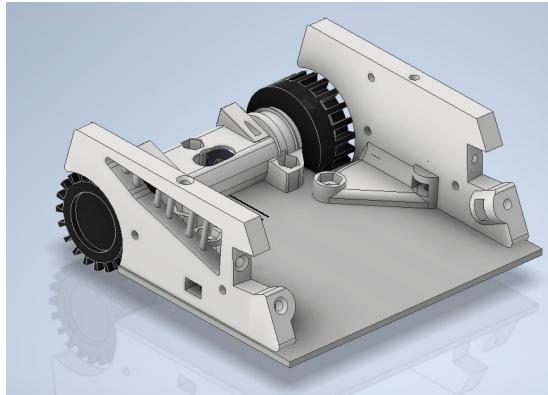
Przy etapie tworzenia płytka z elektroniką należało uwzględnić odpowiednie rozmieszczenie elementów w taki sposób aby robot mógłby być możliwie jak najniższy. Niestety nie udało się tego osiągnąć w dużym stopniu przez baterię która okazała się zbyt duża.



Rysunek 13: Rozmieszczenie komponentów w srodku robota

### 5.2 Model 3D

W pierwszej kolejności projektowania mechaniki robota został stworzony pomocniczy model 3D który miał zapewnić łatwiejszą pracę na dalszym etapie.



Rysunek 14: Model bez pokryw



Rysunek 15: Model z pokrywami

### 5.3 Konstrukcja

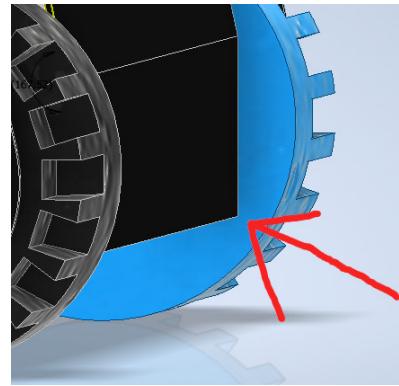
Do wykonania głównych ścianek mocujących oraz uchwytów na silniki użyto druku 3D. Podstawa i plug zostały wykonane z aluminiowej blachy o grubości 2mm. Na obudowę została wykorzystana blacha 0,5mm, którą pomalowano na kolor czarny w celu dużej absorbcji światła wysłanego przez czujniki przeciwnika, co w konsekwencji przekłada się na łatwiejsze uniknięcie wykrycia.

## 5.4 Koła

Felgi zostały wykonane w technologii druku 3D a opony zapożyczone z klocków LEGO. Umieszczono je w taki sposób żeby nawet przy znacznym podniesieniu przodu przez przeciwnika, kontakt kół z podłożem był zachowany.



Rysunek 16: Felgi i opony



Rysunek 17: Zbliżenie na koła

## 5.5 Dociążenie

W ostatecznej wersji robota konieczne było użycie ołowianych obciążników w celu dociążenia robota ponieważ bez tego ważył tylko połowe maksymalnej masy. Sprawiło to że robot nie miał by szans z innymi robotami na zawodach. Przy jego dociążaniu ważne było to aby dobrze rozmieścić mase. Zbyt duża masa z przodu powodowała ślizganie kół. Zbyt duża masa z tyłu podnoszenie się przodu przy gwałtownym cofaniu.

## 5.6 Pierwszy prototyp



Rysunek 18: Widok z przodu



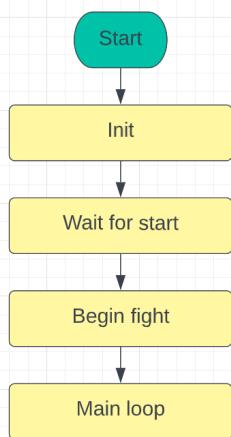
Rysunek 19: Widok z tyłu

## 6 Program

Program robota został napisany w środowisku VSCode przy użyciu wtyczki PlatformIO. Bazą dla całego kodu jest biblioteka "Arduino.h" która zawiera wszystkie podstawowe funkcje.

### 6.1 Szkielet Programu

Szkielet programu oparty jest o następujące bloki:



**Init** - Zawiera definicje używanych funkcji, ustawienia pinów oraz wszystkie podstawowe instrukcje które są wykonywane po włączeniu zasilania, (np. zapalenie diody informującej o włączeniu zasilania czy ustawieniu wachlarzy w góre).

