Dokumentacja

Robot mobilny klasy minisumo „Scalak”

Eryk Możdżeń

18 Września 2021

Spis treści

1 Wstęp

W niniejszym dokumencie zostały zawarte informacje na temat budowy oraz działania robota „Scalak". Pomysł na tę konstrukcję narodził się podczas międzynarodowych zawodów „XI Robotic Arena" w styczniu 2019 roku we Wrocławiu, na których pierwszy raz zetknąłem się z robotami kategorii minisumo. Celem projektu było rozwinięcie swoich umiejętności w dziedzinie elektroniki, programowania miktrokontrolerów oraz sterowania robotami mobilnymi.

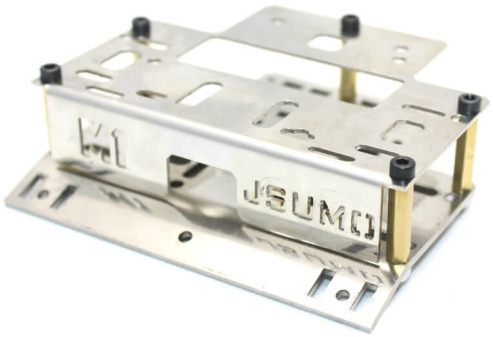
2 Założenia projektowe

- Konstrukcja w pełni autonomiczna
- Użycie płytki rozwojowej Arduino Nano oraz zintegrowanego środowiska Arduino IDE
- Wykorzystanie enkoderów inkrementalnych
- Kompatybilność z standardem modułów startowych
- Zastosowanie 3 punktów podparcia (2 koła + pług)
- Konstrukcja o obrysie mieszczącym się w polu 100x100mm
- Waga nieprzekraczająca 500g

3 Mechanika

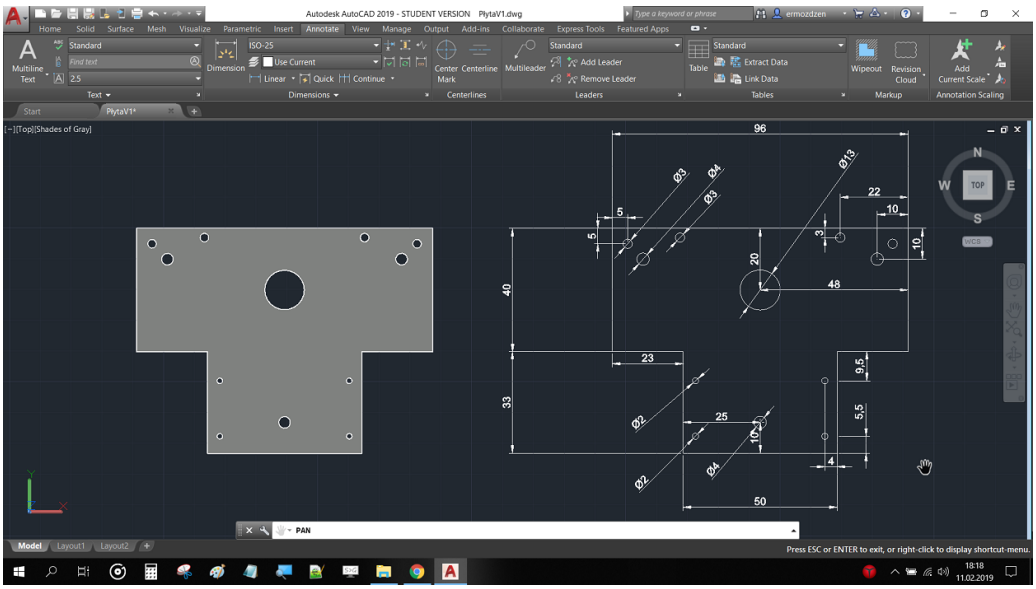
3.1 Rama

Główny założeniem całego projektu miała być prostota oraz nie wielka ilość części potrzebny do powstania konstrukcji. Początkowo dużą inspiracją dla projektu była rama JSumo. Do końcowego projektu trafiła koncepcja pojedynczej płyty głównej z otworami montażowymi, które umożliwiają przytwierdzenie reszty elementów. Usunięcie górnej blaszki ochronnej było spowodowane chęcią uproszczenia konstrukcji oraz obniżenia wysokości robota.



Rysunek 1: Rama ze sklepu JSumo, inspiracja dla konstrukcji,
dwie wyprofilowane płyty z otworami montażowymi, blaszka osłaniająca wnętrze,
mocowanie pługu, dystanse

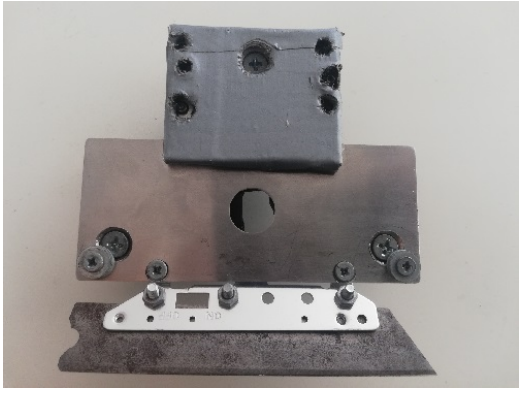
Projekt został wykonany w programie AutoCAD 2019, a sama płyta została wycięta frezarką sterowaną numerycznie z płyty aluminiowej o grubości 2mm. Otwory zostały ręcznie sfazowane tak, aby trapezowe łepki śrub mogły się zmieścić w jej obrys.



