Szkoła Główna Gospodarstwa Wiejskiego w Warszawie
Wydział Zastosowań Informatyki i Matematyki

Piotr Perzyna – 161195
Kamil Kostana – 154875

Samobalansujący pojazd dwukołowy

Projekt i budowa pojazdu typu Segway ®

Self-balancing two-wheeled vehicle
Design and construction vehicles type as Segway ®

Praca inżynierska
na kierunku – Informatyka

Praca wykonana pod kierunkiem
dr hab. inż. Arkadiusz Orłowski
Katedra Informatyki

Warszawa, 2016

Oświadczenie promotora pracy

Oświadczam, że niniejsza praca została przygotowana pod moim kierunkiem i stwierdzam, że spełnia ona warunki do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data ..... Podpis promotora pracy .....

Oświadczenie autorów pracy

Świadomi odpowiedzialności prawnej oświadczamy, że niniejsza praca dyplomowa została napisana przez nas samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczamy również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczamy ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data ..... Podpis autora pracy .....

Data ..... Podpis autora pracy .....

STRESZCZENIE

Praca została zainspirowana seryjnie produkowanym pojazdem – Segway Human Transporter ®. Głównym jej celem było skonstruowanie własnego robota samobalansującego, który umożliwiałyby transport ludzi. Na początku naszej pracy dyplomowej zogniskowaliśmy założenia jakie ma spełniać urządzenie. Drugą kluczową kwestią, którą opisaliśmy, była fizyka poruszania się pojazdu oparta o model odwróconego wahadła. Teoria ta była podstawą przy opracowaniu oprogramowania dla układów elektronicznych. W części projektowej skupiliśmy się nad przygotowaniem konstrukcji i wyborze adekwatnych komponentów składowych spełniających nasze wymogi. Na końcu pracy poruszyliśmy również temat potencjalnych ograniczeń i zagrożeń.

Słowa kluczowe – samobalansujący pojazd, robot, Segway ®

SUMMARY

The project was inspired by mass-produced vehicle – Segway Human Transporter ®. The main object was to create a self-balancing robot that allows to transport people. At the beginning of our thesis, we presented our assumptions of device. General establishment was the use of the most accessible and universal components. The second most important issue that we de-scribed was physics of the vehicle, based on the inverted pendulum model. The physics was on the bases of software development and created electronic circuits. In the part of construction, we focused on the preparation of structure and selection of adequate components comprising complying with our assumptions. In the summary, we include potential limitations and safety.

Keywords – self-balancing vehicle, robot, Segway ®

SPIS TREŚCI

WSTĘP

Na początku XXI wieku rosnące zainteresowanie pojazdami elektrycznymi przyczyniło się do rozwoju robotów samobalansujących. W 2002 roku zostało zaprezentowane przez Deana Kamena pierwsze tego typu urządzenie, które nazwano Segway Human Transporter ® [11]. Pojazdy tego typu stają się coraz bardziej popularne ze względu na ich mobilność oraz rozmiary. Robot ten porusza się i utrzymuje stabilizację na zasadzie odwróconego wahadła. Wahadło to, jest nieliniowym dynamicznie niestabilnym układem. Wymagana jest jego aktywna koordynacja, aby pozostawało w pozycji pionowej. Dzięki zastosowaniu specjalnych systemów sterowania może się on z łatwością poruszać w każdym kierunku. Dodatkowo wykorzystanie dwóch kół umożliwiło zawracanie w miejscu, lecz nie utrudniło poruszania się po każdej nawierzchni. Na rynku dostępne są również inne rozwiązania wykorzystujące samobalans, m.in: samobalansujący wózek inwalidzki, samobalansująca dwu-kołowa deska oraz Airwheel, czyli pojazd jednokołowy.

Celem pracy było zbudowanie dwukołowego pojazdu samobalansującego umożliwiającego transport ludzi. W pierwszym rozdziale zostały zawarte informacje na temat podstawowego założenia, ograniczeń i bezpieczeństwa poruszania się pojazdem.

Drugi rozdział opisuje prawa fizyki, dzięki którym możliwe jest poruszanie się robota oraz zostały wyjaśnione podstawowe informacje dotyczące jazdy.

Następnie skupiliśmy się na przedstawieniu zastosowanych materiałów mechanicznych takich jak korpus pojazdu, zestaw kołowy oraz przeniesienie napędu. Zawarliśmy również informacje o wykorzystanych elementach elektronicznych i schemacie ich połączenia.

Czwarty rozdział opisuje oprogramowanie jakie wykorzystaliśmy do implementacji logiki sterowania. Poruszyliśmy również problem filtracji danych pochodzących z czujników, przedstawiając trzy podstawowe filtry, które są głównie stosowane przy tego typu projektach.

Końcowa część to podsumowanie wyników pracy stworzonego robota samobalansującego. Zostały również zawarte ważne spostrzeżenia i uwagi, które dotyczą budowy oraz użytkowania pojazdu.

Ze względu na złożoność projektu, praca dyplomowa posiada dwóch autorów. Niesie to ze sobą zalety takie jak: większa efektywność szukania rozwiązań oraz podział pracy między autorów według wiedzy oraz umiejętności, a zatem mniejsze prawdopodobieństwo pomyłki. Praca dyplomowa była wykonana przez obu autorów z takim samym zaangażowaniem oraz wkładem pracy. Można jednak wyszczególnić części, na których każdy z nich był bardziej skoncentrowany oraz włożył w nią więcej pracy, zarówno ze względu swoją większą wiedzę, oraz doświadczenie: do Kamila Kostany należał opis części mechanicznej i oprogramowania, a do Piotra Perzyny – teoria sterowania oraz część elektroniczna.

1 MODEL POJAZDU SAMOBALANSUJĄCEGO

1.1 Założenie

Przedstawione we wstępie urządzenie Deana Kamena jest stosunkowo kosztowne. Najtańszy model kosztuje około 25 000 zł, dlatego priorytetowym założeniem całej pracy było skonstruowanie robota samobalansującego z ogólnie dostępnych oraz niedrogich komponentów z zachowaniem wszystkich jego właściwości jezdnych.

