AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

Projekt dyplomowy

Skanowanie powierzchni wykrywaczem metalu przy pomocy pojazdu autonomicznego

Area scanning with metal detector mounted on autonomous vehicle

Autor: *Jakub Bronicki*
Kierunek studiów: *elektrotechnika*
Opiekun pracy: *dr inż. Grzegorz Hayduk*

Kraków, 2020

Spis treści

1. Wstęp

Wraz z postępem technologicznym autonomiczne pojazdy czy roboty stały się pewnym nieodłącznym elementem naszej rzeczywistości. Ludzkość w coraz większym stopniu zmierza w kierunku pełnego zautomatyzowania pewnych czynności, które, zwłaszcza jeśli są monotonne, nie muszą być przez nich wykonywane. Korzystając z najnowszych osiągnięć jesteśmy w stanie zastąpić pracę kilku osób, które muszą zostać odpowiednio przygotowane i przeszkolone do pracy. Z pomocą przychodzą nam w pełni autonomiczne roboty, które po zaprogramowaniu, wymagają jedynie znikomej reakcji lub kontroli ze strony operatora. Zasilane energią elektryczną, nie potrzebują przerwy w pracy, wykonują swoją pracę powtarzalnie oraz z dużo lepszą skutecznością.

1.1 Cel i zakres pracy

Celem mojej pracy było wykonanie projektu i prototypu pojazdu wyposażonego w detektor metalu, zdolnego skanować obszar o zadanej powierzchni i budującego na tej podstawie mapę siły sygnału z detektora. Następnie mapa ma zostać przekazana do komputera celem graficznego zwizualizowania wyników skanowania. Pojazd ma posiadać napęd pozwalający na poruszanie się po nierównym terenie i zapewnić przeskanowanie obszaru o definiowalnych maksymalnych wymiarach <szerokość>x<długość>, omijając wykryte przeszkody. Aby spełnić podane założenia należało skupić się na rozwiązaniu kilku problemów jakimi były: dobór odpowiedniej konstrukcji pojazdu, dobór metody wykrywania elementów metalowych oraz wybranie sposobu dwustronnej komunikacji, która pozwala na przesłanie wymiarów obszaru i prezentację wyników.

1.2 Układ pracy

Praca została rozdzielona na część teoretyczna i projektową.

W części teoretycznej, w kolejnych podrozdziałach, znajdują się opisy oraz schematy wszystkich użytych w prototypie podzespołów, a także kryteria doboru poszczególnych elementów i dopasowanie ich do założeń projektowych. Następnie, w części projektowej, znajdują się opisy budowy prototypu oraz sposób wykonania ewentualnych modyfikacji podzespołów, które wynikały z kolejnych przeprowadzanych testów. Ostatnim rozdziałem pracy są wnioski, w których zawarte jest podsumowanie wykonanych doświadczeń, odniesienie się do założeń projektowych, sposoby na usprawnienie prototypu oraz dalsze możliwości rozwoju.

2. Część teoretyczna

2.1 Platforma mobilna

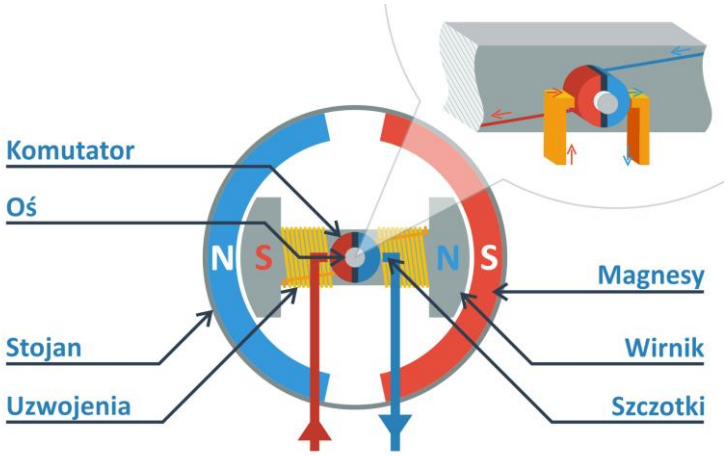
Rys 2.1 Platforma robota z napędem. [1]

Platformy mobilne do budowy robotów można podzielić na dwa podstawowe rodzaje. Konstrukcja pierwszego z nich jest oparta na zastosowaniu napędu gąsienicowego. Pojazdy tego typu cechują się bardzo dobrą przyczepnością do podłoża oraz możliwością poruszania się w trudnych warunkach terenowych. Jego wadą jest wysoka cena, w porównaniu do drugiego rodzaju platform, które wykorzystują napęd kołowy. Napęd ten charakteryzuje się mniejszą skutecznością, podczas poruszania się w trudnym terenie na przykład po błocie. Platformy z napędem kołowym można dzielą się na kolejny dwa rodzaje, w zależności od ilości kół napędowych. W przypadku wykorzystania czterech niezależnych silników, wzrasta trudność wykonania prototypu, wynikające z konieczności zastosowania większej ilości sterowników i wykorzystanych portów mikrokontrolera oraz skomplikowanie programu sterującego. Rozwiązanie to zostało zatem odrzucone. Drugim rodzajem jest platforma z zamontowanymi dwoma kołami napędowymi oraz jednym kołem obrotowym. Konstrukcja ta pozwala na obrót pojazdu w miejscu.

Zarówno napęd gąsienicowy, jak i kołowy spełniają założenia pracy, polegające na zastosowaniu napędu pozwalającego na poruszanie się po nierównym terenie. Oba porównywane rozwiązania opierają się na sterowaniu niezależnym dwoma silnikami. Zatem niezależnie od zastosowanego napędu, nie zmienią się pozostałe elementy prototypu oraz zastosowany algorytm sterujący. Z uwagi na niższą cenę do budowy prototypu została wykorzystana podwozie Chassis Rectangle 2WD, posiadające platformę montażową wykonaną z akrylu, dwa koła z oponami o średnicy 65 [mm], znajdujące się z tyłu pojazdu, dwa silniki prądu stałego z przekładniami oraz jedno metalowe koło obrotowe, znajdujące się z przodu pojazdu. [2]

2.2 Silniki elektryczne

Silniki elektryczne z przekładnią zostały dostarczone w komplecie z platformą mobilną. Są bardzo często stosowanym źródłem napędu amatorskich robotów, ze względu na swój niewielki rozmiar, energooszczędność oraz niską cenę, w porównaniu do profesjonalnych silników.



Rys. 2.2 Schemat budowy szczotkowego silnika DC. [3]

Typowy silnik prądu stałego jest zbudowany z:

- **wirnika,** odpowiadającego za przewodzenie prądu na zewnętrzną oś,
- **uzwojenia,** które jest nawinięte na wirniku,
- **stojana,** zawierającego magnesy stałe,
- **komutatora,** przełączającego bieguny na uzwojeniu,
- **szczotek,** które doprowadzają zasilanie do komutatora,

Silniki wykorzystane w projekcie posiadają także przekładnie mechaniczne, służące do uzyskania większego momentu obrotowego kosztem zmniejszenia prędkości obrotowej.

Do zacisków silnika DC zostaje podłączone zasilanie. Silnik rozpoczyna ruch w momencie, pokonania przez siłę elektromagnetyczną, siły oporu statycznego jaką działa wał silnika. Następnie prąd płynie kolejno przez szczotki, styki komutatora i uzwojenia. W tym momencie wokół wirnika wytworzone zostanie pole elektromagnetyczne. Pole to w wyniku występowania siły magnetycznej oddziałuje z magnesami stałymi w stojanie, poruszając wirnik, wraz z pozostałymi elementami silnika. W trakcie obrotu komutator zmienia polaryzację prądu, przez co cały proces zostaje powtórzony. Wykorzystane zostaje tu bardzo proste zjawisko fizyczne, polegające na odpychaniu się dwóch takich samych biegunów magnetycznych. Silnik szczotkowy prądu stałego posiada również wady. Silnik ten prawidłowo przełącza bieguny, jeśli pole magnetyczne jest wystarczająco silne do rozpędzenia wirnika do odpowiedniej prędkości. Z powodu wysokiej prędkości obrotowej, wymagane jest wykorzystanie opisanej wyżej przekładni. W sterowaniu prędkością obrotową silnika niepraktyczne jest zastosowanie stabilizatora napięcia. Do tego celu został wykorzystany układ z wbudowanym mostkiem H. [4]

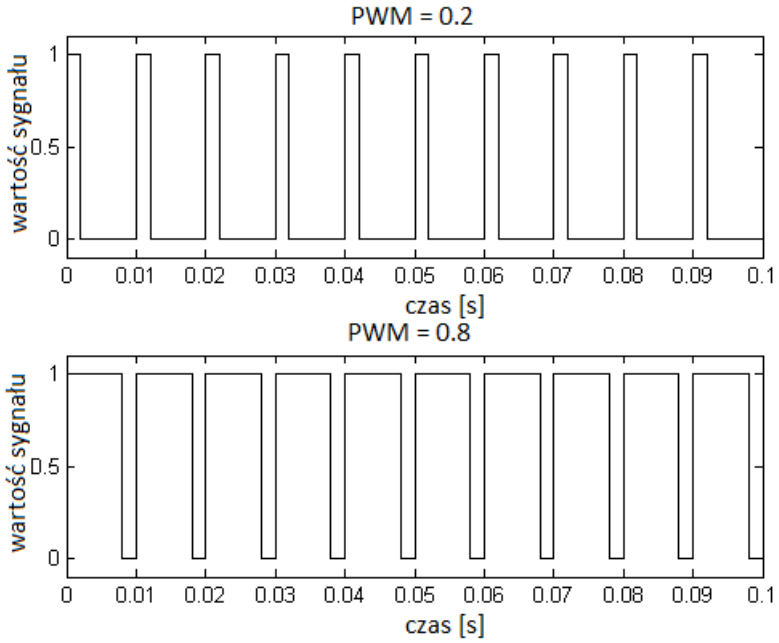
2.3 Kontroler silnika DC

Mostek H na podstawie sygnału PWM, reguluje napięcie, a dzięki temu zmienia prędkość obrotu silnika. W celu uzyskania odpowiedniego przebiegu sygnału, dobiera się właściwe wypełnienie, przy zachowaniu stałej wartości częstotliwości oraz amplitudy. Dobór wypełnienia jest istotny, ponieważ pojazd poruszający się ze zbyt wysoką prędkością, spowoduje spadek dokładności wykonanych pomiarów przejechanej odległości, ponieważ układ pomiarowy nie będzie w stanie wychwycić wszystkich impulsów, a układ z czujnikiem metalu nie wykona pomiarów we właściwym miejscu. Zbyt niski współczynnik, może być powodem nie osiągnięcia wystarczającej siły do poruszania pojazdem. [5]

Współczynnik wypełnienia impulsu jest to stosunek wypełnienia impulsu do jego okresu, zgodnie ze wzorem:

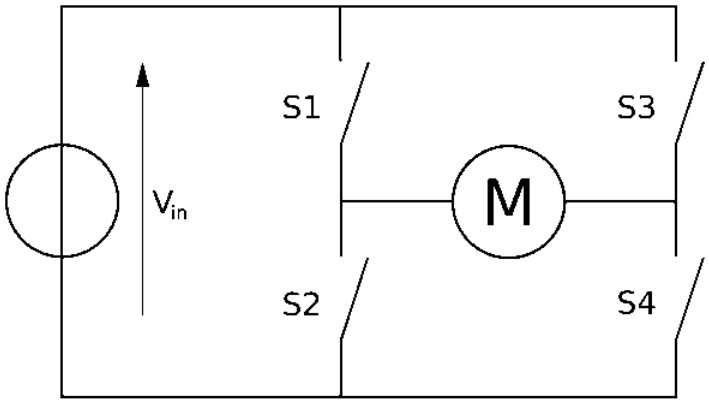
$$k_w = \frac{\tau}{T} \tag{2.1}$$

gdzie:
τ – czas trwania impulsu [s]
T – okres sygnału [s]



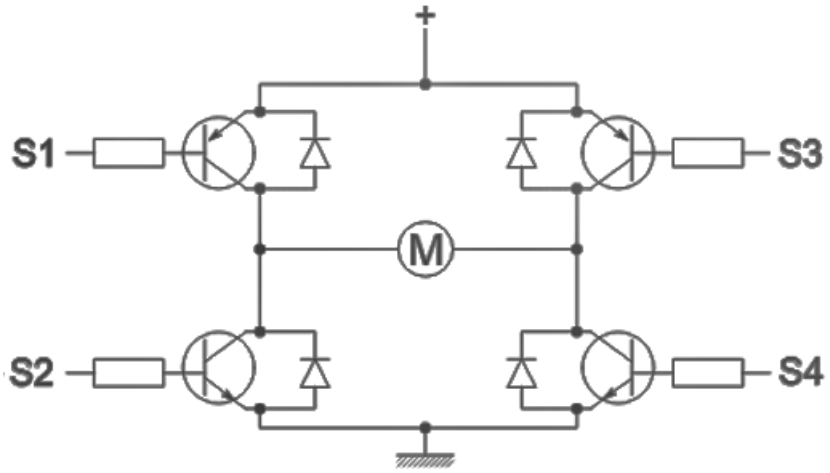
Rys 2.3 Wykres przebiegu sygnału PWM. [6]

Mostek H pozwala na sterowanie kierunkiem obrotu silnika elektrycznego.