Rysunek 2: Zrzut ekranu z projektu płyty głównej w programie AutoCAD 2019

(a) widok z przodu: pług, śruby dystansowe, obciążnik ołowiany wewnętrzny, mocowanie silników

(b) widok z tyłu: śruby dystansowe, obciążnik ołowiany wewnętrzny, mocowanie silników, śruba utrzymująca silniki w miejscu



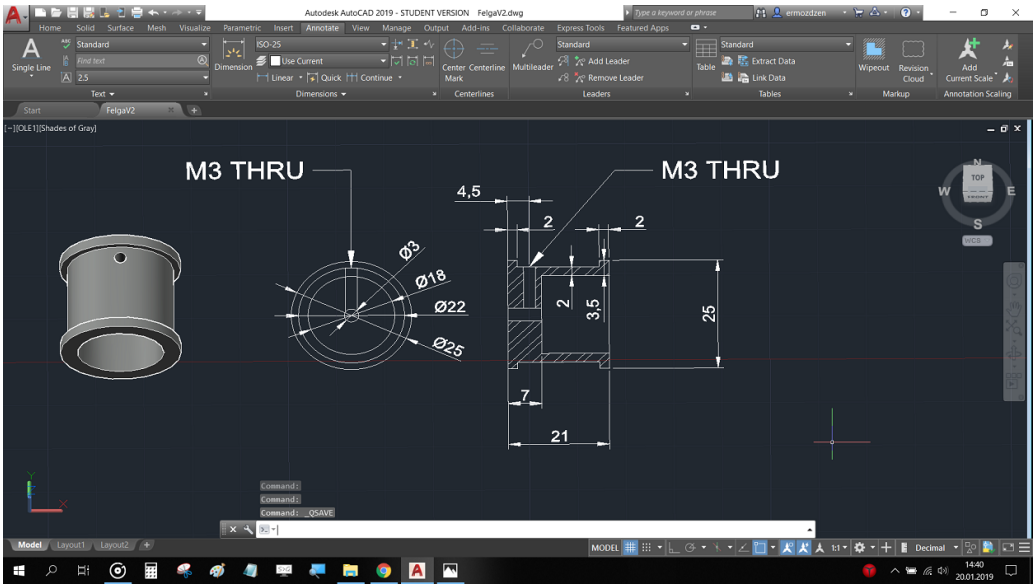
(c) widok z dołu: mocowanie pługu, śruby czujników linii, obciążnik ołowiany zewnętrzny, centralny otwór na przewody

Rysunek 3: Rama robota wraz z resztą elementów montażowych

3.2 Koła

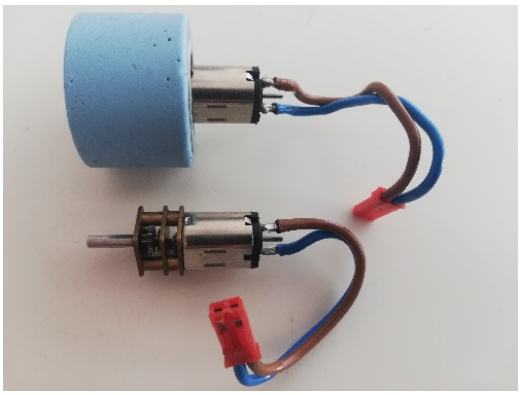
Felgi

Felgi zostały wytoczone z aluminiowego walca według wymiarów przedstawionych na rysunku poniżej. Zostały zaprojektowane tak, aby silniki napędowe „wpościć" w głąb koła, dla zaoszczędzenia miejsca. Dwa pierścienie na obwodzie chronią oponę przed obsunięciem się. Całość jest przyciskana do osi, poprzez śrubę mocującą wkręcaną w nagwintowany otwór wewnątrz felgi.



Rysunek 4: Zrzut ekranu z projektu felgi w programie AutoCAD 2019

(a) felgi,
widoczny otwór na śrubę mocującą

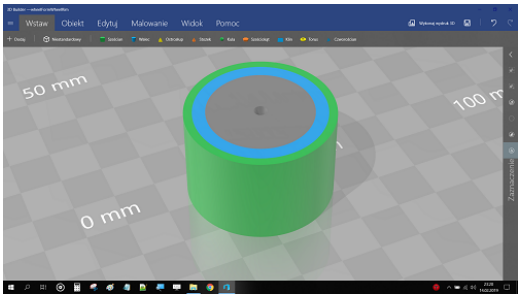


(b) koło zamontowane na osi silnika,
widoczne „wpuszczenie" silnika wewnątrz koła

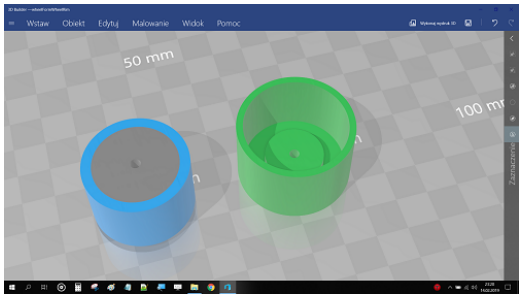
Rysunek 5: Docelowe felgi do kół napędowych robota

Opony

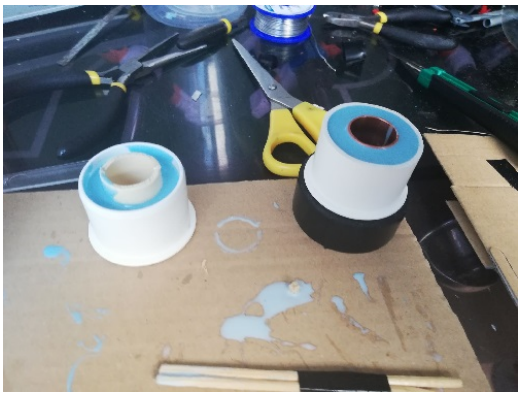
Opony zostały odlane z sylikonu formierskiego MM922 z niebieskim katalizatorem o twardości 22Sha. Proces odlewania odbywał się w wytoczonej z aluminium formie, w której wcześniej została przykręcona felga (z zaklejonym otworem na śrubę mocującą). Poglądowe modele zostały stworzone w programie 3D Builder.



(a) poglądowy model opony w formie,
felga – szary, opona – niebieski, forma – zielony



(b) poglądowy model opony poza formą,
felga – szary, opona – niebieski, forma – zielony



(c) testowe formy w trakcie twardnienia

(d) od lewej: forma, felga z oponą, opona

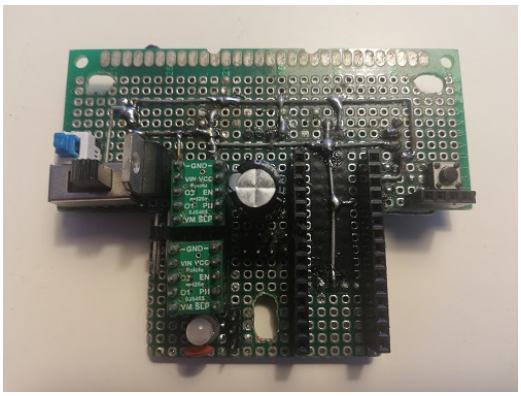
Rysunek 6: Proces odlewania opon

4 Elektronika

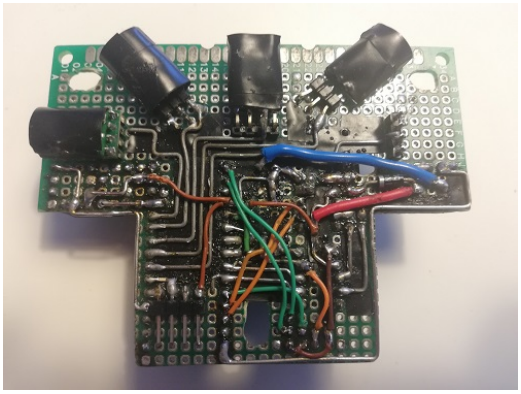
4.1 Płytka główna

Płytka z uwagi na brak wcześniejszych doświadczeń z projektowaniem płytek PCB została wykonana z płytki drukowanej uniwersalnej, polutowanej za pomocą drutów ze spinaczy oraz drobnych kabli z odzysku.