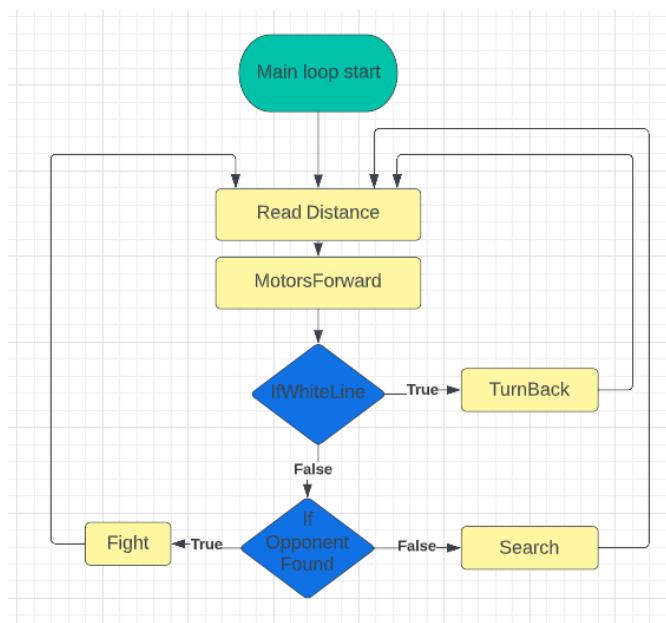
**Wait For Start** - Pętla "while" wykonująca się do czasu otrzymania sygnału ze startera. W trakcie trwania pętli dodatkowo za pomocą przycisku można ustalić kierunek, w którym robot skręci po starcie.

**Begin Fight** - Przygotowanie do walki, wykonywany jest obrót (w lewo lub w prawo, zależnie od wyboru), opuszczanie wachlarzy w dół oraz zaświecenie diody informującej o starcie robota(taka dioda pozwala uniknąć nieporozumień w trakcie zawodów, np. nie odebrania sygnału z pilota).

**Main Loop** - główna pętla programu.

### 6.2 Główna pętla programu - algorytm walki

Algorytm walki robota opiera się głównie o odczyty z czujników odległości, na ich podstawie wykonywane są poszczególne akcje.



**ReadDistance** - Odczytanie wartości z czujników odległości oraz zapamiętanie ostatniej pozycji w której zostało wykryty przeciwnik.

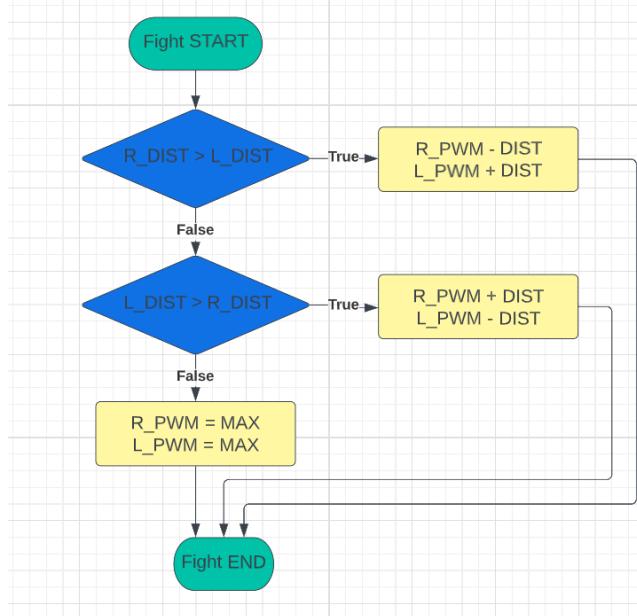
**MotorsForward** - Ustawienie kierunku obrotu silnikami do przodu.

**IfWhiteLine** - Sprawdzenie czy wystąpiła biała linia na podstawie odczytu z czujników podłożu. Jeśli biała linia wystąpi, robot przerywa jazdę do przodu i zwraca w stronę zależną od tego po której stronie czujnik podłożu wykrył białą linię. Jest to zabezpieczenie przed wypadnięciem z ringu.

**IfOpponentFound** - Sprawdza czy w danym odczycie przeciwnik został wykryty.

### 6.3 Fight

Funkcja "Fight" uruchamiana jest w momencie gdy robot wykrył przeciwnika.