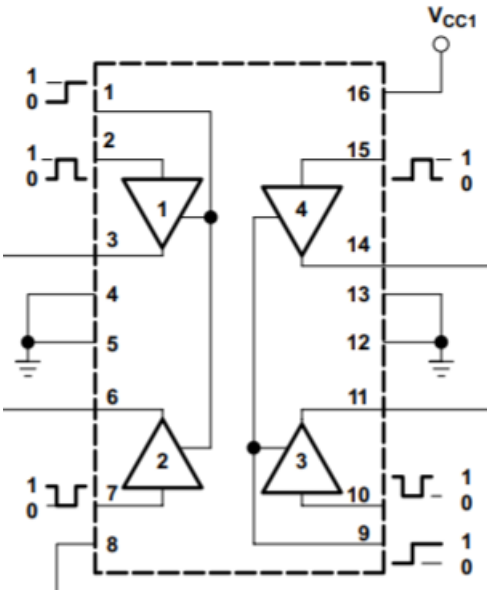
Rys. 2.4 Schemat ogólny mostka H. [7]

Zamykając jedną z par styków znajdujące się po przekątnej (S1 i S4 lub S2 i S3), a drugą parę pozostawiając otwartą, możemy zmienić kierunek przepływu prądu przez silnik. Dzięki zmianie polaryzacji prądu możemy regulować kierunek działania silnika. Układy te są zbudowane z elementów półprzewodnikowych takich jak tranzystory bipolarne lub MOSFET, natomiast rzadziej są to elementy mechaniczne. [8]



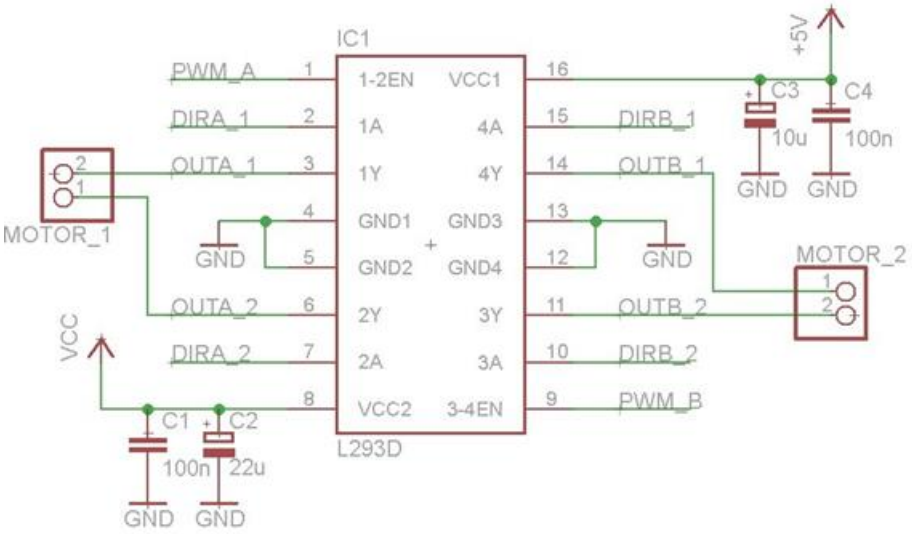
Rys 2.5 Schemat mostka H z tranzystorami bipolarnymi. [9]

Sygnał PWM oraz metoda sterowania mostkiem H są zastosowane w układzie L293D. Układ ten dzięki swojej konstrukcji, pozwala na uruchamianie, sterowanie prędkością i zmianą kierunku ruchu dwóch niezależnych zespołów napędowych lub uruchamianie i sterowanie prędkością czterech niezależnych zespołów napędowych. Do projektu został dobrany ze względu na pierwszą z tych możliwości. Dzięki temu można ułatwić wykonanie prototypu poprzez zmniejszenie ilości elementów, a w związku z tym obniżyć koszty wykorzystanych układów.



Rys 2.6 Schemat układu L293D. [10]

Elementy oznaczone 1 i 4 odpowiadają parze kluczy oznaczonych jako S1 i S2, a 2 i 3 – S3 i S4 z rys. 2.4. To jak są zbudowane w rzeczywistości elementy 1,2,3,4 jest tajemnicą producenta.



Rys 2.7 Schemat podłączenia sterownika. [11]

PIN		TYP	OPIS
NAZWA	**NUMER**		
1,2EN	1	WEJŚCIE	Sygnał PWM kanału 1
<1:4>A	2, 7, 10, 15	WEJŚCIE	Wybór kierunku ruchu silnika
<1:4>Y	3, 6, 11, 14	WYJŚCIE	Zaciski silnika
3,4EN	9	WEJŚCIE	Sygnał PWM kanału 2
GROUND	4, 5, 12, 13	-	Uziemienie
V_{CC1}	16	-	Zasilanie części logicznej
V_{CC2}	8	-	Zasilanie silników

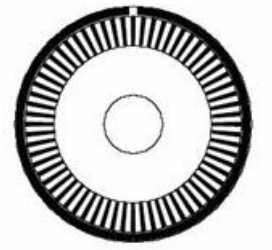
Tab2.1 Opis wyprowadzeń układu L293D. [12]

2.4 Pomiar rzeczywistej przejechanej odległości

Pomiar rzeczywistej przejechanej odległości był jednym z głównych założeń pracy. Wymagane było, aby pomiar odwzorowywał rzeczywistą przejechaną drogę. W terenie pojazd jest narażony na kontakt z wszelkiego rodzaju przeszkodami uniemożliwiającymi dalsze poruszanie się oraz możliwość utraty przyczepności w jednym z kół przez co może nastąpić zmiana kierunku ruchu. Najpopularniejszą metodą jaka jest stosowana w przypadku takich robotów jest nałożenie dysku enkodera na wał silnika. Przez takie rozwiązanie mierzona jest faktycznie ilość obrotów silnika, a nie pokonana droga. Rozwiązaniem tego problemu jest zaprojektowanie dodatkowej osi, która obraca się niezależnie od kół pomiarowych. Zamontowane na niej koła o dobrej przyczepności luźno toczą się po podłożu, poruszając dołączone elementy pomiarowe. Schemat tej osi znajduje się w rozdziale 3.2. W celu wykonania pomiaru zastosowane zostały dyski enkoderów ze szczelinami oraz optyczny moduł czujnika szczelinowego.

2.4.1 Dysk enkodera

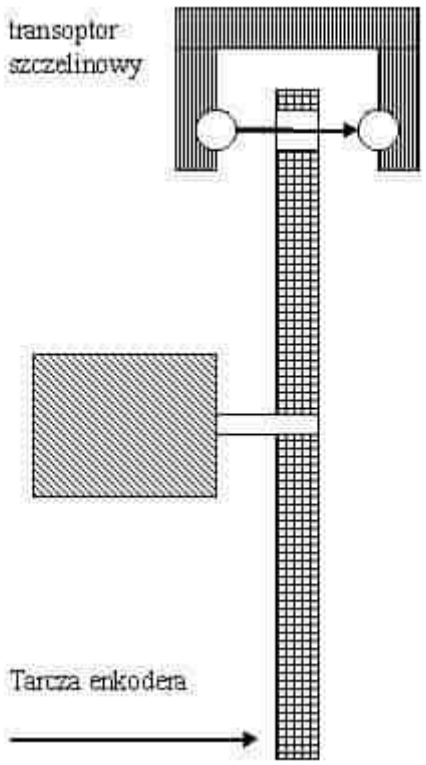
Dysk enkodera jest zbudowany z tworzywa sztucznego w kształcie koła, na którego obwodzie są wycięte szczeliny. Średnica dysku to 25 [mm], a grubość 3 [mm]. Został dobrany do pracy, ponieważ jest to jedyny dostępny na zamówienie model.



Rys 2.8 Schemat dysku enkodera. [13]

2.4.2 Optyczny czujnik szczelinowy

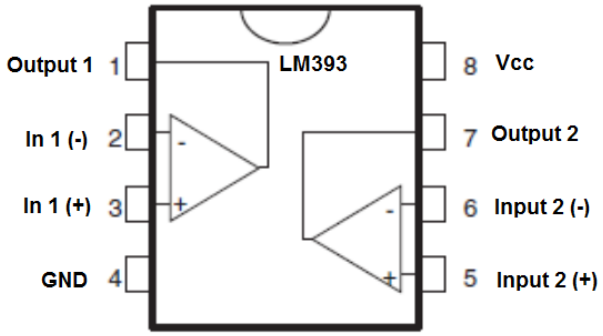
Optyczny czujnik szczelinowe służy do sprawdzania czy w szczelinie modułu następuje zmiana stanu dysku enkodera. Szerokość szczeliny, w której porusza się dysk to 10 mm.



Rys 2.9 Schemat montażu modułu wraz z dyskiem enkodera. [14]

Dwa popularne układy czujnika różniły się tylko obecnością wyjścia analogowego. Do pracy został wybrany układ bez tego wyjścia, ponieważ nie znalazło by ono zastosowania w projekcie. Układ ten posiada wyjście cyfrowe. Czujnik bada obecność szczeliny używając do tego układu z nadajnikiem w postaci diody oraz odbiornikiem, którym jest fototranzystor. W przypadku wykrycia szczeliny dysku enkodera, na wyjściu załącza się stan wysoki – 5 [V], a w przeciwnym wypadku stan niski – 0 [V]. Dekoder jest oparty na komparatorze LM393.

Układ LM393 posiada dwa niezależne komparatory porównujące napięcia. Poszczególne komparatory, posiadają dwa wejścia: jedno nieodwracające (In 1 (+)) oraz drugie odwracające (In 1 (+)) oraz jedno wyjście (Output 1). Jeżeli wartość napięcia na wejściu odwracającym jest większa od wartości na wejściu nieodwracającym, na wyjściu pojawia się stan wysoki. W pozostałych przypadkach na wyjściu pojawia się stan niski. [15]



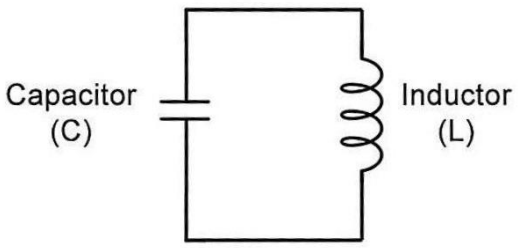
Rys 2.10 Schemat komparatora LM393. [16]

Wejście odwracające jest połączone z układem fototranzystora. Wejście nieodwracające podłączone jest do stałego źródła napięcia. W momencie wystąpienia szczeliny w dysku enkodera na wejściu *3* pojawia się napięcie wyższe od napięcia na wejściu *2*, czego skutkiem jest prądy płynący do wyjścia *1*. W sytuacji niewystąpienia szczeliny, prąd nie płynie do wyjścia *1*. Układ komparatora może pracować przy różnicy potencjałów do 36 [V]. Dokładne wartości napięć występujących w czujniku szczelinowym nie zostały podane przez producenta.

2.5 Czujnik metalu

Czujnik metalu jest konieczny do zrealizowania założeń projektu. W celu wykonania pomiaru rozważone zostały dwa rozwiązania. Pierwszym z nich było zastosowanie czujnika Halla. Czujnik ten wykrywa pole magnetyczne. Został odrzucony ze względu na wykrywanie elementów metalowych, które są namagnesowane, oraz ze względu na bardzo małe pole skanowanej powierzchni. Drugim rozwiązaniem jest zastosowanie układu z cewką. Ten element elektroniczny wykrywa zmianę pola magnetycznego jakie wytwarza się wokół niego, a którą to zmianę powoduje pojawienie się elementów metalowych, na które cewka oddziałuje. Zaletą takiego rozwiązania jest także możliwość montażu cewki o dowolnej średnicy.

Do zbudowania czujnika wykrywającego elementy metalowe został wykorzystany układ rezonansowy, w którym szeregowo został połączony kondensator i cewka.