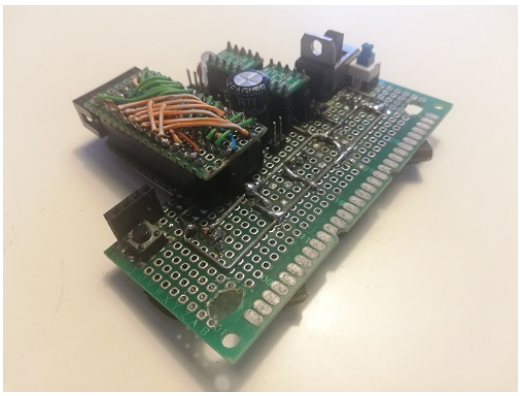
W płytkę na stałe wlutowanych jest 5 czujników wykrycia przeciwnika (produkt Pololu 2578) owiniętych taśmą izolacyjna, aby nie wykrywały ramy robota. Każdy z nich skierowany jest w inną stronę (od lewej: $-90°$, $-45°$, $0°$, $45°$, $90°$).



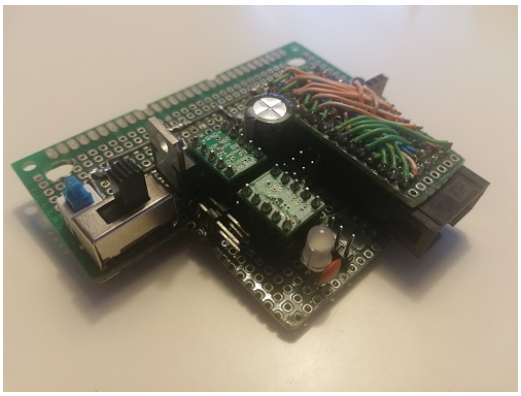
(a) widok z góry, od lewej:
przełacznik dwupozycyjny, włącznik, stabilizator napięcia, konektory silników, sterowniki silników, dioda RGB, konektor UART, konektor baterii, konektor enkoderów, gniazdo mikrokontrolera, przycisk, konektor modułu startowego



(b) widok z dołu,
czujniki obecności przeciwnika,
konektor płytki odbiciowych czujników linii



(c) widok z przodu,
pusta przestrzeń pozostawiona na baterię,
przejściówka z mikrokontrolerem

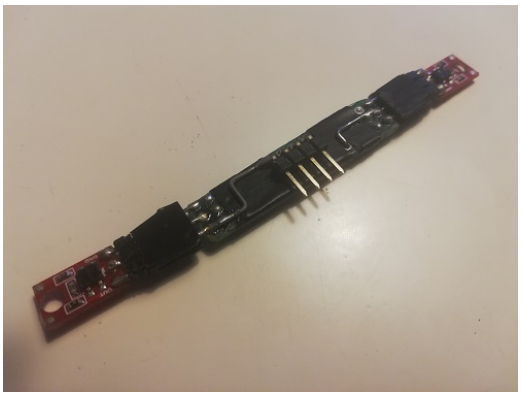


(d) widok z tyłu,
przejściówka z mikrokontrolerem

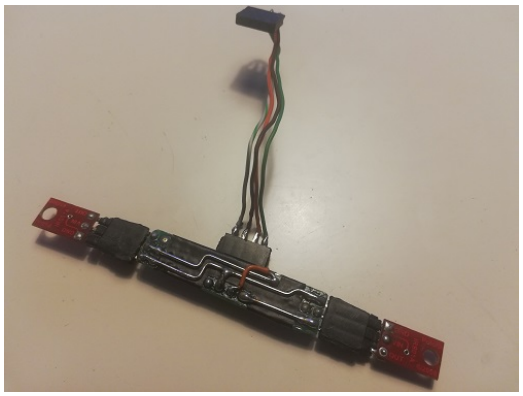
Rysunek 7: Autorska płytka do robota

4.2 Płytka odbiciowych czujników linii

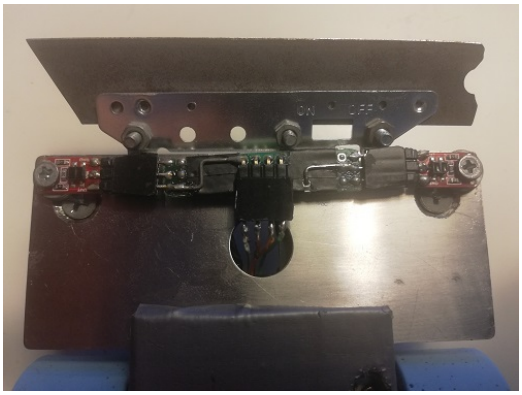
Aby zagregować sygnały oraz linie zasilające dwóch modułów QTR-1A (zawierające w sobie popularne KTIR0711S) powstałą płytka pomocnicza. Jest przykręcana za pomocą dwóch śrub do ramy od spodu. Odległość czujników od podłoża wynosi około 3mm.