1.2 Ograniczenia

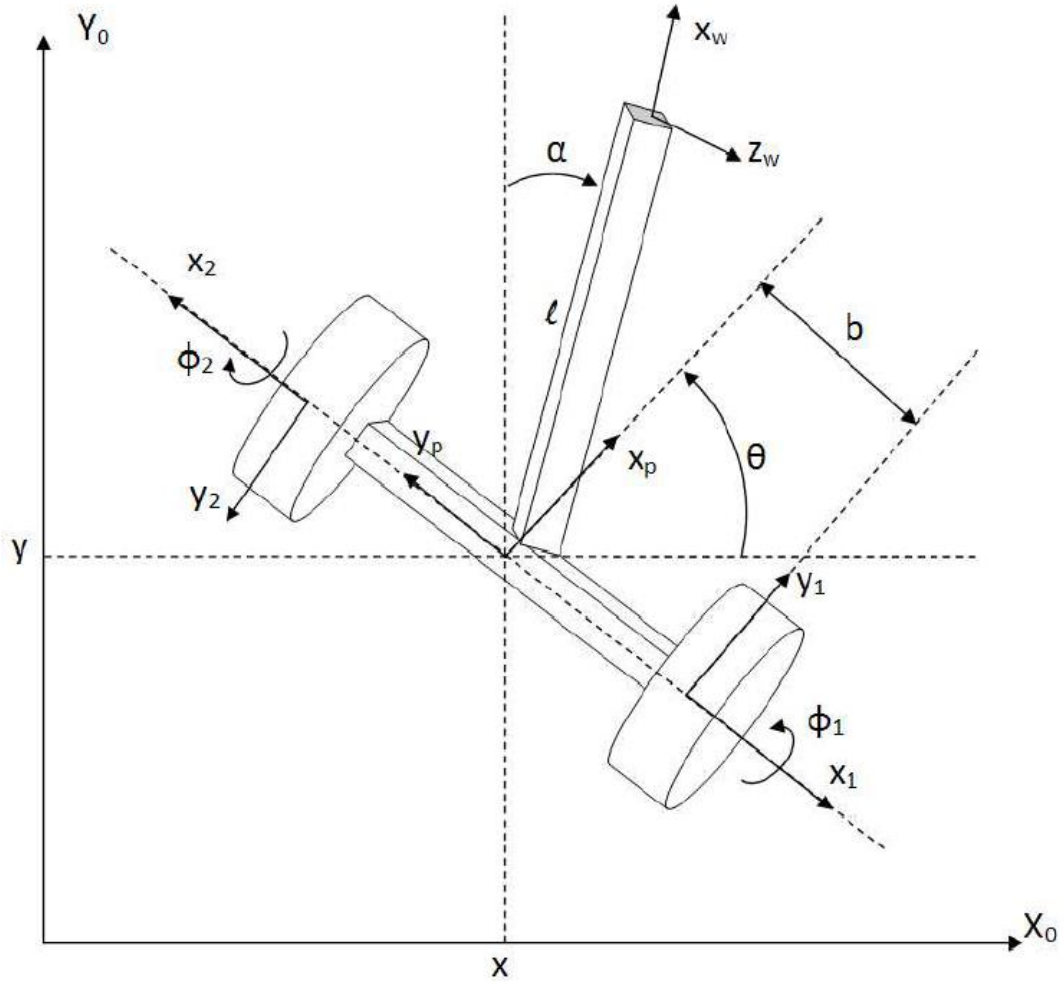
Głównym problemem, z jakim się zderzyliśmy podczas projektowania pojazdu była dostępność gotowych rozwiązań. Ze względu na istniejące komercyjne urządzenie, które jest chronione prawem patentowym, na rynku nie istnieją części, z których łatwo można zbudować pojazd [8]. Dlatego do naszego projektu wykorzystaliśmy wiele części z różnych dziedzin życia (np. żeglarstwo, rolnictwo, gospodarstwo domowe, modelarstwo, przemysł) zmieniając ich pierwotne zastosowanie i dostosowując do naszych potrzeb.

Rysunek 1 - Segway Human Transporter ®

Źródło: www.segway.com

2 ODWRÓCONE WAHADŁO

Odwrócone wahadło to niestabilny układ o dwóch stopniach swobody [1]. Z tego powodu, aby pozostało w stanie równowagi niezbędna jest jego aktywna regulacja. Fizycznie, to urządzenie zaliczane jest do klasy robotów balansujących, którego przeznaczeniem jest zapewnienie stabilizacji pozycji pionowej zamocowanych elementów, których środek ciężkości jest powyżej osi obrotu. Nasz robot może być rozważany, jako duże odwrócone wahadło. Jedyną konstrukcyjną różnicą względem klasycznego odwróconego wahadła jest brak ściśle zdefiniowanej platformy jezdnej. W naszej konstrukcji, bezpośrednio do osi obrotu są przymocowane dwa koła. Taki układ umożliwia ciągłe przekazywanie odpowiedniego momentu obrotowego na koła, dążąc tym samym do uzyskania stanu równowagi.



Rysunek 2 - model odwróconego wahadła

Źródło: Jan Kędzierski - Robot balansujący - model i sterowanie

3 BUDOWA TRANSPORTERA KOŁOWEGO

3.1 Konstrukcja mechaniczna

Rysunek 3 - dwukołowy pojazd samobalansujący

Źródło: opracowanie własne

3.1.1 Korpus pojazdu

3.1.1.1 Platforma

Podstawowym elementem korpusu robota samobalansującego jest metalowa platforma o grubości 3 mm i wymiarach 46x36 cm. Na platformie zostały wykonane otwory montażowe, które umożliwiają zamocowanie pozostałych elementów konstrukcji, takich jak: moduły elektroniczne, silniki, akumulatory, kierownica. Drugim ważnym elementem, który został przymocowany do platformy, jest stalowy, chromowany pręt, na końcach, którego zostały zainstalowane koła jezdne, wraz z łożyskami i kołami zębatymi do przeniesienia napędu.