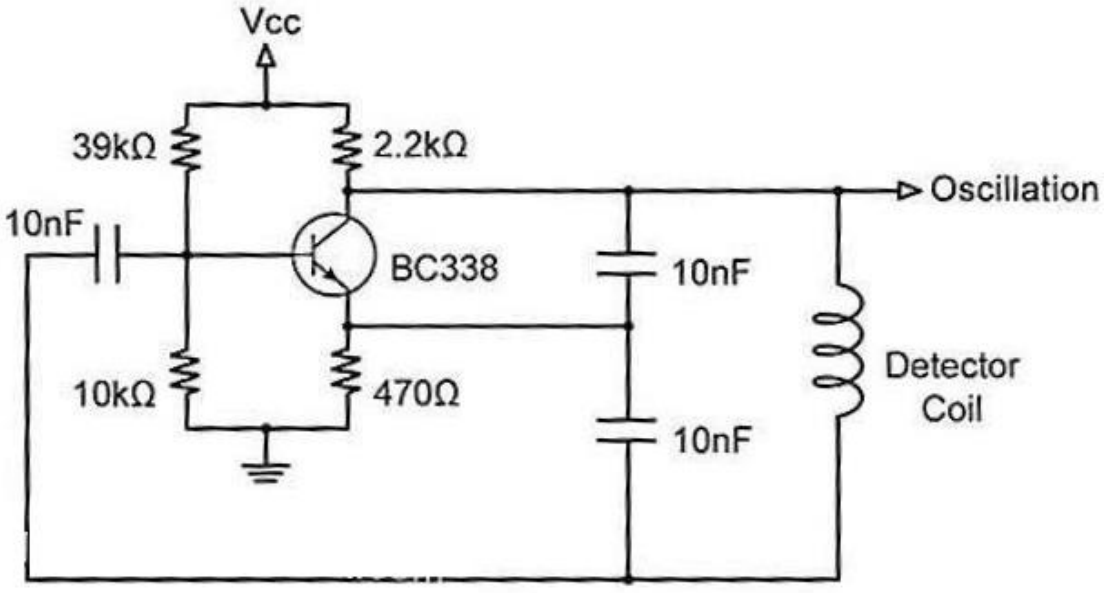


Rys 2.11 Układ rezonansowy z cewką i kondensatorem. [17]

W zaprezentowanym układzie, w przypadku, gdyby kondensator i cewka były by elementami idealnymi, energia mogła by być przesyłana między nimi wielokrotnie, w nieskończoność. W momencie rozładowywania się kondensatora, prąd przepływa przez cewkę, wokół której powstaje pole magnetyczne. Pole to, w przypadku całkowitego rozładowania się kondensatora, zaczyna maleć, dzięki czemu utrzymuje przepływ prądu. Kondensator zaczyna się ładować z odwrotną polaryzacją niż poprzednio. Kiedy pole magnetyczne całkowicie zaniknie, zjawiska zaczynają zachodzić od początku, z tą jedną różnicą, że prąd płynie w odwrotnym kierunku. Jeżeli do tego induktora zbliżymy metalowy element, znajdzie się on w jej w polu magnetycznym. W tym przypadku zmieni się przepuszczalność magnetyczna rdzenia cewki, która wpłynie na jej indukcyjność, a zatem zmieni się częstotliwość oscylacji. Zastosowane elementy pasywne w rzeczywistości nie są idealne. W trakcie trwania oscylacji powstają straty energii, przez co oscylacje, po pewnym czasie wygasną. W celu zapobieżeniu temu efektowi zastosowano wzmacniacz odwracający, zbudowany z tranzystora bipolarnego BC338. Tranzystor ten pozwoli na wzmocnienie sygnału.

Parametry tranzystora BC338:
- Tranzystor bipolarny NPN
- Napięcie maksymalne kolektor-emiter V_{ce}: 25 [V]
- Prąd kolektora: 0,8 [A]
- Układ wyprowadzeń: CBE (kolektor, baza, emiter)

Cewka została wykonana z miedzi, ponieważ posiada ona bardzo korzystne właściwości:
- najwyższa wartość przewodności elektrycznej dla metali nieszlachetnych,
- elastyczność,
- zdolność do rozciągania,
- odporność na korozję (ważna przy pracy na zewnątrz).



Rys 2.12 Układ do pomiaru wraz z wzmacniaczem. [18]

Zastosowana konstrukcja opiera się na oscylatorze Colpittsa. Oscylacje, które występują w węźle przed cewką są przesunięte w fazie względem oscylacji występujących w węźle po cewce. W związku z tym należy zastosować pętlę zwrotną, która doprowadza te drgania do bazy tranzystora, który wzmacnia i odwraca sygnał, a następnie przekazuje go dalej do obwodu. Układ ten generuje oscylacje o stałej częstotliwości. Do zasilenia układu wykorzystano napięcie stałe o wartości 5 [V]. Elementy metalowe prototypu, nie wpływają negatywnie na wynik pomiaru, ponieważ są one stałe i uwzględnia je każdy wykonany pomiar. Nie jest wymagana znajomość indukcyjności cewki, ponieważ wynik pomiaru jest interpretowany przez program sterujący. [19]

2.6 Zasilanie

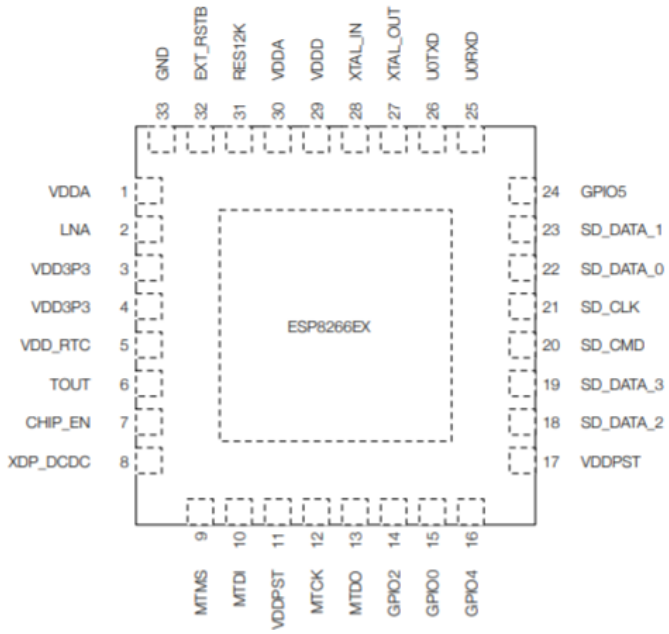
Do zasilenia prototypu wykorzystano dwa źródła prądu stałego. Pierwsze z nich służące do zasilenia silników, które wymagają napięcia między 4 V a 6 V oraz natężenia prądu między 190 mA a 250 mA każdy, zostało wykonane z czterech baterii alkalicznych o oznaczeniu LR6, połączonych szeregowo. Każde z tych ogniw ma napięcie znamionowe o wartości 1,5 [V], zatem ich szeregowe połączenie pozwala na uzyskanie na wyjściu napięcia 6 [V]. Baterie posiadają różną pojemność sięgającą rzędu 2500 [mAh]. Ogniwa te nie mają możliwość ładowania, ponieważ energia elektryczna, pochodzi z reakcji chemicznych, które nie są odwracalne. Do zasilenia pozostałych układów zastosowano jako źródło napięcia, wyjście napięcia 5 V znajdujące się na płytce Arduino, opisaną poniżej w rozdziale 2.7. Na jej wejście VIN zostało doprowadzone napięcie 8 [V], pochodzące z dwóch akumulatorów o napięciu 4 [V] każdy, połączonych szeregowo. Akumulatory posiadają dwa stany działania. Pierwszy z nich to praca, w trakcie której wydzielają energię elektryczną, pochodzącą z energii chemicznej. Natomiast drugi stan to ładowanie, kiedy energia elektryczna jest zamieniana na energię chemiczną.

Dzięki zastosowaniu dwóch różnych źródeł zasilania, układ ten nie ma problemów ze stabilnym działaniem. W przypadku, kiedy wszystkie elementy są zasilane z jednego źródła, mogą pojawiać się spadki mocy lub zakłócenia, związane z procesem rozpędzania się silników, zwłaszcza przy częstej zmianie kierunku poruszania. [20] [21]

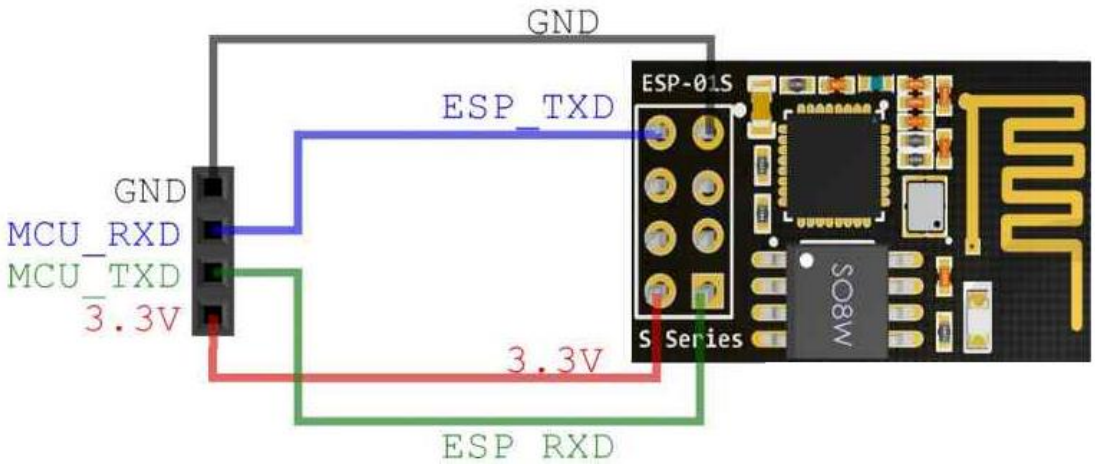
2.6 Komunikacja

Sieć Wi-Fi jest standardem pozwalającym na bezprzewodową komunikację między urządzeniami oraz budowaniu sieci komputerowych. W zależności od zastosowanych technologii pozwala na osiągnięci zasięgu od kilku metrów do kilku kilometrów.

Wybrany został najpopularniejszych obecnie moduł do komunikacji przez sieć Wi-Fi, układ scalony ESP8266, zamontowany na płytce PCB ESP-01S. Działa on w standardzie Wifi 802.11 b/g/n na częstotliwości 2,4 GHz. Posiada 3 wyprowadzenia GPIO będące wejściami/wyjściami cyfrowymi, 2 piny UART, które posłużą do komunikacji z mikrokontrolerem oraz wbudowaną pamięć Flash 1MB. Układ ten wymaga zasilania 3.3 [V]. [22]



Rys 2.13 Schemat mikroprocesora ESP8266E. [23]

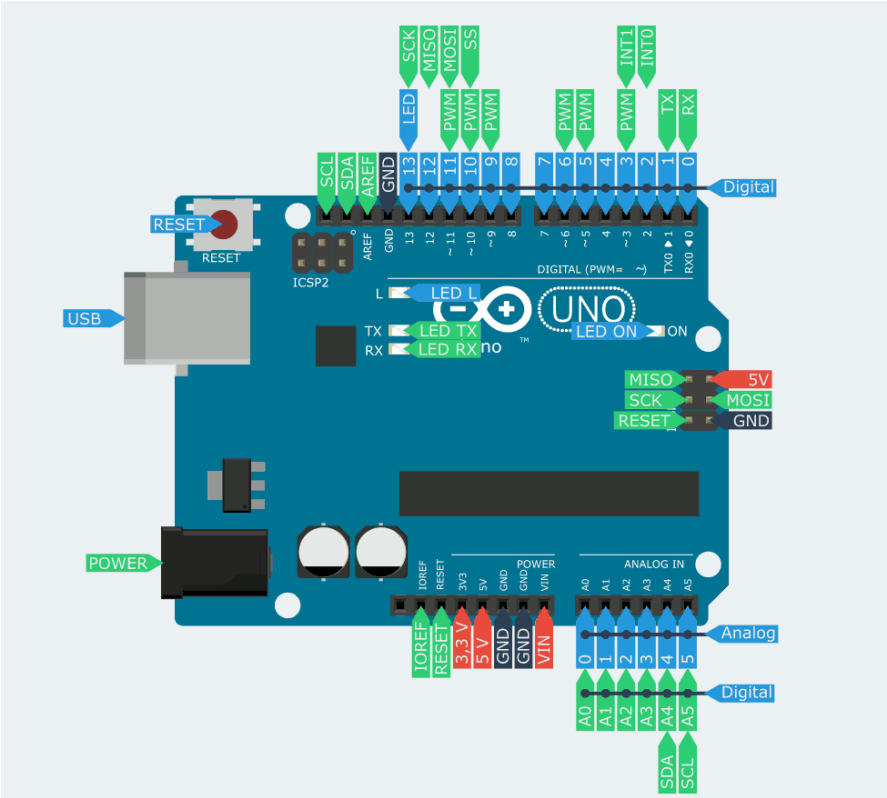


Rys 2.14 Schemat podłączenia układu. [24]

2.7 Moduł Arduino

Najważniejszym elementem całego projektu jest wybór odpowiedniego mikrokontrolera. Najlepszym do tego celu wyborem jest płytka z rodziny Arduino, która pozwoliła zaoszczędzić czas i środki jakie wymagało by wykonanie odpowiadającej jej płytki PCB z mikrokontrolerem. Znając wymaganą ilość potrzebnych wejść/wyjść cyfrowych, wyjść generujących sygnał PWM, wyjść analogowych oraz zasilających, dobrana została płytka Arduino UNO.