(a) widok z dołu,
dwa czujniki w gniazdach



(b) widok z góry,
przewód sygnałowy



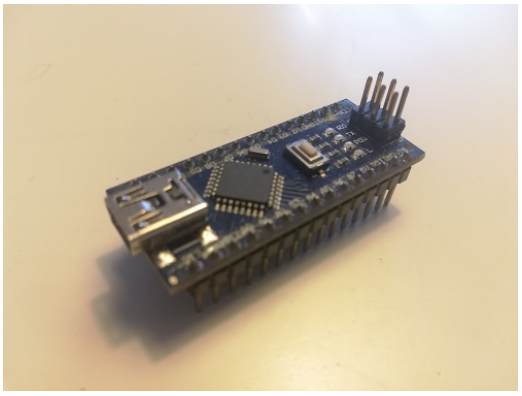
(c) widok z dołu,
płytka wmontowana w ramę robota

Rysunek 8: Płytka odbiciowych czujników linii

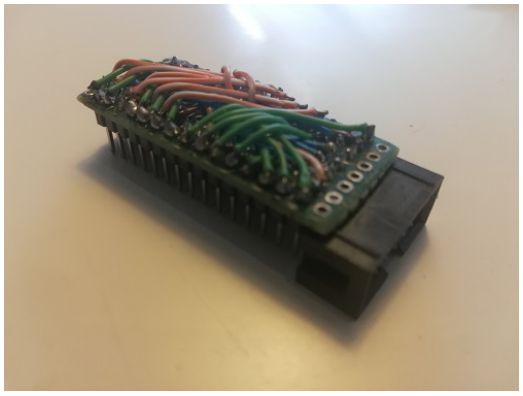
4.3 Mikrokontroler

Początkowo, projekt opierała się na powszechnie znanej płytce Arduino Nano (Rys. 9a), zawierającej mikrokontroler ATmega328p, wybranej z powodu dostępności materiałów do nauki, łatwości implementacji, dedykowanego środowiska oraz z uwagi na wcześniejsze doświadczenia z tym systemem.

W pewnym momencie biblioteki Arduino oraz wszyskie "ułatwienia" jakie miały za tym iść stały się ograniczajace, więc po stworzeniu odpowiedniej przejściówki (Rys. 9b) w robocie został zainstalowany zwykły chip ATmega328p w obudowie DIP28.



(a) płytka Arduino Nano,
bootloader, gniazdo USB, diody sygnalizacyjne



(b) surowy chip w przejściówce,
wyprowadzenia kompatybilne z Arduino Nano,
wyjście IDE 10-pin programatora USBasp

Rysunek 9: Zastosowane mikrokontrolery ATmega328p

5 Program

Program początkowo w języku C++, tworzony był w zintegrowanym środowisko Arduino IDE, a pliki biblioteki edytowane za pomocą edytora VS Code. Po przejściu na "surową" ATmegę, projekt został przepisany na język C w środowisku dedykowanym mikrokontrolerom z rodziny AVR - AtmelStudio.

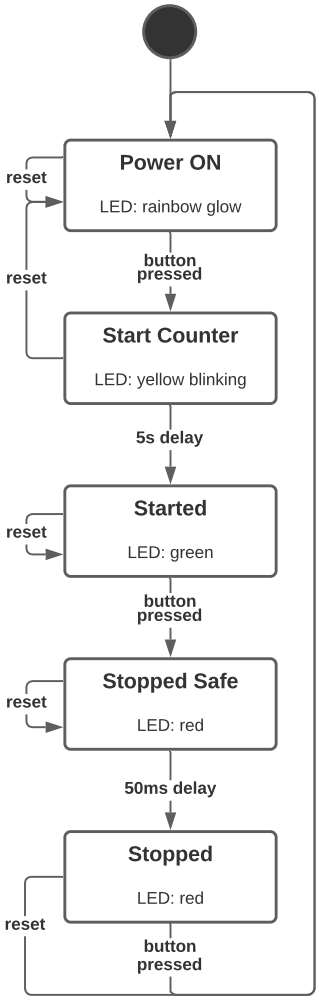
5.1 Budowa programu

Program został oparty o pojęciu "maszyny stanów", której przejścia pomiędzy stanami są zależne od informacji docierającej do robota. Główne wyodrębnione stany:

- Power ON – inicjalizacja peryferiów, oczekiwanie na sygnał
- Started – konstrukcja rozpoczęła algorytm walki sterowany odrębną maszyną stanów
- Stopped Safe – stan zabezpieczający przed odbezpieczeniem zasilania robota w warunkach nagłego zatrzymania silników np.: szok elektromagnetyczny, zakłócenia zasilania (dotyczy głównie większych konstrukcji)
- Stopped – stan unieruchomienia robota

Kontrola przyciskiem

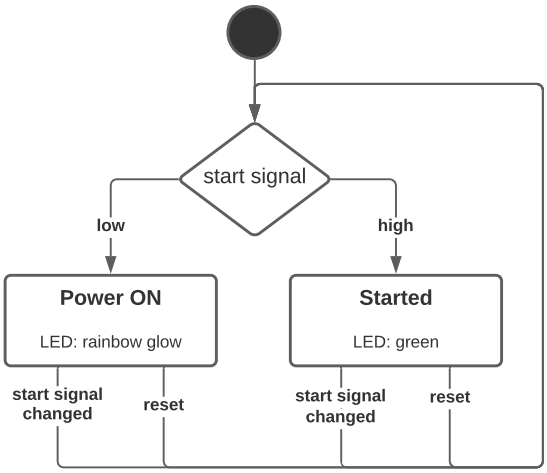
Jest to najprostsze rozwiązanie, dawniej standard podczas zawodów. Po ustawieniu robotów na planszy, zawodnicy oczekują sygnału sędziego do startu i gdy to nastąpi, naciskają wcześniej przygotowane przyciski będące częścią konstrukcji. Po 5 sekundach, potrzebny do bezpiecznego oddalenia się zawodników i sędziego (stan "Start Counter"), roboty zaczynają walkę. Forma wyłączania nie jest określona, lecz w tym przypadku dla wygody odbywa się to poprzez ponowne naciśnięcie przycisku.