Podczas projektowania, ważnym elementem było zaplanowanie rozmieszczenia wszystkich elementów robota [7]. Powstały dwa plany zagospodarowania miejsca na platformie. Pierwotny plan zakładał umieszczenie mocowania kierownicy na górze platformy, co w dużej mierze ograniczałoby miejsce dla użytkownika. Jednak ostatecznie wszystkie elementy takie jak czujniki, akumulatory oraz kontrolery zostały umieszczone na spodzie platformy. Dzięki temu, użytkownik robota ma dostępną całą górną przestrzeń platformy. W celu zapewnienia bezpieczeństwa prowadzącemu, w bocznych częściach platformy zamontowano gumowo-metalowe osłony kół, które zostały opisane w części pod tytułem Wykończenie.

Rysunek 4 - dolna część platformy

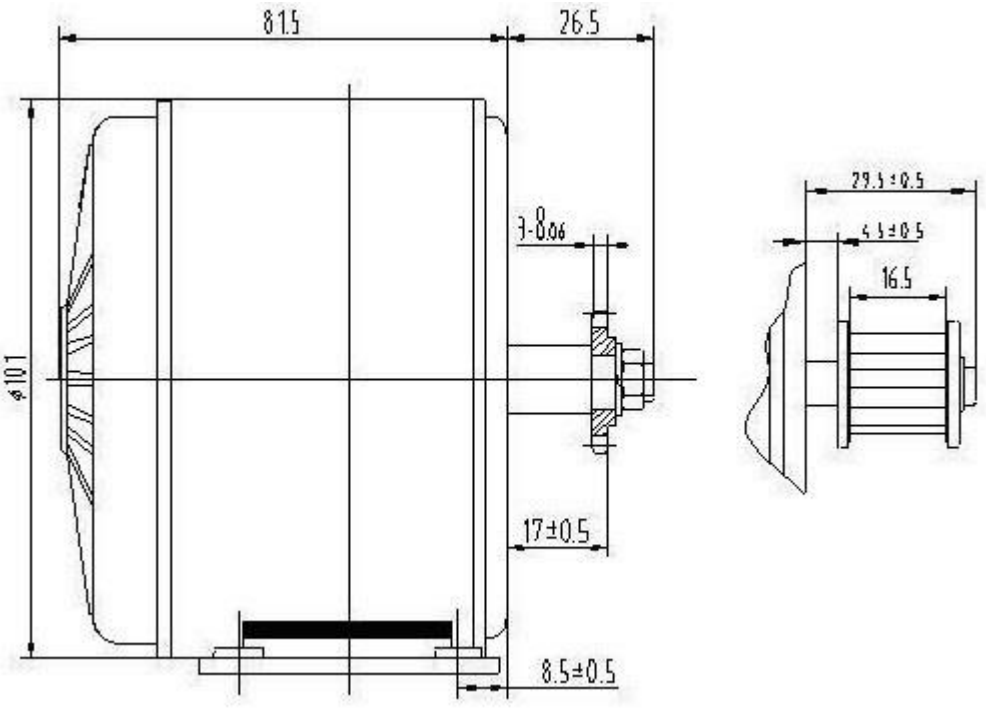
Źródło: opracowanie własne

3.1.1.2 Kierownica

W przedniej części platformy została przymocowana kierownica, która umożliwia użytkownikowi kontrolę skręcania. Wykonana została z rury aluminiowej, częściowo gumowanej o średnicy 25 mm i przymocowana do platformy przy użyciu dwóch łożysk kulkowych w obudowach, aby zapewnić jej stabilność pionową oraz ruchomość poziomą. Za stabilizację kierownicy odpowiadają również sprężyny pionujące, które utrzymują kierownice w osi. Rura użyta do budowy kierownicy została specjalnie wyprofilowana, aby przesunąć środek ciężkości bliżej części centralnej platformy. Dla wygody oraz bezpieczeństwa zamontowano rękojeść, stacyjkę na kluczyk oraz wyłącznik bezpieczeństwa. W razie problemów kierujący wciskając przycisk ma możliwość odłączenia zasilania całego układu elektrycznego. W dolnej części kierownicy znajduje się jeden z modułów żyroskopu z akcelerometrem do wyznaczania aktualnego wychylenia.