Arduino UNO jest programowalną płytką posiadającą wbudowany mikrokontroler ATmega328. Posiada 14 cyfrowych wejść/wyjść, z których 6 można zaprogramować jako wyjścia PWM oraz 6 wyprowadzeń analogowych. Taktowanie procesora wynosi 16 MHz, pojemność pamięci Flash to 32kB, a pamięć operacyjna SRAM to 2kB. [25]



Rys 2.15 Schemat Arduino UNO. [26]

Płytka ta posiada specjalne wyprowadzenia:

- wyjścia cyfrowe (PIN 0, 1) służące do komunikacji UART, pozwalające na komunikację dwustronną z komputerem lub modułem łączności bezprzewodowej,
- wyjścia PWM (PIN 3, 5, 6, 9, 10, 11) pozwalają w sposób sprzętowy, bez pośrednictwa procesora, generować sygnał PWM, o którym traktuje rozdział 2.2,
- wyprowadzenie zasilające VIN, na które może zostać podane zewnętrzne źródło zasilania o zalecanym przez producenta, napięciu między 7 [V] i 12 [V].

W celu zaprogramowania płytki, wykorzystany został program Arduino IDE. Program ten jest dostarczany na zasadach darmowej licencji, przez producenta płytek Arduino. Środowisko to pozwala na pisanie kodu, w językach C i C++, na mikrokontrolery zgodne z tym projektem. Program ten posiada bardzo rozbudowane biblioteki, ponieważ są one rozwijane przez społeczność.

Najważniejszym elementem tej płytki jest mikrokontroler zaprojektowany przez firmę Atmel, ATmega328. Rodzina tych procesorów składa się z ośmiobitowych mikrokontrolerów.

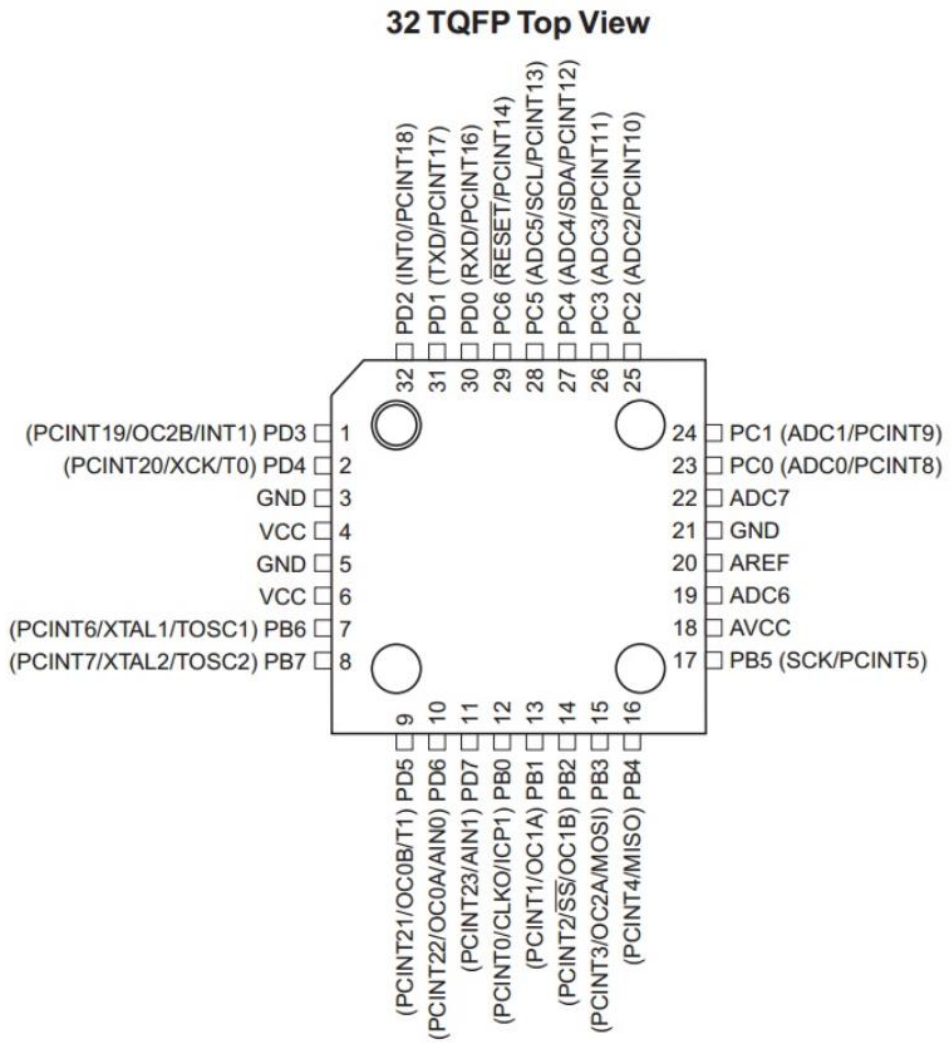
Procesor jest oparty na architekturze RISC co oznacza:

- zredukowanie liczby rozkazów do absolutnego minimum,
- zredukowaniu trybów adresowania,
- ograniczeniu do dwóch dedykowanych instrukcji mogących przesyłać dane pomiędzy pamięcią i procesorem, pozostałe instrukcje muszą wykorzystywać rejestry,
- zwiększenie liczby rejestrów.

Budowa całego układu opiera się na zasadach architekturze harwardzkiej co charakteryzuje się:

- rozdzieleniem pamięci programu i pamięci danych,
- zastosowaniu dwóch oddzielnych magistrali adresowych,
- szybszym działaniem od układów opartych na architekturze von Neumanna
- możliwością jednoczesnego pobierania rozkazów i danych,
- znajduje szerokie zastosowanie w procesorach sygnałowych i mikrokomputerach z pojedynczym układem scalonym.

Rodzina tych procesorów posiada ośmiobitowe mikrokontrolery. Przesyłanie danych z pamięci odbywa się tylko i wyłącznie z wykorzystaniem rejestrów. Są one powszechnie wykorzystywane za względu na łatwość użytkowania i programowania. [27]



Rys 2.16 Opis wyprowadzeń Mikrokontroler AVR - ATmega328P-AU SMD. [28]

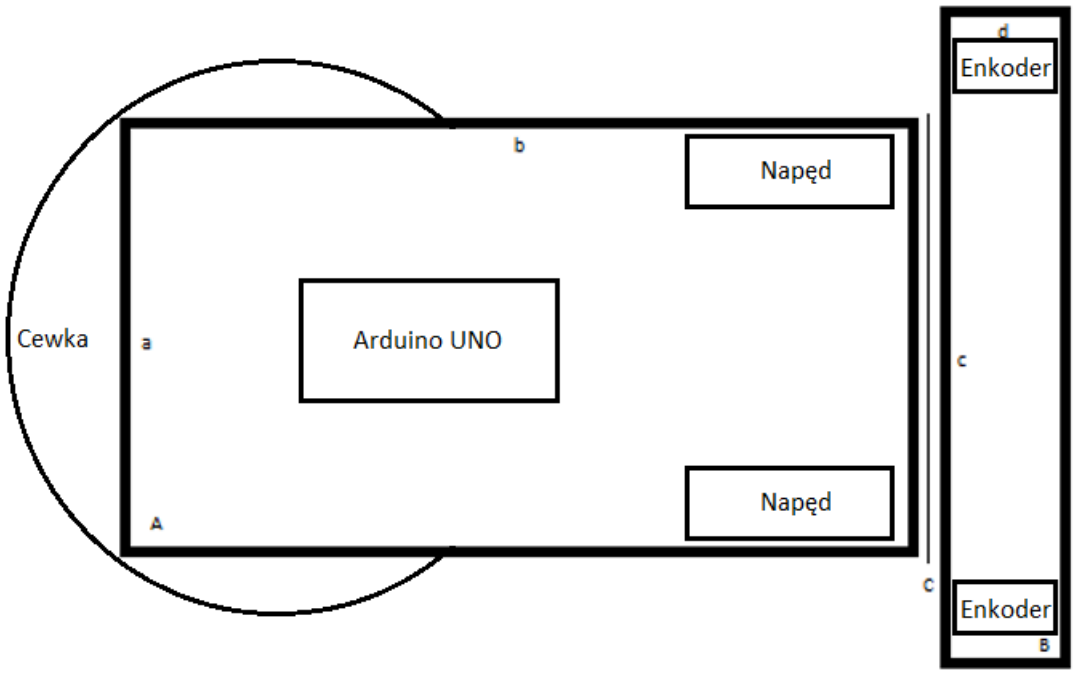
Specyfikacja:

- zasilany napięciem - 1,8 [V] – 5,5 [V],
- taktowanie - 16 [MHz],
- pamięć Flash – 32 [KB],
- 23 linie wejść/wyjść, w tym:
    - 6 kanałów PWM
    - 6 kanałów 10-bitowego przetwornika analogowo-cyfrowego,
- Interfejs do komunikacji: USART, SPI, TWI.

3. Część projektowa

3.1 Montaż platformy

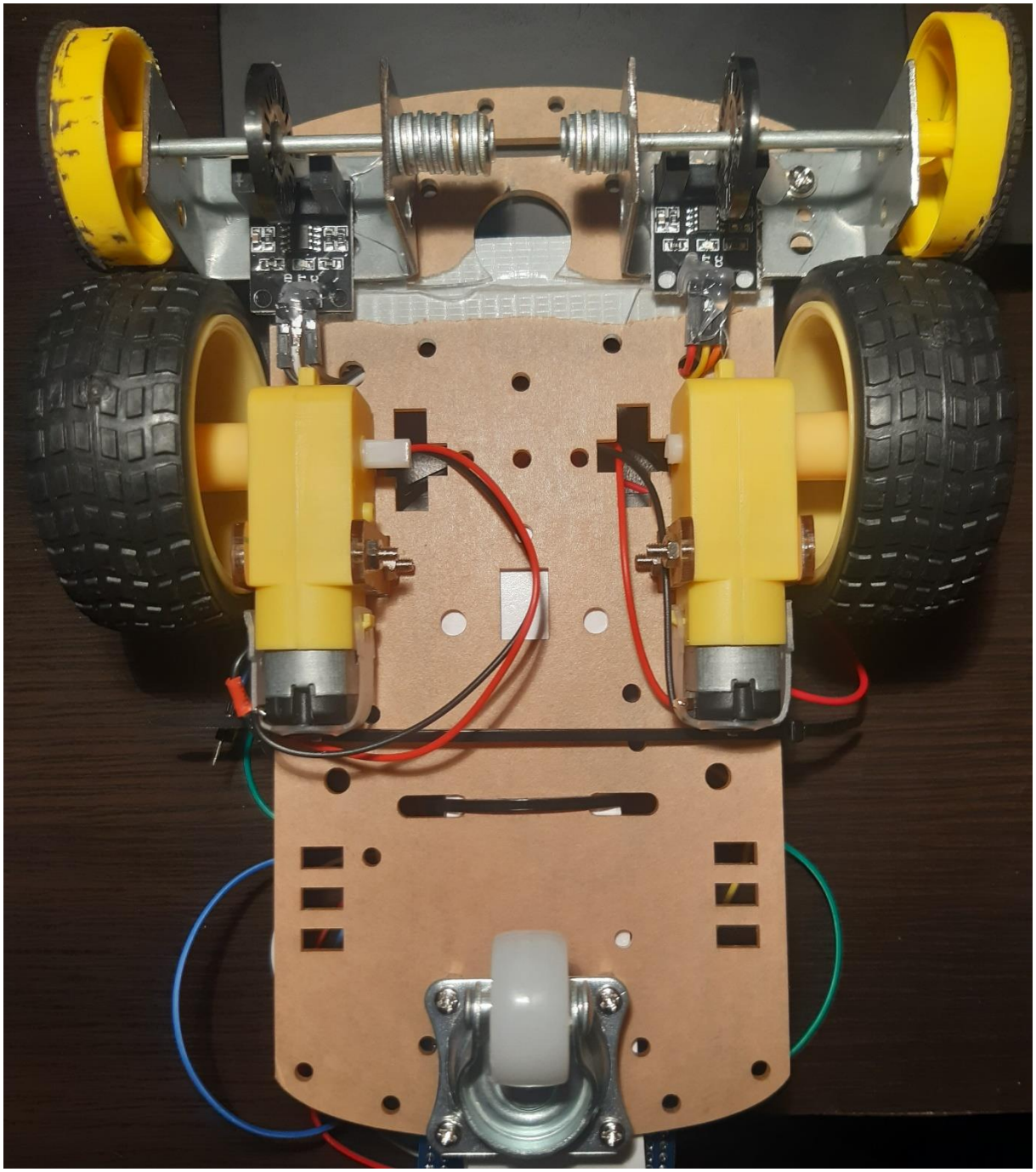
Całość projektu została wykonana, na wspomnianej wcześniej platformie Chassis Rectangle 2WD. Jest ona wykonana z tworzywa sztucznego i umożliwia w łatwy sposób montaż urządzeń peryferyjnych, dzięki wykorzystaniu gotowych otworów montażowych, czy też wywierceniu własnych. Platforma ta zastała podzielona na dwa elementy połączone ze sobą elastyczną taśmą. Większy z elementów posiada elementy napędowe, sterujące pojazdem oraz wykrywacz metalu. Na mniejszym elemencie zostały zamontowane osie z elementami służącymi do pomiaru przejechanej odległości. Dzięki zastosowaniu elastycznej taśmy, część pomiarowa ma możliwość utrzymywania stałego kontaktu z podłożem i nie wywiera wpływu na poruszanie się prototypu.