Rysunek 10: Schemat UML maszyny stanów podczas kontroli konstrukcji przyciskiem

Kontrola modułem startowym

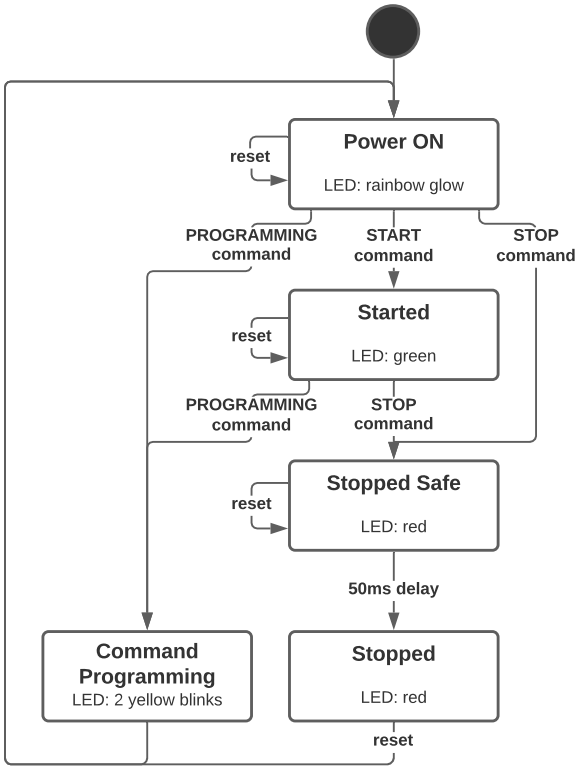
Moduły startowe stały się standardem wymaganym na większości zawodów, w celu zwiększenia bezpieczeństwa zawodników i konstrukcji, przyśpieszenia przeprowadzania walk oraz wyeliminowania falstartów. Oficjalna strona przedstawiająca standard modułów jest pod adresem: https://p1r.se/startmodule/.



Rysunek 11: Schemat UML maszyny stanów podczas kontroli konstrukcji modułem startowym

Emulator modułu startowego

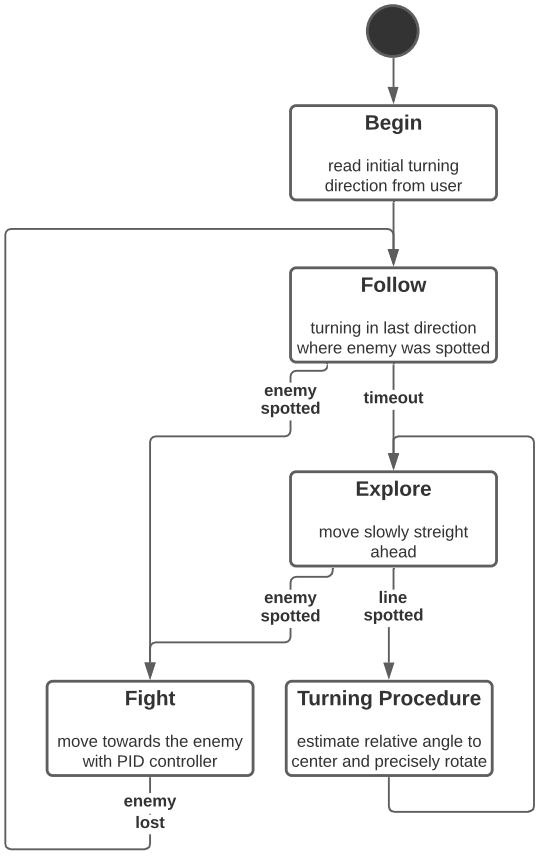
Z powodu zainteresowania tematem oraz początkowym brakiem dostępu do wyżej wymienionych modułów, w miejsce wolnych wyprowadzeń enkodera inkrementalnego został wpięty prosty odbiornik podczerwień TSOP31238 wraz z odpowiednimi komponentami (rezystor i kondensator). Potrafi on wykryć obecność sygnału 38kHz, używanego w pilotach startowych. Paczki danych są zakodowane sygnałem RC5, który mikrokontroler odczytuje, co pozwala na rozszerzenie funkcji robota do bycia swoim własnym modułem startowym.



Rysunek 12: Schemat UML maszyny stanów podczas kontroli konstrukcji odbiornikiem RC5

5.2 Algorytm walki

Algorytm walki w większości polega na znalezieniu przeciwnika (szczególnie w kategoriach minisumo enchanted, gdzie średnia dyhlo osiąga 1.54m). Po nawiązaniu pierwszego kontaktu, stery przejmuję kontroler PID, który dąży do utrzymania kursu robota na wprost przeciwnika (dążenie do odchyleń powoduje ciekawe zachowanie robota, który próbuje "orbitować" wokół przeciwnika).



Rysunek 13: Schemat UML maszyny stanów podczas kontroli konstrukcji przyciskiem

Estymacja pozycji przeciwnika

Konstrukcja zawiera 5 czujników wykrywający obecność przeciwnika w zakresie $\pm 90°$. Znajomość tego które czujniki zgłaszają wykrycie obiektu i założenia, że na ringu znajduje się tylko jeden przeciwnik oraz dostępna przestrzeń ogranicza się do płaszczyzny ringu, pozwala na określenie przybliżonego kąta obecności przeciwnika względem kierunku frontu robota.

Odbywa się to poprzez nadanie wartości katów poszczególnym czujnikom (w tym przypadku są to od lewej: $-90°$, $-45°$, $0°$, $45°$, $90°$), a następnie wykonanie średniej arytmetycznej z wszystkich wartości, których czujniki stwierdziły wykrycie obiektu.

Estymacja pozycji względem krawędzi dyhlo

Przy założeniu, że krawędź ringu jest linią prostą (dla dużych promieni ringu generuje to mały błąd), nie występują poślizgi kół oraz że sytuacja nie jest "ekstremalna" (kąt jest bliski $\pm 90°$), możliwe jest przybliżenie kąta pod jakim robot najeżdża na krawędź ringu i o jaki kąt powinien skręcić, aby powrócić na środek planszy (kierunek prostopadły do stycznej krawędzi w punkcie położenia robota).

Po wykryci linii na którymś z czujników odbiciowych, robot zwalnia tempo poruszania i dostępnymi metodami (impulsy enkoderów, czas) mierzy odległość którą przebywa do momentu wykrycia linii czujnikiem po drugiej stronie kadłuba.

Znając rozstaw czujników $D_{cz} = const$ i przejechany dystans d kąt najechania na krawędź θ_n i kąt wymaganego obrotu θ_o wynoszą odpowiednio:

$$\theta_n = \arctan\left(\frac{d}{D_{cz}}\right) \qquad \theta_o = \pi - \theta_n$$

6 Napotkane problemy

6.1 Waga robota

Po pierwszym złożeniu wszystkich elementów okazało się, że robot uzyskiwał niewielki docisk kół do podłoża, co znacznie odbijało się na możliwościach konstrukcji podatności na ataki przeciwników.