3.1.2 Napęd pojazdu

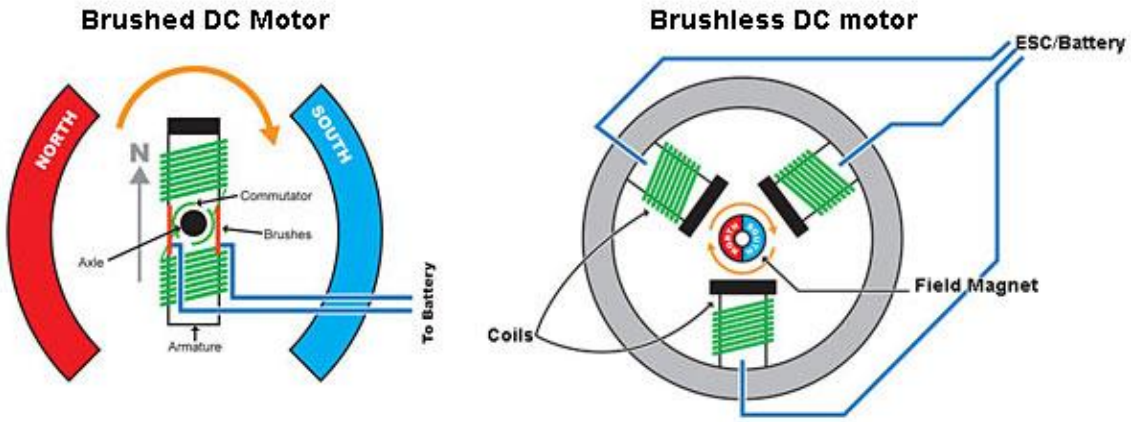
3.1.2.1 Silniki



Rysunek 5 - schemat silnika

Źródło: www.magmapolska.pl

Jednym z kluczowych komponentów pojazdu są silniki. W modelarstwie amatorskim przy tego typu projektach stosowane są najczęściej dwa typy silników: silniki szczotkowe oraz bezszczotkowe. Teoria działania obydwu wariantów została przedstawiona na rysunku 6. Budowa silnika szczotkowego charakteryzuje się mechanicznym połączeniem pomiędzy komutatorem i szczotkami węglowymi o niskiej ścieralności. To właśnie dzięki szczotkom zostaje doprowadzony prąd i w wyniku oddziaływania płynącego prądu oraz pola magnetycznego wytwarzanego przez stojan, wirnik silnika zostaje wprawiony w ciągły ruch. Zaletą takiego rozwiązania jest to, iż nie wymagają specjalnego czujnika położenia wirnika oraz, że mogą być zasilone napięciem stałym DC (ang. direct current) np. z akumulatorów, przez co często stosuje się je w urządzeniach bezprzewodowych. W przypadku silników bezszczotkowych BLDC (ang. BrushLess Direct-Current motor) zamiast szczotek zastosowano elektrycznie sterowany komutator, w którym cewki są nieruchome, a magnesy znajdują się na wirniku. Główną zaletą takiej konstrukcji jest dużo wyższa trwałość i niezawodność wynikająca z wykluczenia szczotek, które były najszybciej zużywającym się elementem i główną przyczyną awarii. Drugim plusem jest możliwość sterowania prędkością obrotową, prawie niezależnie od momentu obrotowego. Jednak silniki te mają jedną kluczową dla nas wadę, która zgodnie z założeniami pracy dyskwalifikuje je: są one kilkukrotnie droższe od silników szczotkowych i wymagają specjalnych sterowników. Dlatego uwzględniając stosunek jakości do ceny zdecydowaliśmy się na mniej kosztowne rozwiązanie. Do napędzania pojazdu zostały użyte dwa klasyczne silniki szczotkowe o mocy 250 W. Przy ich wyborze kierowaliśmy się dokumentacją dostarczoną przez dystrybutora. Ostatecznie ten typ silników okazał się odpowiednim wyborem, silniki dysponują kilkunastoprocentowym zapasem mocy.



Rysunek 6 - porównanie silnika szczotkowego i bezszczotkowego

Źródło: www.thinkrc.com

Model	1016 - 250W24V
Waga	1.92 (kg)
Bez obciążenia	
Napięcie	24 (V)
Prąd	0.6 - 1.4 (A)
Obroty	3250 ± 5% (rpm)
Z obciążeniem	
Moc	250 (W)
Prąd	≤ 13.5 (A)
Obroty	2600 – 2850 (rpm)
Moment	0.84 - 0.92 (N.m)
Efektywność	n ≥ 78 (%)

Tabela 1 - specyfikacja silnika 1016 – 250W24V

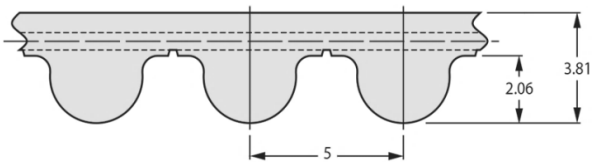
Źródło: www.magmapolska.pl

3.1.3 Przeniesienie napędu

Kolejnym bardzo ważnym zagadnieniem jest układ przeniesienia napędu, służący do napędzenia pojazdu, poprzez dostarczenie energii mechanicznej z silnika do kół. W skład zaprojektowanego przez nas układu wchodzą: reduktory, czyli koła zębate, które mają na celu zwiększenie momentu obrotowego kosztem prędkości obrotowej oraz pasy zębate, które przenoszą moment obrotowy z silników na koło.

W naszym projekcie w początkowej fazie testowaliśmy dwa różne układy przeniesienia napędu. Pierwszym sprawdzonym przez nas systemem była konstrukcja składająca się z metalowego łańcucha oraz zębatek. Ten mechanizm okazał się bardzo nieefektywny, ze względu na małą elastyczność. Powodował wytwarzanie dużych wibracji przy wprawianiu pojazdu w ruch oraz przy pokonywaniu nierówności. Dodatkowo, łańcuch wymagał smarowania i generował spory hałas podczas pracy, który znacząco obniżał komfort jazdy. Układ ten, miał jednak zasadnicze dla nas zalety, był łatwo dostępny i nie wymagał dużego nakładu finansowego.

Drugim weryfikowanym przez nas układem były pasy i koła zębate. Pasy zębate wykorzystuje się tam, gdzie potrzeba dokładnej synchronizacji i wysokiej wydajności układu. Taśmy te bardzo dobrze radzą sobie z nagłymi zmianami kierunku i prędkości, utrzymując przy tym odpowiednią pozycję na przenośniku. Miało to dla nas kluczowe znaczenie. Nie wymagają one również smarowania, jak ma to miejsce w wypadku napędu łańcuchowego oraz ich zastosowanie zmniejsza poziom wytwarzanego hałasu, przyczyniając się do spokojnej, przyjemnej jazdy. Pasy przeważnie są wykonane z neoprenu lub poliuretanu. Dzięki zastosowaniu elastycznego materiału, koła zostają wprawiane płynnie w ruch unikając przy tym szarpnięć. Nie wymagają one mocnego naciągu, dzięki czemu łożyska zużywają się dużo wolniej, co przekłada się na mniejsze koszty serwisowania. W naszym projekcie wykorzystaliśmy dwa pasy zębate 5M-600-15, wykonane z neoprenu. Szczegółowa specyfikacja została przedstawiona w tabeli 2.



Rysunek 7 - schemat paska zębatego

Źródło: www.ondrives.com

Dane techniczne paska zębatego					
Producent	Typ paska	Szerokość	Liczba zębów	Długość robocza	Podziałka
Optibelt	wzmocniony	15mm	120	600 mm	HTD5M

Tabela 2 - specyfikacja paska zębatego

Źródło: opracowanie własne

Kolejnym elementem układu są dwa koła zębate użyte jako reduktor i wykonane z poliamidu, który charakteryzuje się dużą twardością, bardzo dobrą wytrzymałością mechaniczną, niską rozszerzalnością cieplna oraz wysoką zdolnością do tłumienia drgań.