Rys3.1 Projekt platformy

gdzie:

A – część główna, o wymiarach a = 100 [mm] i b = 160 [mm],
B – część do pomiaru przejechanej odległości, o wymiarach c = 150 [mm] i d = 40 [mm],
C – taśma łącząca elementy A i B.

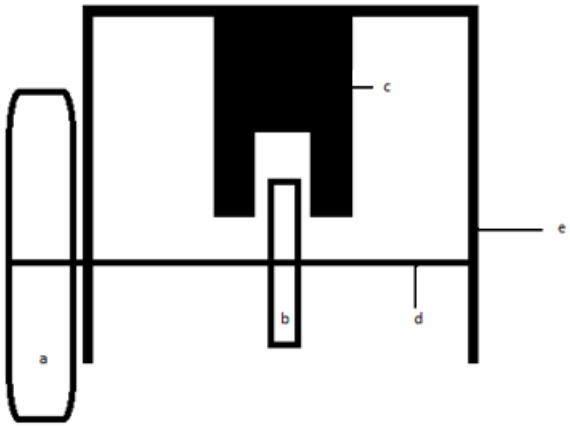


Rys 3.2 Rzeczywisty wygląd prototypu (widok z dołu)

Koła napędowe z oponami, zostały zamocowane na wcisk na wale silników. Silniki zostały przymocowane za pomocą uchwytów w prawej części modułu A (według rys. 3.1). W części lewej tego modułu, na środku zostało zamocowane metalowe koło obrotowe. Dzięki takiej konstrukcji pojazd posiada możliwość obrotu w miejscu, przy użyciu silników kręcących się w przeciwnych kierunkach. Dzięki takiemu zastosowanemu rozwiązaniu, nie wystąpiła potrzeba montażu dodatkowych silników w przedniej części pojazdu lub osi skrętnych, co w znaczący sposób uprościło konstrukcję prototypu oraz napisanie funkcji sterujących.

3.2 Montaż czujników odległości

W module B zostały zamontowane dwie niezależne od siebie osie z układem do pomiaru rzeczywistej przejechanej odległości. Konstrukcja każdej z nich została zaprojektowana przy zastosowaniu dwóch kątowników z wywierconym otworem na oś. Przy obliczaniu odległości, jaka jest między podstawą pojazdu na otworem na oś, należało wziąć pod uwagę średnicę kół napędowych, średnicę kół pomiarowych oraz średnicę dysku enkodera i wysokość, na której znajdowała się dioda czujnika optycznego. Z uwagi na pomijalnie małe opory tarcia, taki sposób wykonania nie wymagał wykorzystania dodatkowych łożysk między uchwytem, a osią. Sama oś została wykonana z metalowego gwoździa, na którym na wcisk, w środkowej części spasowano dysk enkodera. Przy jednym z jego końców, przed główką nałożone zostały podkładki służące do regulacji długości wysunięcia koła pomiarowego. Samo koło pomiarowe zostało nałożone na drugim końcu osi. Posiada ono średnicę mniejszą od koła napędowego. Początkowo miało oponę o szerokości 10 [mm], jednakże wywierało ono za duże tarcie na podłoże, przez co powodowało problemy z obrotem. Rozwiązaniem było usunięcie części opony i pozostawienie paska o szerokości 2 [mm], co rozwiązało występujący problem, jak również w żaden sposób nie wpłynęło na wykonywane ruchy. Pośrodku zmontowanej konstrukcji, został zamocowany moduł czujnika szczelinowego, w sposób umożliwiający mu wykonywanie pomiarów zmiany stanu dysku enkodera.



Rys. 3.3 Rysunek podglądowy konstrukcji do pomiaru odległości

gdzie:

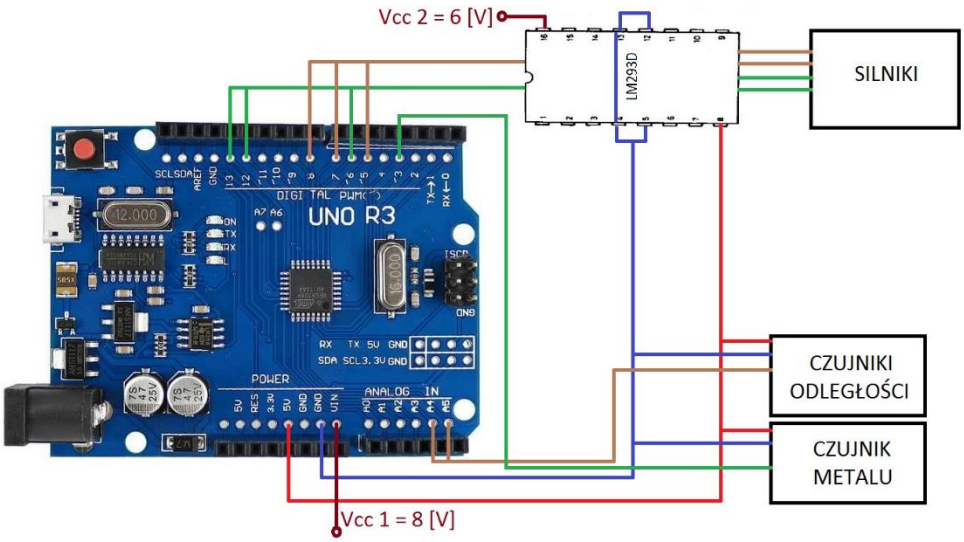
- *a – koło,*
- *b – dysk enkodera,*
- *c – czujnik szczelinowy,*
- *d – oś,*
- *e – uchwyt.*

Pierwsza wersja prototypu została zbudowana z jednego kawałka tworzywa, bez podziału na części A i B. W trakcie testów takiego rozwiązania, które to polegały na jeździe prosto i wykonywaniu skrętów na płaszczyźnie, z przeszkodą w postaci otwartego zeszytu okazało się, że sztywność podstawy łączącej koła powoduje szereg problemów. Zostały zaobserwowane przypadki unoszenia się kół napędowych, w przypadku kontaktu kół pomiarowych z wysokością nieprzekraczającą 10 [mm] lub z płaszczyzną w kształcie łuku. Wpływało to niekorzystnie na dwa aspekty: na wykonywany pomiar odległości oraz zmianę kierunku jazdy,

w szczególnych przypadkach takich jak utrata przyczepności tylko jednego z kół. W wyniku dogłębnego przeanalizowania problemu oraz wykonaniu kilku doświadczeń z obciążeniem, rozwiązaniem okazało się zmiana projektu i podzielenie podstawy na dwie odrębne części A i B. Podzielenie to zostało wykonane w miejscu rozszerzania się tworzywa, zgodnie z rys 3.1. Następnie obie części zostały połączone przy pomocy elastycznej taśmy. Zastosowanie takiego rozwiązania przyniosło pozytywne efekty, przy powtórzonym doświadczeniu element B pozostawał zawsze w stałym kontakcie z powierzchnią, nie powodował utraty przyczepności kół napędowych. Również w doświadczeniach, polegających na kręceniu się pojazdu wokół własnej osi, opory związane z obecnością gumowych opon na kołach pomiarowych zmalały, ponieważ elastyczne połączenie umożliwiało kilku stopniowe zginanie się. Podobno rozwiązanie konstrukcyjne można zaobserwować w elemencie przegubowym autobusu przegubowego.

3.3 Montaż elementów elektronicznych

Następnym wykonanym krokiem było wstępne podłączenie elementów związanych z pomiarem odległości oraz napędem, do płytki Arduino. Całość została zamontowana na płytce prototypowej, w oparciu o jedno źródło zasilania 6 [V]. Źródło to miało zasilać mikrokontroler, sterownik silników, dwa czujniki szczelinowe oraz silniki. W trakcie wykonania doświadczeń polegających na jeździe prosto, okazało się, że taka ilość urządzeń nie wpływa korzystnie na działanie układów. Silniki elektryczne potrzebowały napięcia 6 [V] do poprawnego działania, ale pozostałe układy, czyli mikrokontroler, enkodery, moduł Wi-Fi oraz układ z cewką skutecznie obniżała ilość dostarczonej energii do tego stopnia, że pojazd nie mógł rozpocząć ruchu bez nadania mu pewnej początkowej energii kinetycznej, pochodzącej z popchnięcia. Silniki te, podczas wielokrotnego rozpędzania się, w trakcie wykonywania skrętów, wywoływały szumy w zasilaniu, przez co zakłócona była praca pozostałych modułów. Problemem na dłuższą skalę okazało się również szybkie rozładowywanie się baterii i spadku wymaganego do pracy napięcia poniżej 5 [V]. W celu rozwiązania problemu zostało zastosowane oddzielne źródło napięcia dla silników i pozostałych elementów układu. Silniki zostały połączone przez układ sterownika do napięcia 6 [V]. Natomiast pozostałe elementy zostały podłączone do wyjścia 5 [V] płytki Arduino, która to przez wejście VIN została zasilona z dwóch akumulatorów, pozwalających na uzyskanie napięcia 8 [V]. Dzięki temu zostały wyeliminowane pierwsze dwa powyższe problemy. Natomiast żywotność baterii podłączonych do silników, znacząco wzrosła, a spadek napięcia wywoływał jedynie wolniejsze kręcenie się kół.