Obciążnik ołowiany wewnętrzny

Jako, że pomiędzy czujnikami wewnątrz robota było dużo wolnej przestrzeni została wypełniona ołowianymi kulkami używanymi w wędkarstwie owiniętych w czarną taśmę izolacyjną. Cały pakunek jest przyklejony do płyty za pomocą taśmy dwustronnej i waży 94g. Element ten jest ściśle uwarunkowany dostępnym miejscem, co powoduje jego charakterystyczny kształt.

Obciążnik ołowiany zewnętrzny

Po dodaniu wewnętrznego dociążenia waga robota jeszcze nie osiągnęła maksymalnej wartości (500g), więc po otrzymaniu kilku porad, wypełniona także została szczelina pomiędzy płytą a podłożem pod silnikami. W tym celu zostały wykorzystane dwie warstwy 2 milimetrowych płytek ołowianych, o masie 81g. Ten element również został owiniętych srebrną taśmą (miejsce to pozwala na wygodne uchwycenie robota, a a skóra dłoni źle reagowała z odsłoniętym ołowiem).

(a) obciążnik wewnętrzny (94g), wędkarskie kulki ołowiane

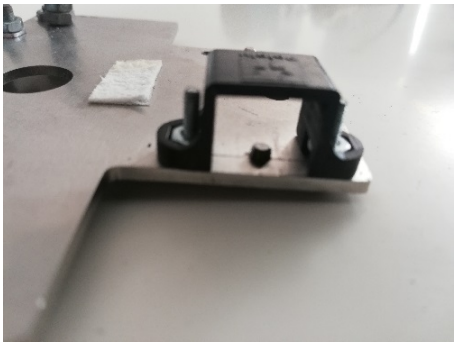
(b) obciążnik zewnętrzny (81g), ołowiana płytka 2mm

Rysunek 14: Dodatkowe obciążniki robota

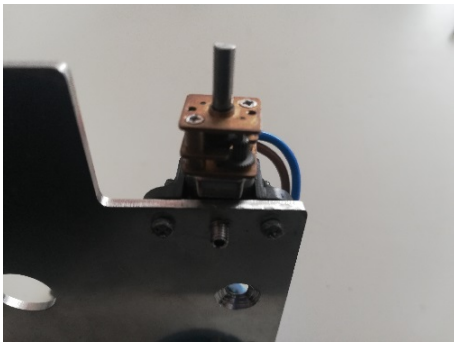
6.2 Mocowanie silników

Mocowania silników napędowych zostały zmodyfikowane tak, aby obejmować sam silnik i umożliwić wpuszczenie przekładni w obrys felg, jednak podczas wykonywania gwałtownych manewrów, silniki miały tendencję do wysuwania się.

Rozwiązaniem stało się dodanie dwóch śrub mocujących (po jeden na każdy silnik), wkręcanych w płytę tak, aby punktowo dociskały silniki do mocowania od dołu, zapobiegając ich przemieszczanie się.



(a) widok od strony silnika



(b) widok od dołu

Rysunek 15: Śruba mocująca silniki napędowe

6.3 Zasilanie

Podczas nagłych zderzeń występowały tzw. „szpilki" napięcia, które były niebezpieczne dla komponentów elektronicznych. Objawiało się to resetowaniem mikrokontrolera robota (co na początku było bardzo kosztowne czasowo z uwagi na wykorzystanie bootloader-a Arduino).

Filtracja zasilania

Problem został rozwiązany, poprzez podłączenie kondensatora elektrolitycznego o dużej pojemności (w tym przypadku 780uF) bezpośrednio pomiędzy linię zasilania siłowego pochodzącego z baterii oraz masę układu.

Zapis stanu w pamięci EEPROM

Restart pracy układu powodował także tragiczne skutki dla pracy maszyny stanów w programie (restart mikrokontrolera rozpoczynał program od nowa). Aby dodatkowo zabezpieczyć software przed resetem, za każdym razem gdy algorytm przechodzi w nowy stan, odpowiednia wartość zapisywana jest w wewnętrznej pamięci trwałej mikrokontrolera EEPROM. Po włączeniu zasilania program próbuje wrócić do stanu przed wyłączeniem, bądź do stanu, który powinien nastąpić po manualnym restarcie (względy bezpieczeństwa).

6.4 Enkodery

Enkodery inkrementalne, mimo że był w zamyśle projektowym od samego początku, finalnie zostały dodane na zbyt późnym etapie. Tarcze magnetyczne umieszczone na osi silników były umieszczone zbyt blisko gęstej sieci przewodów, która skumulowała się w przestrzeni pomiędzy silnikami napędowymi. Problemem była awaryjność konstrukcji, ponieważ często jeden z przewodów lekko zmieniał swoje położenie, co prowadziło do znacznych oporów na osi silnika. Wynikiem czego silnik nie był w stanie poruszać robotem.

Mimo usilnych prób, nie zostało znalezione rozwiązanie tego problemu, które nie wymagało by przebudowy większości konstrukcji, więc aby uchronić się przed kosztowną czasowo awarią np. podczas zawodów, enkodery zostały wymontowane całkowicie, kosztem możliwości sterowania silnikami w zamkniętej pętli oraz obliczania przejechanego dystansu.

6.5 Pług

Mocowanie

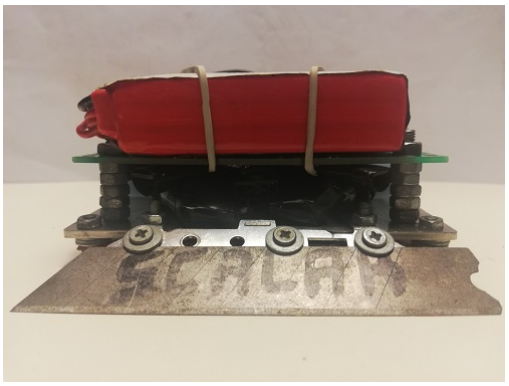
Do zamocowania pługu została użyta odpowiednio wyprofilowana blaszka, pozyskana ze starej zabawki. Utrzymuje ona pług, czyli w tym przypadku ostrzę noża do tapet za pomocą kilku śrub z podkładkami. Jako, że prawie połowa ciężaru robota spoczywała na nożu, na blaszce oddziaływały duże naprężenia, co doprowadziło raz do jej złamania w miejscach zgięć.

Przy mocniejszych zderzeniach z przeciwnikiem, nóż wypadał spod podkładek, którymi był dociskany do mocowania, co prowadziło do odrzucenia pługu poza strefę wali.