Na rynku jest mało dostępnych rozmiarów kół zębatych, ponieważ są one najczęściej stosowane w specjalistycznych rozwiązaniach. Na potrzeby naszego projektu musieliśmy skorzystać z usługi profesjonalnej firmy oferującej możliwość przygotowania koła za pomocą obrabiarki CNC (ang. Computerized Numerical Control) z wcześniej przygotowanego projektu 3D. Pierwsze koło, które posiada 13 zębów zostało zamocowane za pomocą połącznia klinowego na wale silnika. Drugie zaś o 112 zębach zostało przyłączone bezpośrednio do koła za pomocą śrub na dystansach. Uzyskano dzięki temu przełożenie stałe w stosunku 112:13.

3.1.4 Zestaw kołowy

Rysunek 8 - koło pneumatyczne

Źródło: www.dariuszkalisz.pl

Na rynku dostępnych jest wiele rodzajów kół. Różnią się one rodzajem ogumienia, materiałem wykonania felgi oraz typem łożyskowania. Główną przeszkodą z jaką miały stawić się koła to pokonywanie zmiennego podłoża podczas jazdy. Wybrany przez nas zestaw: opona dętkowa oraz felga stalowa z otworem na zamocowanie własnej piasty wydał się, dla potrzeb naszego projektu, najbardziej optymalny. Koła o średnicy 45cm pozwalają bezproblemowo poruszać się po większości podłoży (asfalt, trawa, piasek). Opona dętkowa dobrze pochłania nierówności na drodze, dzięki czemu uzyskaliśmy stabilną oraz komfortową jazdę. Dokładna specyfikacje kół została przedstawiona w tabeli 3.

Parametry koła					
Średnica	Szerokość	Otwór na piastę	Rodzaj opony	Grubość felgi	Nośność
45 cm	100 mm	75 mm	4-płócienna (4PR)	4mm (2 x 2 mm)	300 kg

Tabela 3 - specyfikacja koła

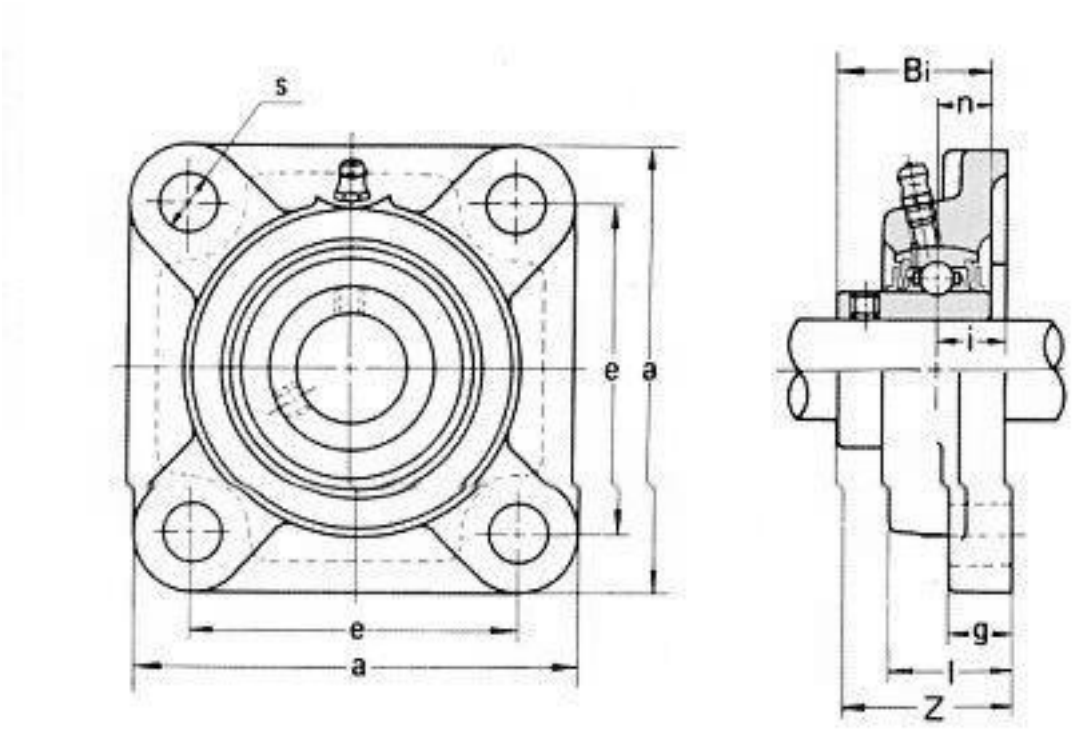
Źródło: opracowanie własne

Do zestawu kołowego oprócz kół zaliczyliśmy łożyska samonastawne TechRol UCF, które pełnią rolę piast i które zostały zamontowane bezpośrednio do felg. Samonastawne zespoły łożyskowe znajdują zastosowanie w wielu dziedzinach, takich jak: technika napędu, rolnictwo, budownictwo, transport. Łatwość montażu oraz niewielki koszt sprawia, że cieszą się one wielką popularnością. Dzięki wahliwości, zespół łożyskowy kompensuje błędy niewspółosiowości, czyli niedoskonałości w montażu. Dodatkowym atutem łożyska jest własny zapas smaru, który można uzupełniać dzięki specjalnemu otworowi. Kolejną zaletą jest duży luz promieniowy, który bardzo dobrze kompensuje ugięcia wału. Parametry łożysk zostały zawarte odpowiednio na rysunku 9 i tabeli 4.

d	**A**	**E**	**C**	**G**	**L**	**S**	**Z**	**B**	**F**	**gwint M**	**Masa(kg)**
40	130	102	21	16	36	16	51,2	49,2	19	M14	1,84

Tabela 4 - specyfikacja łożyska

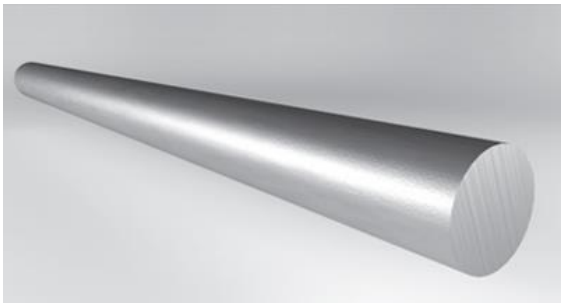
Źródło: www.akcesoria.cnc.info.pl



Rysunek 9 - schemat łożyska

Źródło: www.akcesoria.cnc.info.pl

Zestaw kołowy został połączony z platformą za pośrednictwem wałka stalowego Φ25, który jest zamocowany przy użyciu trzech zacisków kabłąkowych 26mm. W pierwszej fazie budowy zastosowaliśmy wałek o średnicy Φ15, jednak podczas testów okazał się niewystarczająco twardy. Zmusiło nas to do wymiany wałka na grubszy oraz wymiany łożysk i zacisków z odpowiednio większymi otworami mocowania.