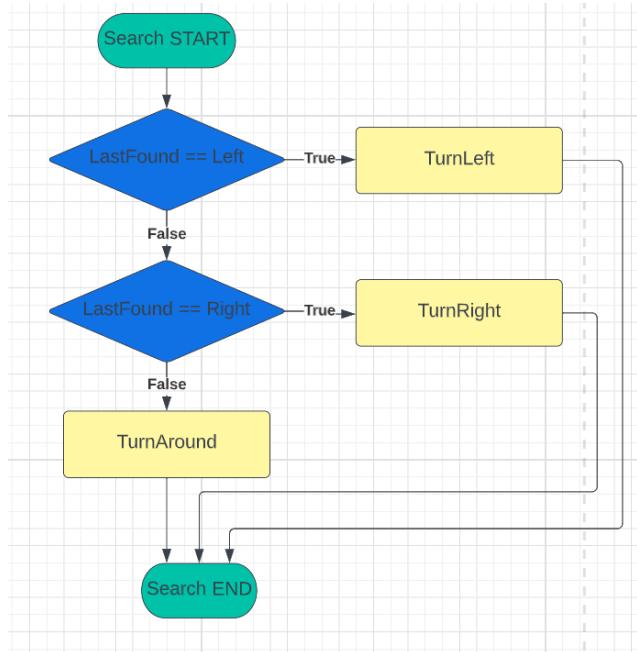


W trakcie jej trwania wykonywane są operacje zmiany szybkości obrotu kół, co pozwala na możliwe najszybsze zepchanie przeciwnika z ringu. W zależności od tego po której stronie czujniki odległości wykryły przeciwnika oraz w jakiej odległości, dobiera odpowiednie wartości PWM. W konsekwencji robot wykonuje skręty o takich kątach które pozwalają na najszybszy kontakt z przeciwnikiem.

Prościej mówiąc gdy przeciwnik jest blisko, robot wykonuje skrót po krótkim łuku, gdy jest daleko, łuk skrętu jest długi. Gdy przeciwnik jest odpowiednio blisko, predkości obrotu obydwu kół ustawiane są na maksymalną wartość.

### 6.4 Search

Funkcja "Search" uruchamiana jest w momencie gdy robot niczego nie wykrył lub przeciwnik uciekł z pola widzenia.



Na podstawie zapamiętanej ostatniej pozycji przeciwnika robot "próbuje" przewidzieć jego obecną pozycję i wykonuje skrót w odpowiednią stronę. W momencie, gdy nie dysponuje ostatnią pozycją, stara się znaleźć przeciwnika poprzez obrót dookoła.

### 6.5 Skręty, obroty i zawroty

Dodatkowo należy wspomnieć, że wszystkie funkcje występujące w programie typu: TurnLeft, TurnRight, TurnBack, TurnAround na bieżąco sprawdzają odczyty z czujników odległości i w razie wykrycia przeciwnika przerywają operacje skrętu.

## 7 Aktualizacje robota

Po stworzeniu wstępnej wersji robota okazało się że niektóre aspekty można znacznie usprawnić.

### 7.1 Nowe koła

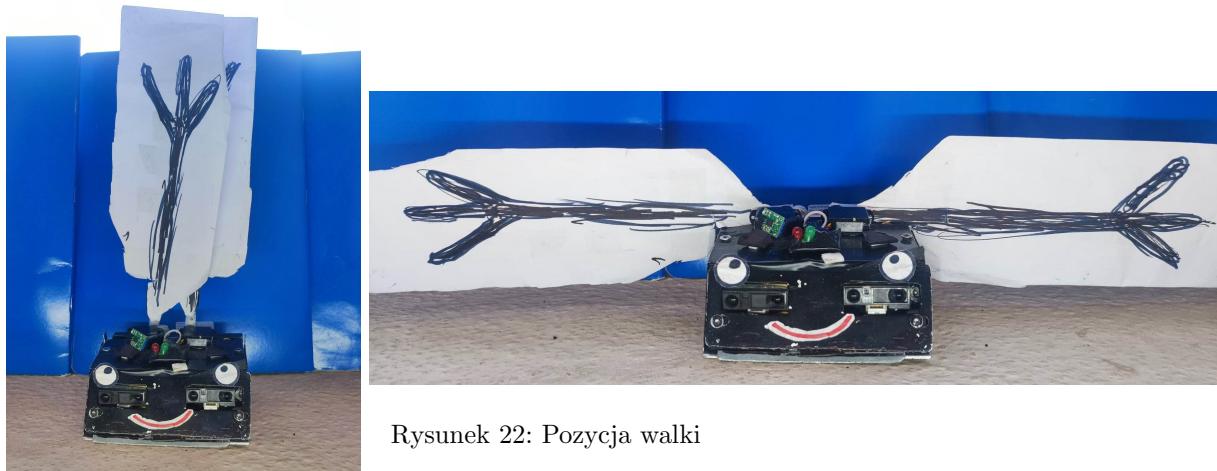
Bardzo dużym problemem był brak odpowiedniej przyczepności do podłoża, opony z LEGO nie były w stanie tego zapewnić. W wyniku tego robot nie miał szans przy bezpośrednich starciach na przepychanie się. W tym przypadku pomogła zmiana kół na opony odlewane z silikonu. Do testów koła zostały pożyczone z robota "XYZ" autorstwa Tomasza Kozłowskiego.



Rysunek 20: Nowe opony

### 7.2 "Technologia wachlarzy" - płachta na byka

Dodatkowe techniki usprawniające taktykę walki zawsze sprawiają że robot ma większą szansę na wygraną. Na zawodach w Bukareszcie okazało się że roboty które były na szczycie posiadały wbudowane, opuszczane po starcie "wachlarze". Po zainspirowaniu się najlepszymi zawodnikami wachlarze zostały zaimplementowane również w Stefanie.