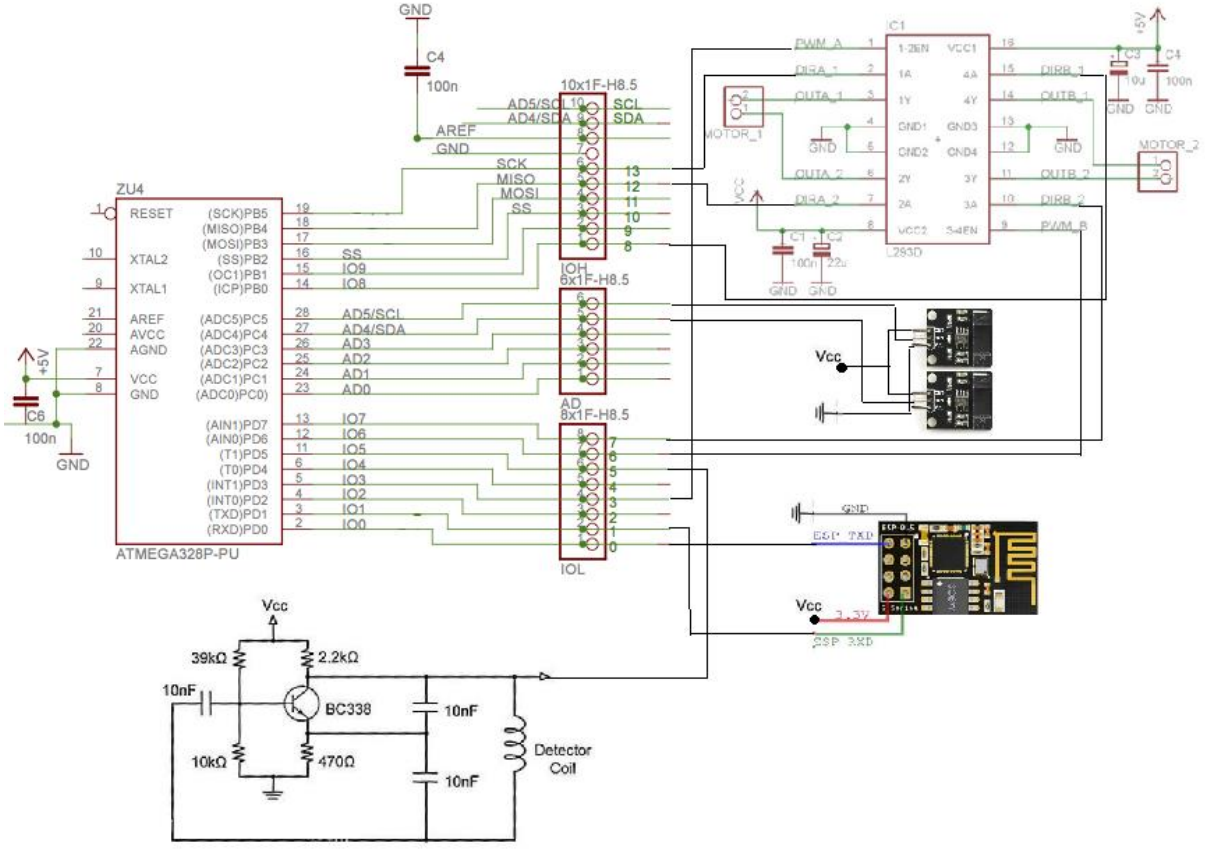


Rys 3.4 Schemat połączeń

gdzie:

- *na wejście VIN Arduino podano napięcie 8 [V],*
- *na wejście 16 LM293D podano napięcie 6 [V],*
- *napięcie 5 [V] z wyjścia Arduino, podano na wejścia V_{cc} LM293D oraz czujników*
- *wyjście GND Arduino, wyjście 8 LM293D, wyjścia GND czujników oraz ujemne bieguny baterii, podłączono do wspólnego potencjału masy,*
- *piny cyfrowe 3 i 5 Arduino, podano kolejno na piny 1 i 9 LM293D,*
- *piny cyfrowe 7, 8, 12, 13, podano kolejno na piny 10, 15, 7, 2 LM293D,*
- *piny 3, 6, 13,14 LM293D, połączono z zaciskami silników DC,*
- *wejścia analogowe A4 i A5 Arduino, podano na wyjścia OUT czujników odległości,*
- *wejścia analogowe A3 Arduino, podano na wyjście Oscillation (rys. 2.11) czujnika metalu.*

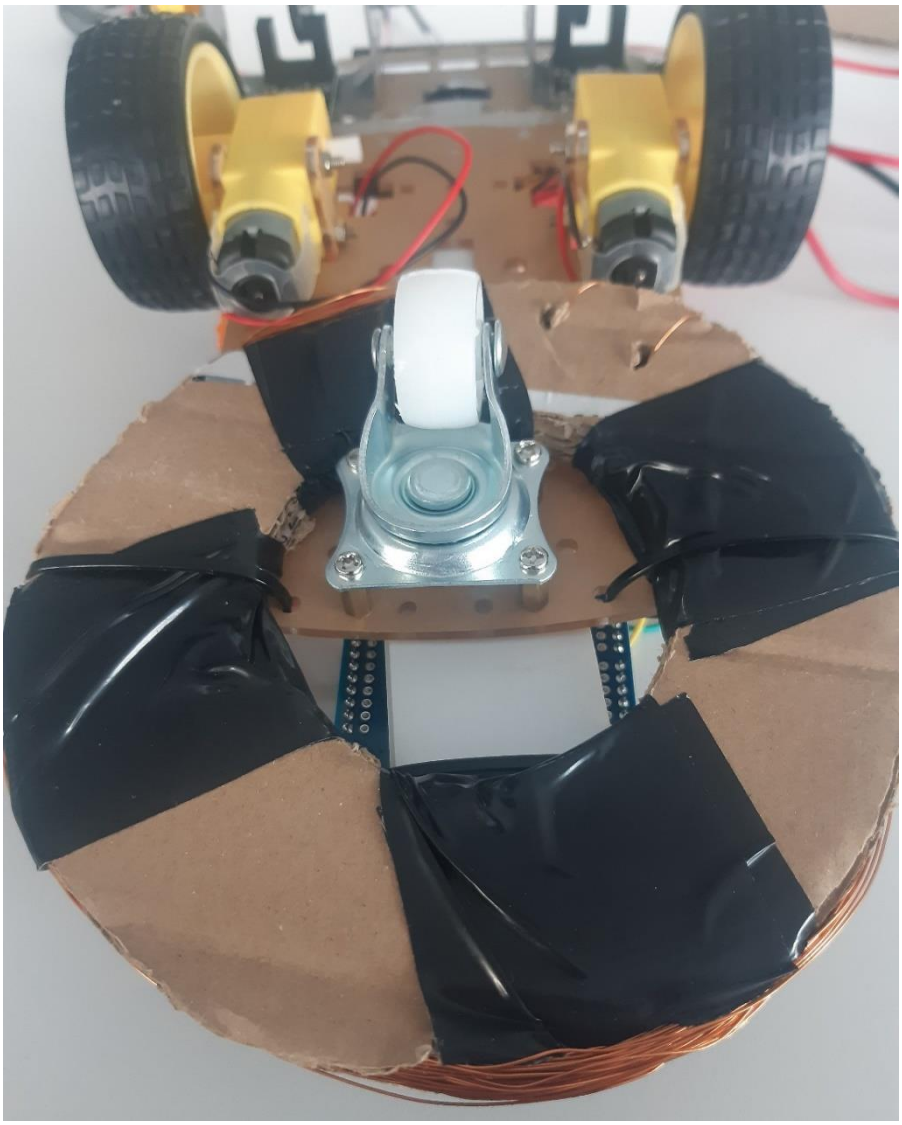
Piny 3 i 5 Arduino zostały skonfigurowane do pracy w sprzętowym trybie PWM.



Rys 3.5 Schemat ideowy [29] [11] [18] [24]

3.4 Montaż czujnika metalu

Szkielet cewki powstał z elementów tekturowych. Są to pierścienie o średnicy zewnętrznej 150 [mm] i średnicy wewnętrznej 60 [mm] oraz grubości 10 [mm]. Konstrukcja ta została opleciona 50 razy drutem nawojowym, miedzianym, emaliowanym o średnicy 0.4 [mm]. Tak wykonana cewka została zamontowana w przedniej części pojazdu wokół obrotowego koła. Kondensator oraz elementy wzmacniacza zostały zmontowane, zgodnie z rys 2.11, na płytce uniwersalnej. Następnie w trakcie testów układu, który polegał na zbliżaniu na różne odległości od cewki, elementów metalowych o różnej wielkości oraz wykonanych z różnych materiałów. Na podstawie otrzymanych wyników, w programie została dobrana odpowiednia czułość, będąca mnożnikiem otrzymanego wyniku, pozwalająca z jak największą dokładnością wykonywać pomiary. Połączenie z mikrokontrolerem odbywa się przez wejście cyfrowe 5 Arduino, ale program napisany w celu obliczenia częstotliwości został skonfigurowany bezpośrednio dla wyjścia T1 mikrokontrolera ATmega, które to zostało zaprojektowane jako licznik czasu. Program Arduino IDE pozwala na konfigurację bezpośrednio wejść i wyjść mikrokontrolera.



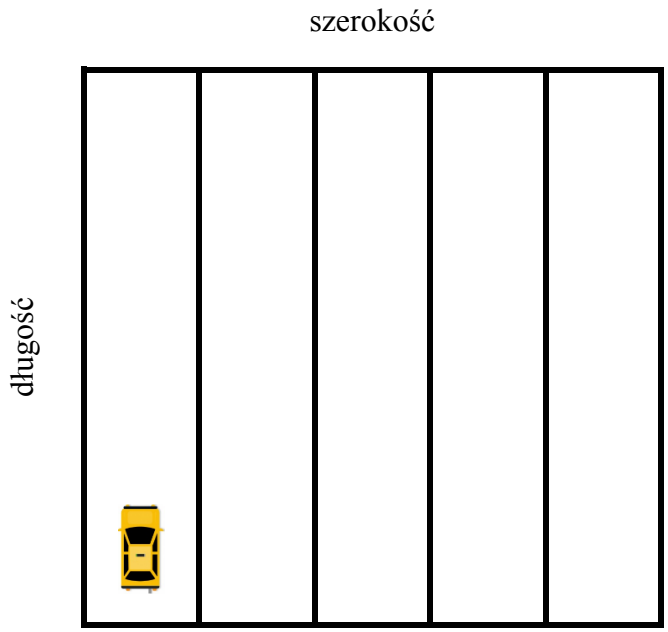
Rys 3.5 Wygląd cewki, miejsce i sposób montażu.

3.5 Schemat programu sterującego

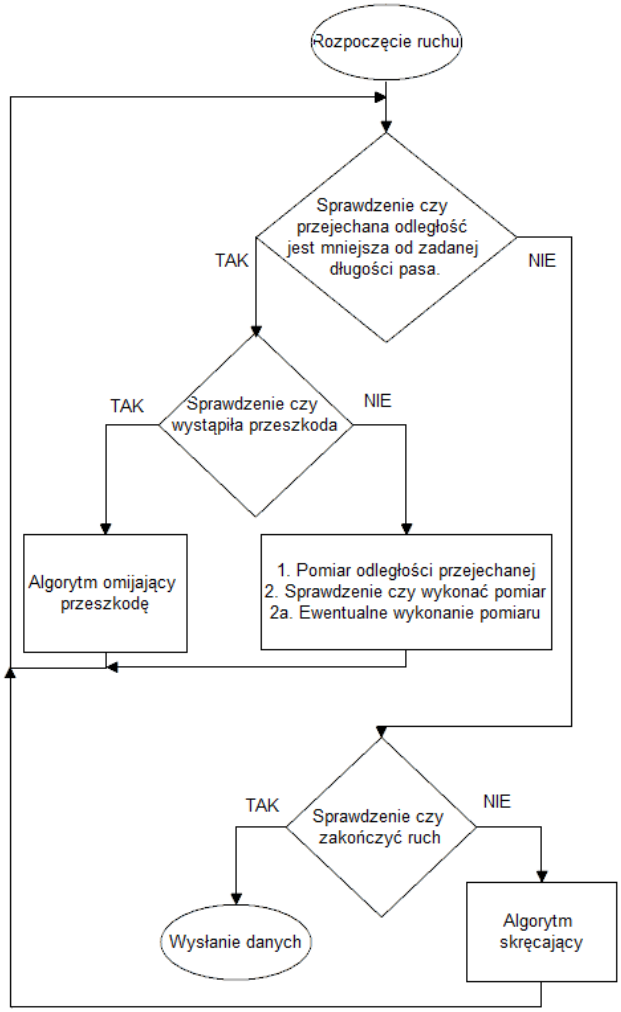
Program sterujący robotem potrzebował szeregu funkcji sterujących, wykonujących pomiary oraz określający położenie pojazdu w polu o kształcie prostokąta o zadanych przez operatora wymiarach. Określenie położenia wymagane było do odpowiedniego umiejscowienia wykonanego pomiaru oraz określenia czy pojazd powinien jechać dalej.

Algorytm działania.

Prototyp ma przyjętą stałą szerokość, jaką jest średnica cewki. Otrzymuje od operatora dwa parametry *długość* i *szerokość*, które są wymiarami prostokątnej płaszczyzny jaka ma zostać przeskanowana. Następnie na podstawie wartości *długość*, oblicza minimalną ilość pasów jakie musi przejechać. Pasy te mają długość o wartości *długość*. Dzięki otrzymanym wynikom buduje w pamięci mapę prostokąta podzieloną na mniejsze obszary. Pojazd zawsze zaczyna swoją pracę w lewym dolnym rogu prostokąta, oraz przy założeniu, że na obwodzie tego prostokąta nie znajdują się żadne przeszkody. Kolejne pomiary są wykonywane po przejechaniu odległości równej średnicy cewki.