Rysunek 11 - pręt chromowany 25mm

Źródło: www.cromostal.pl

Rysunek 10 - zacisk kabłąkowy

Źródło: www.linynametry.pl

3.1.5 Wykończenie

Pracą zwieńczającą konstrukcję mechaniczną było przymocowanie elementów wykończeniowych. Dzięki nim, nasz robot przybrał elegancką postać. Od zewnętrznej strony kół zostały przymocowane kołpaki, zasłaniające tym samym elementy takie jak felgi i łożyska. Kolejnym elementem wykończenia jest gumowa antypoślizgowa mata, którą przymocowaliśmy na górze platformy. Dzięki temu, zostały zamaskowane wszystkie śruby, które zostały użyte do zainstalowania elementów na spodniej części platformy.

Rysunek 12 - plastikowa osłona felgi i łożyska

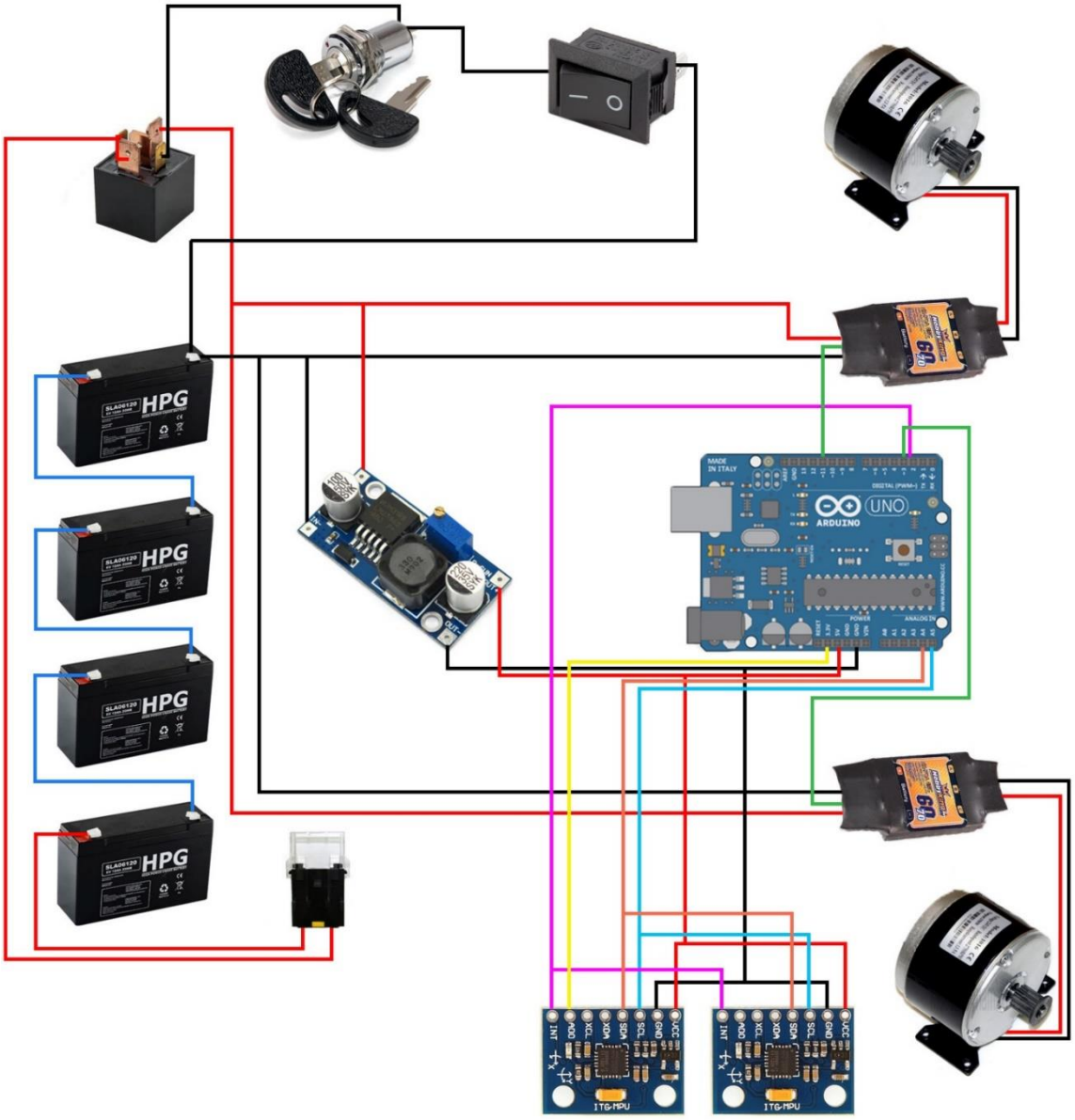
Źródło: opracowanie własne

Rysunek 13 - podłoga pojazdu

Źródło: opracowanie własne

3.2 Konstrukcja elektroniczna

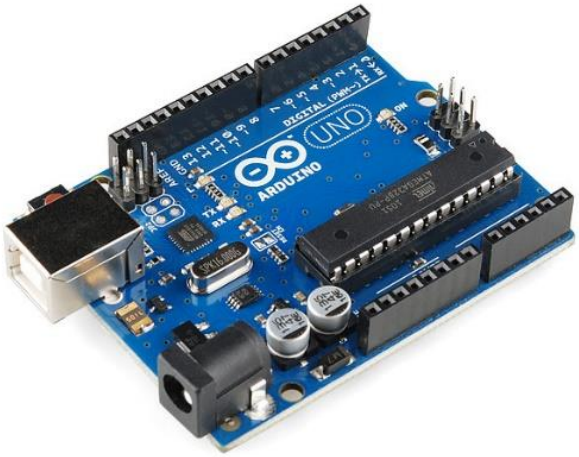
3.2.1 Schemat połączeń



Rysunek 14 - schemat połączeń elektrycznych

Źródło: opracowanie własne

3.2.2 Arduino UNO R3



Rysunek 15 - Arduino Uno R3

Źródło: www.arduino.cc

Arduino powstało w 2005 roku, jako ustandaryzowany programowalny gotowy kontroler umożliwiający odczyt informacji z różnych źródeł oraz sterowanie wielorakimi układami [14]. Celem projektu było dostarczenie narzędzi dla studentów i hobbystów, które byłyby ogólnodostępne, niewymagające dużych nakładów finansowych, łatwe w obsłudze oraz elastyczne. Arduino jest również idealną alternatywą dla osób, które nie mogą skorzystać z bardziej zaawansowanych kontrolerów, wymagających wielowymiarowych narzędzi. Popularność projektu i jego ustandaryzowanie przyczyniło się do powstania wielu dodatkowych rozszerzeń (ang. shield). Na rynku istnieją moduły pozwalające rozbudować podstawowy układ o komunikację przez sieć bezprzewodową Wifi, Bluetooth oraz sieć lokalną LAN. Pozostałe często spotykane rozszerzenia to: lokalizacja GPS, żyroskop, akcelerometr oraz ekran LCD.

Nazwa:	Arduino Uno R3	**Napięcie zasilania:**	od 5 V do 12 V
Mikrokontroler:	ATmega328	**Pamięć SRAM:**	2 kB
		Pamięć Flash:	32 kB
		Pamięć EEPROM	1 kB
		Max. częstotliwość zegara:	16 MHz
Porty I/O:	14	**Wyjścia PWM:**	6
Gniazda:	USB A	**Interfejsy szeregowe:**	UART
	DC		SPI
Wejścia analogowe:	6		I2C

Tabela 5 - specyfikacja Arduino UNO R3

Źródło: opracowanie własne

W zależności od zastosowanego mikrokontrolera, rozróżnia się wiele dystrybucji. W naszym projekcie wybraliśmy najnowszy model Arduino Uno R3. Kontroler został oparty na mikrokontrolerze AVR ATmega328 w wymiennej obudowie. Szczegóły specyfikacji zawarliśmy w tabeli 5. Językiem programowania w Arduino, dzięki bibliotece „Wiring" wchodzącej w skład całego środowiska Arduino IDE (ang. Arduino Integrated Development Environment), jest C/C++. Środowisko zostało napisane w języku Java, co dało możliwość łatwego przenoszenia kodu pomiędzy systemami operacyjnymi.