Rysunek 21: Pozycja startowa

Wachlarze w pozycji startowej muszą mieścić się w obrysie robota aby spełniać regulamin. Dopuszcza on zwiększenie rozmiarów robota po starcie, co tutaj bardzo dobrze jest wykorzystywane.

Opuszczane wachlarze sprawiają że nasz robot staje się znacznie trudniejszy do wykrycia ponieważ przeciwnik, który niekoniecznie jest na to przygotowany "myśli" że, w miejscu w którym coś wykrył znajduje się nasz robot. Przeciwnik stara się wtedy z najwyższą prędkością wypchać poza ring naszego robota. Sprawia to że, robot przeciwnika "prześlizguje się" przez wachlarz i sam wypada poza ring. W ostateczności bardzo przeszkadza mu to w wykryciu naszego robota.

Do opuszczania wachlarzy zostały wykorzystane dwa mikro-serwa.

### **7.3 Dodatkowe czujniki odległości**

Jako kolejne ulepszenie robota planowane są dodatkowe czujniki odległości. Umieszczone obecnie dwa czujniki odległości z przodu robota są nie do końca wystarczające. Robot widzi tylko to co ma przed sobą. Umieszczenie dodatkowych czujników z boku, znacznie poprawiło by zdolność do wykrywania przeciwnika, czym samym do wygrywania walk.

### **7.4 Nowa płytka z elektroniką**

Jako kolejne ulepszenie chciałbym wykonać projekt płytki PCB, aby elektronika robota była bardziej niezawodna.

## **8 Wady konstrukcyjne**

- Bateria zastosowana w robocie ma niepotrzebnie tak dużą objętość. Można zastosować mniejszą co sprawiło by że robot mógłby być niższy.
- Niezaprzeczalnie jedną z największych wad jest metoda wykonania płytki z elektroniką. Użyty sposób wykonania płytki ma znacznie więcej wad niż zalet oraz nie jest on zbytnio profesjonalny i wytrzymały. Połączenia kabli często ulegały awarii, co znacznie przeszkadzało w pracy przy projekcie.
- Brak noża dzięki któremu łatwiej jest podważyć przeciwnika.
- Czujniki odległości umieszczone są za wysoko co sprawia że czasami przeciwnik może zostać niezauważony przez swoją wysokość.
- Program robota wymaga sporej optymalizacji ponieważ działa powoli i często robot nie reaguje odpowiednio szybko na wykrycie przeciwnika. Bywają też momenty że przeciwnik w ogóle nie zostanie wykryty.

## 9 Zawody

### 9.1 RoboChallange Bukareszt 5-6.11.2022

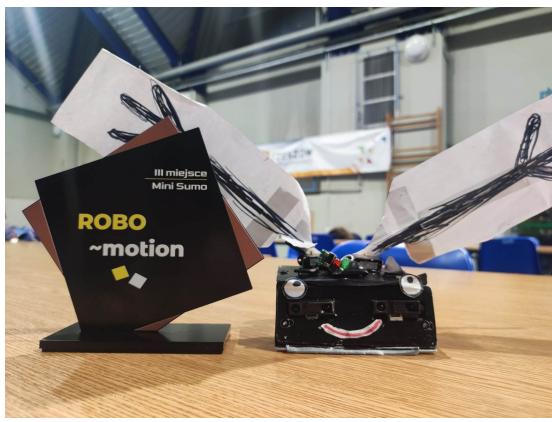
Zawody w Bukareszcie były pierwszymi zawodami na jakich Stefan wystartował. Poradził on sobie niespodziewanie dobrze. Wygrał on jedną walkę w grupie, co wystarczyło na przejście do kolejnego etapu - finałów. Dzięki tym zawodom można było przekonać się co w robocie wymaga poprawy i ulepszeń.



Rysunek 23: Stefan z Paskudą i MiniRikishi w Bukareszcie.

### 9.2 Robo Motion Rzeszów 26.11.2022

Były to pierwsze zawody Stefana na których był już wyposażony w nowe koła i wachlarze. Te dwa ulepszenia okazały się znaczące. Po pięknych, dwóch wygranych walkach w grupie, wyszedł z niej na drugim miejscu- przegrywając jedynie z robotem "Er-sześć". Pokonując kolejnych przeciwników ostatecznie zajął 3 miejsce!



Rysunek 24: Stefan po zawodach w Rzeszowie



Rysunek 25: Podium w Rzeszowie

## Literatura

- Eryk Moźdzeń - Robot mobilny klasy minisumo „Sneak100”: <https://github.com/Eryk-Mozdzen/minisumo-sneak100>
- SumoStartModule: <https://p1r.se/startmodule/>, <https://github.com/p1rse/robot-sumo-start-module>