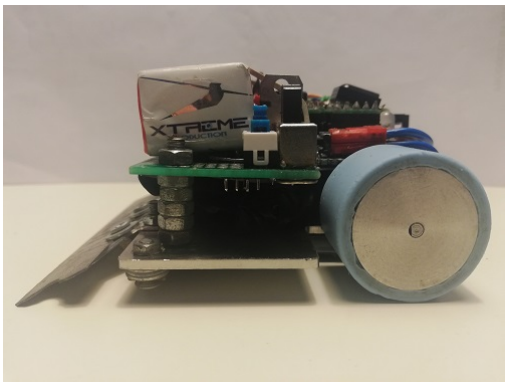
Nóż

Zastosowanie noża do tapet w roli pługa przyniosło ze sobą zaskakująco dobre efekty (elastyczność, dopasowanie do podłoża, dobre własności ofensywne i defensywne), jednak było to osiągnięte kosztem ryzyka zarysowania planszy podczas poruszania się, a także zagrożeniem dla użytkownika w przy niezachowaniu odpowiedniej uwagi.

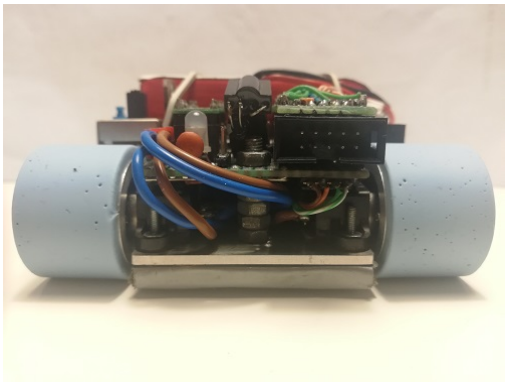
7 Zdjęcia



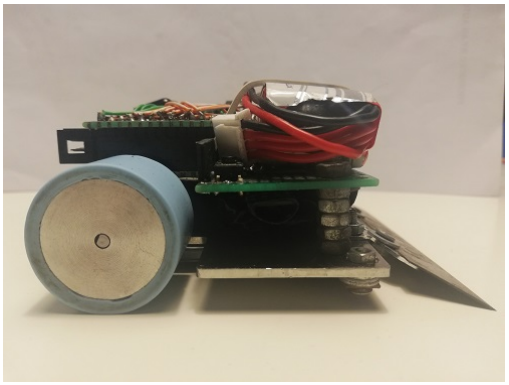
(a) widok z przodu



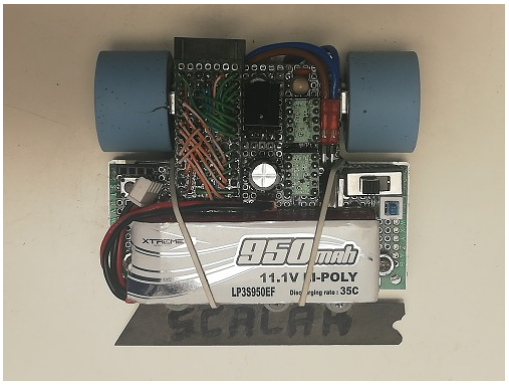
(b) widok z lewej



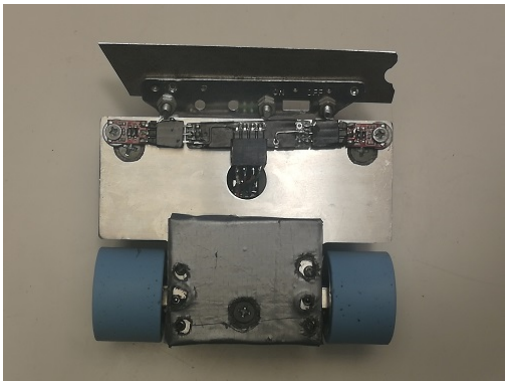
(c) widok od tyłu



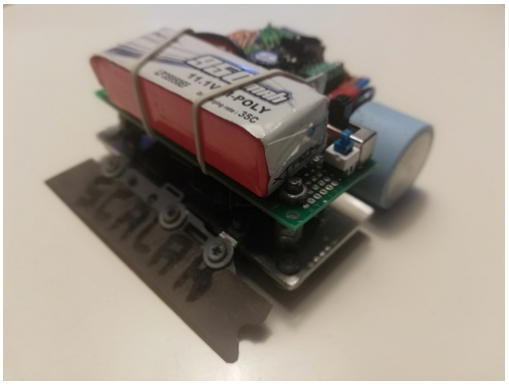
(d) widok z prawej



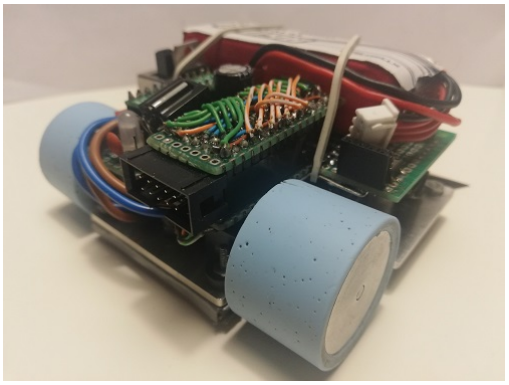
(e) widok z góry



(f) widok od dołu



(g) widok pod skosem z przodu



(h) widok pod skosem od tyłu

Rysunek 16: Konstrukcja złożonego robota

8 Podsumowanie

Aspekt mechaniki został wykonany w całości według pomysłu, który narodził się na samym początku, rozbudowany o elementy poznane podczas pracy nad projektem. Pozostawia wiele miejsc do poprawy, lecz spełnia swoją rolę jako prostej, dwukołowej platformy o skutecznych własnościach defensywnych w omawianym kontekście.

Wielokrotnie pojawiały się plany na przejście na projekt płytki PCB wykonanej na zamówienie, lecz nie można ukryć że zatrzymałoby to rozwój tej konstrukcji. Toporność wykonania i dostępność materiałów sprawiła, że mimo posiadania prostej lutownicy, piły i kombinerek w ciągu jednego dnia możliwe było przerobienie kluczowych elementów.

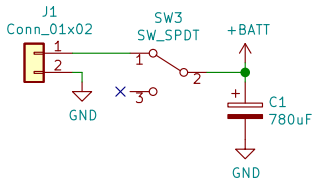
Niniejszy dokument oraz repozytorium, jako kompleksowy zbiór materiałów został wykonany zdecydowanie za późno (prawie 2 lata po fakcie w celu archiwizacji), co skutecznie utrudniało powracanie do projektu po dłuższym czasie pomiędzy poszczególnymi eventami.