Rysunek 16 - schemat ideowy połączeń

Źródło: opracowanie własne

Kontroler ma kilka prostych zadań w naszym projekcie. Pierwszym celem układu jest pobieranie informacji od czujników MPU-6050 o położeniu pojazdu oraz wychyleniu kierownicy. Kolejnym zadaniem jest wysyłanie odpowiednio zmodulowanego sygnału do układów wykonawczych HK-SS70A, aby skorygować położenie pojazdu na podstawie logiki, która została omówiona w rozdziale czwartym. Dodatkowo Arduino zostało tak zaprogramowane, aby wykrywać swobodny upadek. Czujniki dzięki wbudowanej obsłudze przerwań wysyłają sygnał odłączenia napędu, gdy kierujący zderzy się z przeszkodą lub upadnie np. na wskutek zbyt dużego wychylenia pojazdu.

3.2.3 Czujniki

Do stabilnej i poprawnej pracy algorytmy sterujące wymagają informacji o prędkości kątowej oraz położeniu względem podłoża. Wymienione dane zostają uzyskane z czujnika, który został opisany w tym rozdziale.

3.2.3.1 MPU-6050

Rysunek 17 - MPU-6050

Źródło: www.botland.com.pl

Głównym elementem pomiarowym w naszym projekcie są dwa czujniki do pomiaru przyspieszeń oraz prędkości kątowej w przestrzeni trójwymiarowej – MPU-6050. Komunikują się one poprzez popularną magistralę I^2C (TWI ang. Two Wire Interface). Układ MPU posiada sprzętową jednostkę DMP (ang. Digital Motion Processor), która pozwala na przeliczanie danych na położenie względem Ziemi.

Nazwa:	MPU-6050	**Osie pomiaru:**	3 osie (X, Y, Z)
Napięcie zasilania:	2 V, 3.3 V, 5 V	**Rozdzielczość na oś:**	16-bitów
Pobór prądu:	~ 5 mA	**Komunikacja:**	I^2C (TWI) - 400 kHz
Zakresy pomiarowe:	Akcelerometr	±2g, ±4g, ±8g, ±16g	
	Żyroskop	±250, ±500, ±1000, ±2000 °/s	
Masa:	0,9 g	**Wymiary płytki:**	20,3 x 12,7 x 2,5 mm

Tabela 6 - specyfikacja MPU-6050

Źródło: Opracowanie własne

W robocie samobalansującym zostały użyte dwa moduły MPU-6050. Pierwszy z nich został zamieszczony w najbliższym możliwym miejscu od centralnego środka platformy, tak, aby był jak najbliżej dwóch osi pojazdu. Dane pochodzące z tego czujnika odpowiadają za informowanie sterownika głównego w naszym przypadku jest to Arduino Uno o aktualnym położeniu względem podłoża, czyli o bieżącym przechyle. Drugi moduł został przymocowany do kierownicy i dostarcza dane o obecnym wychyleniu drążka kierownicy. Oba czujniki wysyłają informacje na temat prędkości kątowej. Niestety, elektroniczne przetwarzanie danych dostarcza również błędnych odczytów, spowodowanych dużą wrażliwością na wibracje sensorów akcelerometrów. Dodatkowo, w żyroskopie na skutek zmian temperatury, szumów oraz cyfrowej reprezentacji sygnału z czujników, pojawia się pewna składowa stała, którą po analizie możemy nazwać dryfem [12]. Fakt ten jest stosunkowo trudny w eliminacji, stąd przyjęliśmy jego obecność i w rozdziale czwartym opisujemy programowe rozwiązania filtracji danych.