Rys 3.6 Zrzut podziału płaszczyzny do pomiarów



Rys 3.7 Algorytm sterujący

Opis bloków:

Sprawdzenie czy przejechana odległość jest mniejsza od zadanej długości pasa:

funkcja mająca za zadanie wykonać pomiar przejechanej odległości i porównać go do zadanej długości pasa.

Fragment kodu, który odpowiada za jazdę do przodu:

```
while(kL <= limit)                    //pętla odpowiadająca za wykonywanie ruchu na odpowiednią odległość
{
  pomiar();                           //funkcja wykonująca pomiar czujnikiem metalu
  pomiarlewe();                       //funkcja wykonująca pomiar przejechanej odległości lewego koła
  pomiarprawe();                      //funkcja wykonująca pomiar przejechanej odległości prawego koła
  przeszkoda();                       //funkcja sprawdzająca czy oba koła są zablokowane

  if (testwspol() && testprze() )     //warunek sprawdzający czy wystąpiła przeszkoda w ruchu
    continue;
  else {
    if( !testwspol() ) {              //warunek sprawdzający czy przeszkoda dotyczy jednego koła
        testwspol_g();                //funkcja wykonująca korektę położenia jednego z kół
        break;
        }
    else if ( !testprze() ) {         //warunek sprawdzający czy przeszkoda dotyczy obu kół
      testprze_g();                   //funkcja wykonująca korektę położenia pojazdu
    }
  }
}
```

Sprawdzenie czy wystąpiła przeszkoda:

dwie funkcje sprawdzają, czy wystąpiła przeszkoda w ruchu, a następnie, jeżeli wstąpiła wykonuje odpowiedni **Algorytm omijający przeszkodę**:

- **testwspol()** - funkcja sprawdzająca czy pomiar odległości pokonanej przez oba koła jest równy sobie:
  1. jeżeli warunek jest niespełniony, oznacza to, że jedno koło straciło przyczepność i buksowało,
  2. algorytm zatrzymuje pojazd,
  3. wykonana zostaje korekcja położenia pojazdu, poprzez cofnięcie koła, które przejechało większą odległość,

- funkcja **przeszkoda()** zlicza ilość przejść pętli, w których przejechana odległość nie uległa zmianie, następnie funkcja **testprze()** sprawdza czy pojazd nie poruszył się przez 10000 przejść pętli,
  1. jeżeli warunek jest spełniony, oznacza to, że pojazd nie może kontynuować poruszania się danym pasem,
  2. algorytm przypisuje wartości *błąd* do pól pomiarowych, które w danym pasie nie zostały przeskanowane,
  3. następnie wycofuje pojazd na początek obecnego pasa,
  4. wykonuje algorytm skręcający,
  5. ustawia pojazd na początku następnego pasa.

Pomiar odległości przejechanej:

funkcja sprawdza jaki sygnał jest doprowadzony do wejścia enkodera, następnie porównuje ten sygnał, z sygnałem występującym przy poprzednim wywołaniu funkcji, jeżeli wystąpiło zbocze narastające lub opadające, zwiększa licznik przejechanej odległości i zapisuje nowy stan sygnału czujnika; jeżeli zbocze nie wystąpiło przechodzi dalej.

```
void pomiarlewe() {                       //funkcja odpowiadająca za pomiar przejechanej odległości przez lewe koło
  if (analogRead(A5) > 200 && pkL == 1) {  //sprawdzenie czy wystąpiło zbocze narastające
    skL=kL;                               //zapisanie wartości poprzedniej przejechanej odległości
    pkL=0;                                //zapisanie wystąpienia zbocza narastającego

    if(k == 0)                            //sprawdzenie czy pojazd porusza się do przodu k=0 czy do tyłu k=1
      kL++;
    else
      kL--;
  }
  else if (analogRead(A5) < 200 && pkL == 0)  {
    skL=kL;
    pkL=1;

    if(k == 0)
      kL++;
    else
      kL--;
  }
  else
    skL=kL;
}
```

Sprawdzenie czy wykonać pomiar:

funkcja sprawdza czy przejechana odległość od ostatniego punktu pomiaru jest większa od średnicy cewki, jeżeli tak **wykonuje pomiar**, następnie sprawdza czy wynik jest większy od częstotliwości odniesienia, jeżeli tak zapisuje pomiar,

Sprawdzenie czy zakończyć ruch:

funkcja sprawdzająca czy ilość przejechanych pasów jest równa ilości obliczonych pasów,

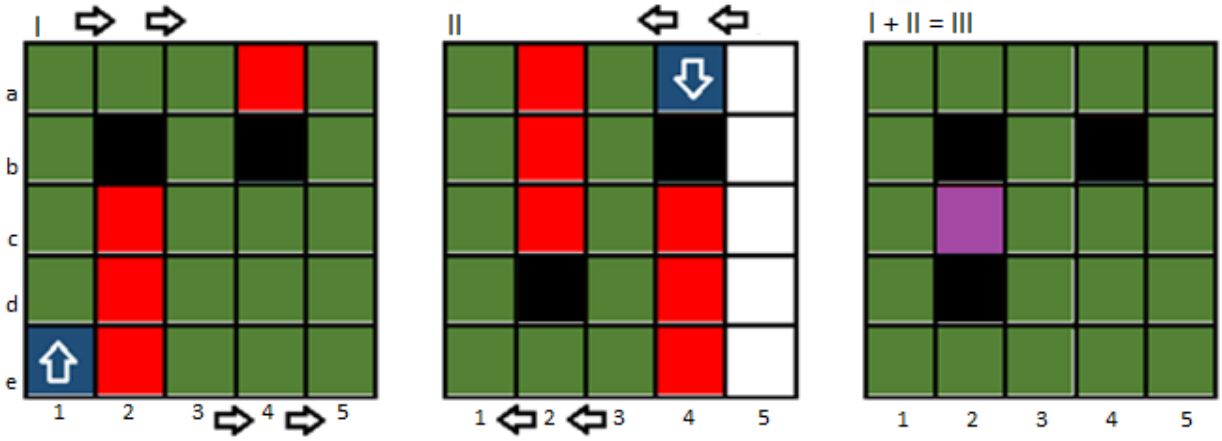
Algorytm skręcający:

jeżeli pojazd musi ustawić się na kolejnym pasie do przeskanowania, sprawdza jaki został wykonany ostatni skręt (pierwszy skręt jest zawsze w prawo), a następnie wykonuje algorytm skręcenia w miejscu o kąt 90° w odpowiednią stronę, podjechania na długość będącą szerokością pasa, a następnie ponowne wykonanie skrętu o kąt 90° w tą samą stronę co w pierwszym kroku, po wykonaniu algorytm zapisuje, w którą stronę skręt został wykonany.

```
skretprawo();                          //funkcja przypisująca parametry do skrętu w prawo

  while(kL < limit)
    pomiarlewe();

  zerowanie();                         //funkcja zerująca przejechaną odległość
  jazdaprzod();                        //funkcja przypisująca parametry do jazdy prosto o odległość równą długości pojazdu
  limit = pros;


  while(kL < limit)
    pomiarlewe();

  zerowanie();
  skretprawo();

  while(kL < limit)
    pomiarlewe();

  zerowanie();

  if (ps == 1)                          //zapamiętanie kierunku wykonanego skrętu
    ps=2;
  else if (ps == 2)
    ps=1;
```

Dodatkową funkcją polegającą na omijaniu napotkanych przeszkód jest algorytm wykonujący następujące kroki:

a. algorytm, na końcu każdego pasa, sprawdza czy ilość przeskanowanych pasów jest równa obliczonej wartości pasów do przeskanowania,
b. jeżeli pojazd dojechał do końca skanowanej powierzchni, sprawdza czy tablica zawierająca wyniki pomiarów zawiera wartości *błąd*,
c. jeżeli warunek jest spełniony, algorytm zwiększa liczbę pasów potrzebnych do przeskanowania dwukrotnie, a następnie zmniejsz wartość o jeden,
d. pojazd zostaje wysterowany tak aby ustawił się na bliższym krańcu pasa poprzedzającego pas końcowy,
e. pojazd wykonuje pomiary na zmniejszonej o jeden pas powierzchni, zapisując wartości pomiarów w drugiej tabeli,
f. w momencie przesłania wyników program porównuje otrzymane wyniki z dwóch tabel i zastępuje wartości *błąd* tabeli pierwszej, ewentualnymi wartościami z tabeli drugiej, wartości zawierające *błąd* w obu tabelach, pozostają niezmierzone.



Rys. 3.8 Kolejne kroki powstawania mapy terenu

gdzie:
kolor zielony - pole przeskanowane,
kolor czarny - wykryta przeszkoda,
kolor czerwony - pole nieprzeskanowane z powodu obecności przeszkody,
kolor niebieski - pole początkowe (zakładamy, że jest to również pole przeskanowane),
kolor biały – pole pominięte dla drugiego cyklu,
kolor fioletowy – pole niedostępne dla pojazdu,
strzałki – kierunek ruchu.

Założeniem tego algorytmu jest jak najdokładniejsze przeskanowanie powierzchni, na której występują przeszkody. Przykładowo, według rys 3.8, algorytm oblicza wymaganą ilość pasów równą 5, a pojazd rozpoczyna I fazę jazdy od pola *1e*. Po dojechaniu do pola *1a*, skręca w prawo i rozpoczyna jazdę na pasie *2* od pola *2a*. W trakcie jazdy, pomiędzy polami *2a* i *2b* wykrywa przeszkodę, uniemożliwiającą dalszą jazdę po pasie *2*. Wycofuje się na pole *2a*, skręca i ustawia się na pasie numer *3* na polu *a*. Następnie skanuje cały pas *3*, przejeżdża na pas *4* i skanuje go od pola *e*. W momencie dojazdu do pola *4b* wykrywa przeszkodę, wycofuje się na pole *4e*, skręca i rozpoczyna skanowanie pasa *5* od pola *e*. Kiedy dojeżdża do pola *5a*, sprawdza, że wystąpiły pola nieprzeskanowane. Zwiększa ilość pasów potrzebnych do przeskanowania według wzoru *5*2-1=9*. Faza II rozpoczyna się od ustawienia się pojazdu na polu *4a*. Pojazd zapisuje pomiar pola 4a, które nie było przeskanowane w fazie poprzedniej. Następnie wykonuje analogiczne operacje jak w fazie I. Po skończonym skanowaniu łączy otrzymane tabele pomiarów i uzupełnia brakujące wartości. Pola nieprzeskanowane w fazie I to *2 b, c, d, e* oraz *4 a, b*. Pola *2e* oraz *4a* udało się przeskanować w tracie fazy II. Pola *2b, d* oraz *4b* zostały oznaczone jako zawierające przeszkodę, natomiast pole *2c* jako pole niemożliwe do przeskanowania.

Zastosowanie powyższej funkcji pozwala na zmierzenie możliwie jak największego obszaru w przypadku wystąpienia przeszkód uniemożliwiających ruch pojazdu. Jednym z ograniczeń takiego rozwiązania jest założenie, że pola na obwodzie badanego prostokąta są pozbawione przeszkód. Wynika to z powodów, że zarówno pierwszy pas, jak i ostatni są pasami granicznymi obszaru poddawanego badaniom, a pojazd nie powinien wyjechać poza ten obszar oraz faktu, że algorytm wykonujący skręt nie zakłada istnienia przeszkód.