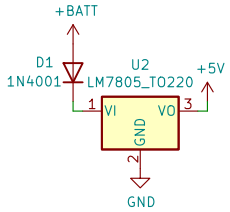
Cały projekt podczas swojego powstawania poruszył znacznie więcej tematów i trwał o wiele dłużej niż początkowo zakładał. Zdecydowanie spełnił swoje zadanie jako platforma do nauki i rozwijania szerokiego wachlarzu umiejętności. Oficjalnie pracę nad nim zostały zakończone w grudniu 2019, lecz pojawia się na zawodach do dzisiaj, niestety bez większych osiągnięć.

9 Odnośnik do repozytorium GIT

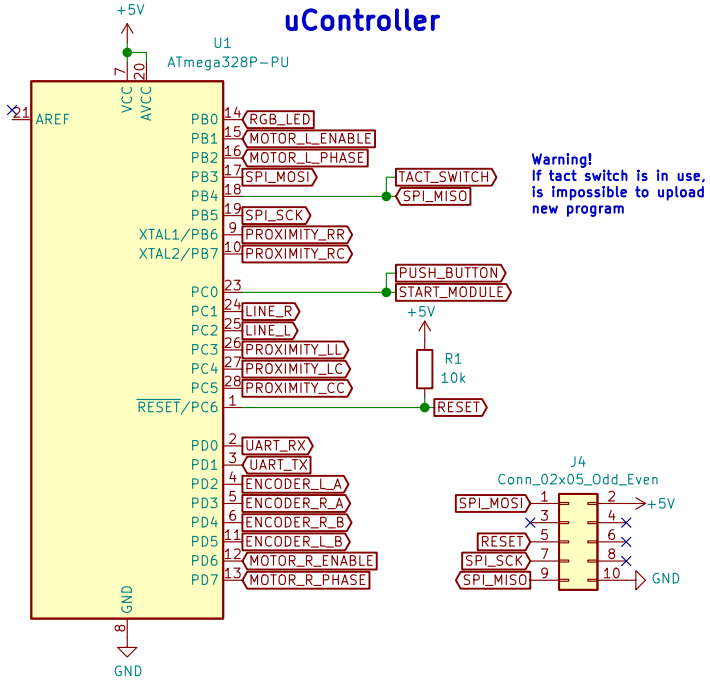
Wszystkie materiały, projekty, pliki, zdjęcia, notatki i dokumenty, które zostały zachowane z tego projektu zostały umieszczone na repozytorium GIT: https://github.com/Eryk-Mozdzen/minisumo-scalak.

Power management

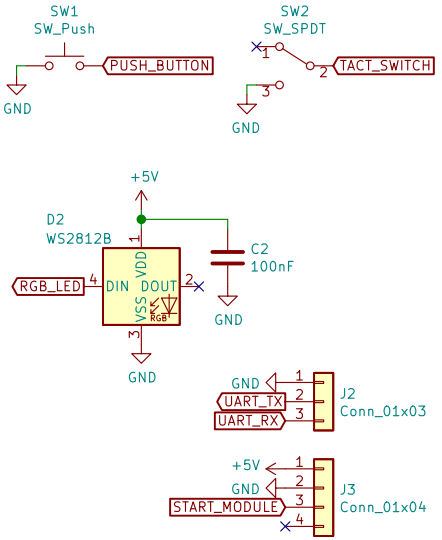




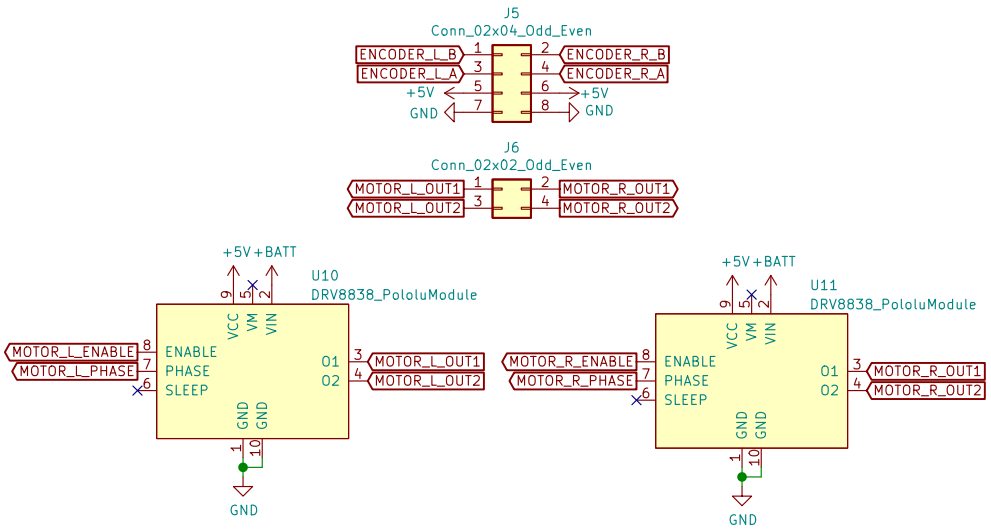
uController



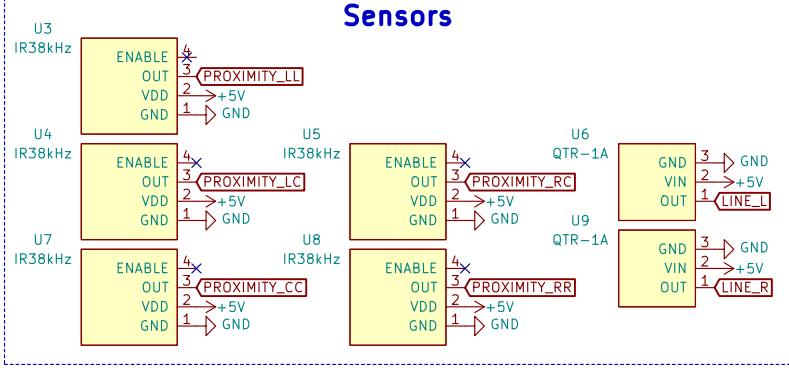
Interface



Motors & Encoders



Sensors



Sheet: /
File: scalak_main_board.sch

Title:

Size: A4	Date:	**Rev:**
KiCad E.D.A. kicad (5.1.6)-1		Id: 1/1