Fragment kodu, który zawiera funkcję przypisującą parametry do jazdy do przodu:

```
void jazdaprzod() {                    //funkcja przypisująca parametry do jazdy do przodu
digitalWrite(7, LOW);
digitalWrite(8, HIGH);

digitalWrite(12, LOW);
digitalWrite(13, HIGH);
limit=dl;                             //zmienna zapamiętująca odległość jaka ma zostać przejechana
}
```

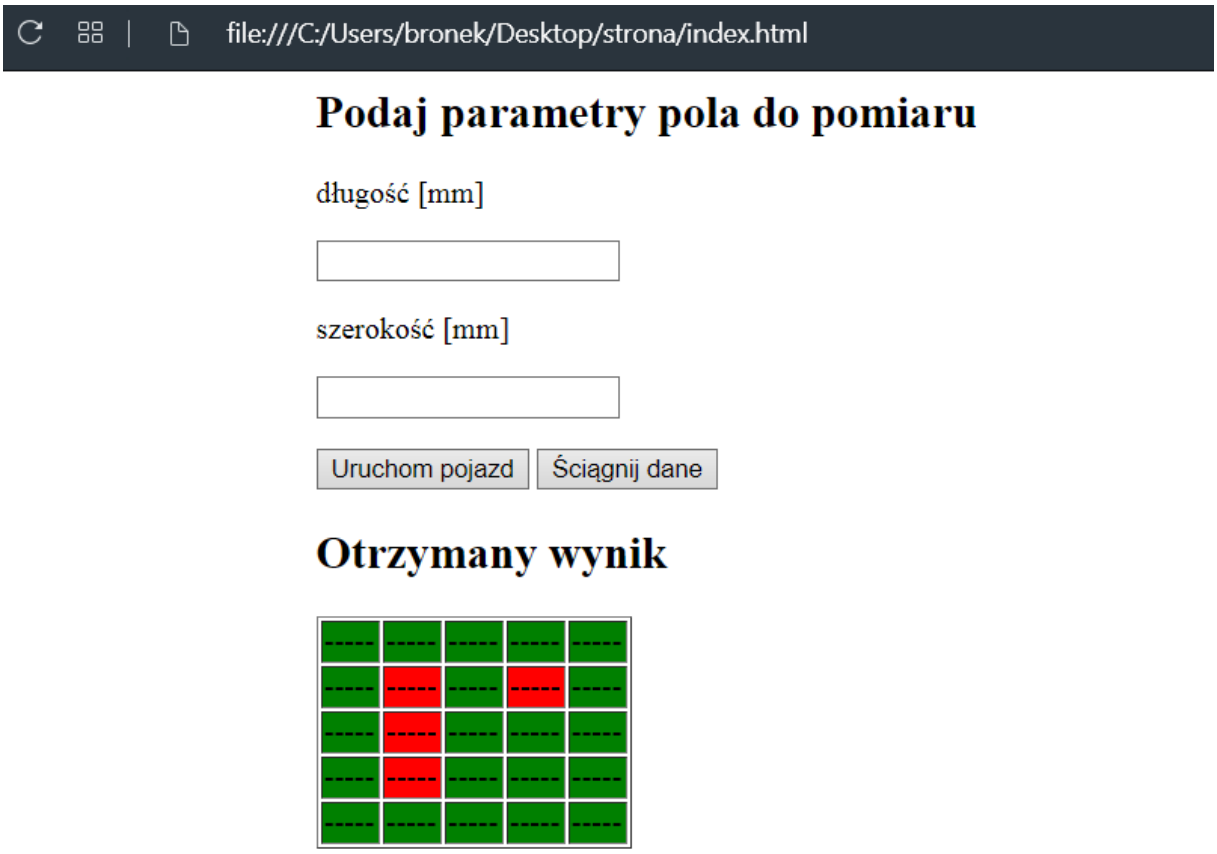
3.6 Interfejs użytkownika

Operator łączy się z zamontowanym modułem Wi-Fi, który jest skonfigurowany jako punkt dostępu, za pomocą dowolnego urządzenia, z którego jest możliwe nawiązanie połączenia oraz przeglądanie Internetu. Następnie za pomocą dowolnej przeglądarki otwiera przygotowaną stronę, która została napisaną w języku HTM, na której ma możliwość

ustawienia parametrów *długość* oraz *szerokość*, uruchomienia algorytmu sterującego oraz pobrania wyników pomiarów i wyświetleniu mapy przeskanowanego terenu.

```
<script type="text/javascript">
    $(document).ready(function(){
        $(".start").click(function(){
            var p = $(this).attr('id'); // pobranie wartości (dla jednego z pinów)

            // wysłanie HTTP GET request do adresu IP z parametrem "start" i wartością "1"
            $.get("http://192.168.4.1:80/", {start:1}); // wykonanie polecenia
        });
    });
</script>
```

Rys 3.9 Fragment kodu HTML, wysyłającego polecenie Start do mikrokontrolera. Adres IP został skonfigurowany w programie sterującym



Rys. 3.10 Strona do obsługi pojazdu wraz z przykładowym wynikiem testu

4. Doświadczenia

Już w trakcie budowania prototypu zostało przeprowadzonych wiele testów, polegających na jeżdżeniu prosto po wyznaczonej trasie oraz wykonywania skrętów. W trakcie zostało rozwiązanych kilka problemów technicznych, jakie były związane z wykorzystanymi elementami. Posłużyły one także w celu dobrania odpowiednich parametrów konfiguracyjnych użytych w programie sterującym, takich jak zbadanie jaka odległość jest mierzona przez enkodery w trakcie obrotu pojazdu o 360°. Następnie na podstawie tego pomiaru oszacowano parametr, wykorzystany w warunku do skrętu o 90°, który to później został ponownie doświadczalnie sprawdzony. Na tej samej zasadzie zostały określone przeliczniki, które są zastosowane przy przeliczaniu ilości zmierzonych obrotów na milimetry. W późniejszych etapach powstawania prototypu zaczęły dochodzić kolejne układy. Podczas pomiarów dokładności cewki, zaobserwowano, że nawet delikatne ruchy mogą wpływać na wynik pomiaru. W celu minimalizacji tych zakłóceń, funkcja wykonująca pomiar została usprawniona, w momencie pomiaru pojazd zatrzymuje się, odczekuje sekundę, następnie wykonuje kolejno dziesięć pomiarów w odstępnie pięciuset milisekund. Jeżeli przynajmniej osiem z tych pomiarów jest zgodnych co do tego, że pod czujnikiem znajduje się metalowy element, zostaje ten wynik zapisany do tablicy wyników. Ostatnim z testowanych elementów był układ Wi-Fi. Zostały sprawdzone dwa możliwe sposoby na komunikację. Pierwszy z nich polegał na stosowaniu układu jako serwera, co wiązało się z przechowywaniem strony, będącej interfejsem użytkownika w pamięci programu oraz udostępnianiu jej w sieci lokalnej. Rozwiązanie to zostało jednak odrzucone ze względu na potrzebę wykorzystania routera, który musiał pośredniczyć w połączeniu lub potrzeby ustawienia komputera czy telefonu w trybie punktu dostępu, a także konfiguracji modułu związanej ze stosowaniem zabezpieczeń w sieciach Wi-Fi. Kolejną przeszkodą był także brak możliwości połączenia komputera do zewnętrznej sieci internetowej. Zastosowane rozwiązanie, które polega na połączeniu się urządzenia kontrolującego z modułem pojazdu, dzięki czemu wystarczy odpali prostą stronę zapisaną w języku HTML. Bardzo dużą przewagą sieci Wi-Fi jest jej bardzo duży zasięg nawet w przypadku występowania przeszkód, w postaci ścian lub drzew, co w komunikacji Bluetooth prowadzi do utraty łączności. W trakcie testów całego prototypu zaobserwowano, że w przypadku pokonywania przez pojazd dużych odległości, ma on tendencje do skręcania w jedną ze stron. Jest to prawdopodobnie spowodowane różną budową silników elektrycznych. Pomimo wielu doświadczeń oraz regulacji za pomocą sygnału PWM, nie udało się w jednoznaczny sposób, określić prawidłowych parametrów, co może być przyczyną pewnych rozbieżności przy zastosowaniu pojazdu na znacznym obszarze. Doświadczalnie zostało także sprawdzone, jak układ cewki pomiarowej zachowuje się w obecności różnych rodzajów metalu. W zależności od zastosowanego elementu, wykonanego z żelaza lub miedzi, nie zostały zaobserwowane znaczące różnice w otrzymanych wynikach. Wykonanych zostało także kilka testów samego algorytmu sterującego, testy polegały na symulowaniu ruchu pojazdu poprzez podawanie na wejście odpowiadające za odbieranie sygnałów z enkodera, sygnału prostokątnego z innego źródła. Algorytm sterujący wszystkie testy przeszedł pozytywnie i nie znaleziono w nim uchybień.

5. Wnioski

Pojazd wraz z oprogramowaniem jest gotowym produktem mogącym znaleźć zastosowanie w wielu dziedzinach życia codziennego, od hobbystycznego poszukiwania skarbów, przez prace archeologiczne czy też jako sprzęt wojskowy. Niewielki rozmiar, duża dokładność oraz niedroga produkcja są znaczącymi atutami i mogą pozytywnie wpływać na doświadczenia związane z użytkowaniem robota. Stworzone oprogramowanie pozwala na zastosowanie szerokiej gamy różnych części, pozwalając na rozwój projektu. Może się ono, w przyszłości, stać odrębnie rozwijanym produktem. Zbudowany prototyp spełnia wszystkie założenia projektowe jakimi były: przeskanowanie zadanego terenu, omijanie przeszkód znajdujących się na drodze, jazda po nierównym terenie oraz budowa mapy terenu i przesyłanie jej do komputera. Dodatkowymi atutami są niewątpliwie możliwość komunikacji bezprzewodowej. W celu dalszego rozwoju projektu, należało by zastanowić się nad wykorzystaniem części stworzonych z materiałów wytrzymalszych czy też dodanie osłon umożliwiających pracę przy złej pogodzie. Algorytm sterujący można rozwinąć o dodatkowe funkcje takie jak wyznaczanie drogi do pól, które nie zostały zmierzony oraz dodanie podglądu danych na żywo czy też skanowanie powierzchni o kształcie innym niż prostokątne. Wraz z rozwojem mikrokontrolerów projekt ten może zostać udostępniony na zasadach otwartej licencji, co pozwoli zainteresowanym ludziom na zbudowanie takiego pojazdu we własnym domu.

5. Wykaz

5.1 Literatury

[2]https://botland.com.pl/pl/podwozia-robotow/7283-chassis-rectangle-2wd-2-kolowe-podwozie-robota-z-napedem.html
[4] https://forbot.pl/blog/jak-dzialaja-szczotkowe-silniki-dc-budowa-i-pomiary-id6788
[5] https://pl.wikipedia.org/wiki/Modulacja_szerokości_impulsów
[8] https://pl.wikipedia.org/wiki/Mostek_H
[12] http://www.ti.com/lit/ds/symlink/l293d.pdf
[15] http://home.agh.edu.pl/~bartus/index.php?action=efekty&subaction=arduino&item=37
[19]https://pl.electronics-council.com/build-your-own-metal-detector-with-an-arduino-16782
[20] https://pl.wikipedia.org/wiki/Bateria_alkaliczna
[21] https://pl.wikipedia.org/wiki/Akumulator_elektryczny
[22]https://botland.com.pl/pl/moduly-wifi/4527-modul-wifi-esp-01-esp8266-black-3-gpio-1mb-pcb-antena.html
[25] https://botland.com.pl/pl/arduino-moduly-glowne/1060-arduino-uno-rev3-a000066-8058333490090.html
[27] http://www.cs.put.poznan.pl/jjozefowska/wyklady/wspia/Informatyka4.pdf

5.2 Obrazów

[1]https://botland.com.pl/46452-thickbox_default/chassis-rectangle-2wd-2-kolowe-podwozie-robota-z-napedem.jpg
[3] https://forbot.pl/blog/wp-content/uploads/2019/03/silniki_dc_kompendium_forbot_04-1.png
[6] https://es.mathworks.com/help/slcontrol/ug/pwm_plot.png
[7] https://upload.wikimedia.org/wikipedia/commons/d/d4/H_bridge.svg
[9] https://www.obliczeniowo.com.pl/rysunki/Elektrotechnika_Podzespoly_Mostek_H.svg
[10] http://www.ti.com/lit/ds/symlink/l293d.pdf
[11] https://botland.com.pl/pl/sterowniki-silnikow-dc/176-l293d-dwukanalowy-sterownik-silnikow-36v06a.html
[13] http://rab.ict.pwr.wroc.pl/~arent/rr/mpr/pliki/czujniki/enkoder2.jpg
[14] http://rab.ict.pwr.wroc.pl/~arent/rr/mpr/pliki/czujniki/enkoder2.jpg
[16] http://home.agh.edu.pl/bartus/images/arduino/37/LM393-pinout.png
[17]https://i.electronics-council.com/electronics-img/projects/167/build-your-own-metal-detector-with-an-arduino.jpg
[18]https://i.electronics-council.com/electronics-img/projects/167/build-your-own-metal-detector-with-an-arduino-2.jpg
[23]https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf
[24]https://2.bp.blogspot.com/-6WM5M0L-T08/WqF3lSosm5I/AAAAAAAAB1k/bNt5qbAIT5obfe afw9ABJ4rRME5MYszHwCLcBGAs/s1600/124_aplikacja_ESP-01S.jpg
[26] http://akademia.nettigo.pl/arduino_uno/arduino_uno_pinouts.png
[28]https://sc01.alicdn.com/kf/HTB1iJOWG1OSBuNjy0Fdq6zDnVXas/Atmega328p-au-Atmega328p-Tqfp-Smd-Microcontroller-Au.jpg_300x300.jpg
[29]https://forbot.pl/blog/wp-content/uploads/2016/08/arduino_uno_schemat.png