3.2.3.2 Czujniki dodatkowe

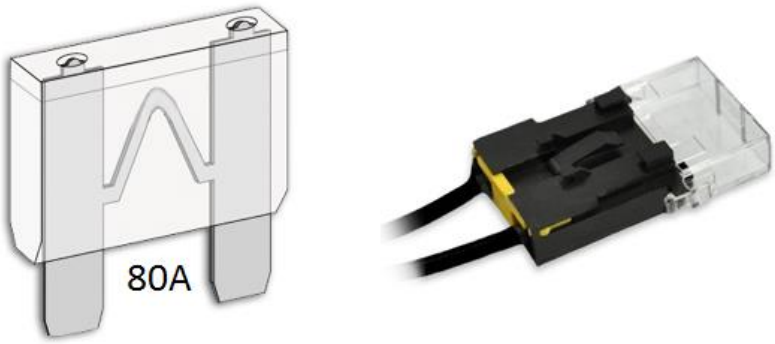
Rysunek 18 - woltomierz

Źródło: opracowanie własne

Do obwodu elektrycznego został włączony równolegle cyfrowy woltomierz z wyświetlaczem LED (7 segmentowym 0.56") oparty na układzie ICL7106. Woltomierz mierzy napięcie prądu stałego w zakresie od 2,5 V do 30 V oraz nie wymaga osobnego źródła zasilania. Przyrząd ten w naszym projekcie sprawdza stan rozładowania akumulatorów, czyli znamieniowe napięcie akumulatorów.

3.2.4 Bezpiecznik

W obwód elektryczny pojazdu został zamontowany bezpiecznik prądowy wraz z obudową o wartości nominalnej 80A. Zabezpiecza on przez zbyt dużym natężeniem prądu. Gdy natężenie prądu przekroczy ustaloną wartość, bezpiecznik przepali się, co spowoduje przerwanie obwodu i prąd nie popłynie. Jest to bardzo istotny element każdej instalacji elektrycznej, ponieważ w skrajnych przypadkach, gdy prąd będzie zbyt duży, może dojść do przepięcia. Przepalonego bezpiecznika nie naprawia się, podlega on wymianie na nowy.

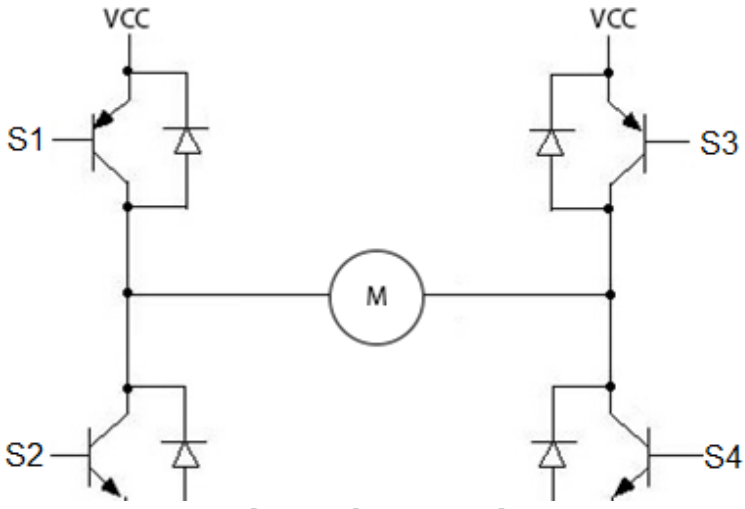


Rysunek 19 - bezpiecznik nadprądowy wraz z obudową

Źródło: www.e-connectors.pl

3.2.5 Sterownik napędu

Sterowaniem napędu, czyli w naszym wypadku silnikami prądu stałego (DC) zajmują się dwa układy nazwane w elektronice mostkami H (ang. H-bridge) [9]. Nazwa pochodzi od charakterystycznego wyglądu ich schematu elektrycznego (rysunek 20). Prosta wersja składa się z czterech tranzystorów MOSFET lub innych elementów półprzewodnikowych. Układy mają za zadanie zmianę polaryzacji napięcia, jakie pojawia się na silniku poprzez „odwracanie" biegunów zasilania. W przypadku, kiedy styki S1 i S4 są zamknięte, a S2 i S3 otwarte, silnik będzie otrzymywał napięcie i zacznie pracować. Analogicznie zamykając styki S2 i S3, a otwierając S1 i S4, silnik znów otrzyma napięcie, ale zmieni się kierunek przepływu prądu, co spowoduje zmianę kierunku obrotu wirnika. Możliwa też jest regulacja prędkości obrotowej silnika, poprzez odpowiednie i stosunkowo szybkie otwieranie i zamykanie poszczególnych styków, które może być realizowane przez modulacje szerokości impulsu sygnału PWM (ang. Pulse-Width Modulation). W naszym sterowniku, sygnał PWM wykorzystywany jest do przekazywania informacji sterownikom i ma rozdzielczość 0-190. Jeśli na silniki zostanie przesłany zmodulowany sygnał PWM o wartości 0 to silniki będą poruszać się z maksymalną prędkością do tyłu. Brak pracy napędu osiągnie się przy wysłaniu sygnały o wartości 95, a maksymalna prędkość do przodu uzyska się przy 190.



Rysunek 20 - schemat mostka H

Źródło: www.hobbyrobotyka.pl

Najczęściej spotykanymi w amatorskich konstrukcjach są zintegrowane mostki z gotowymi układami scalonymi. Do grupy najbardziej popularnych układów należą L298N i L293D. Charakteryzują się one skutecznością zastosowań. Niestety, w naszym projekcie zbyt duża wartość natężenia prądu chwilowego nie pozwalała wykorzystać tych rozwiązań. Dlatego zgodnie ze specyfikacją silników, uwzględniając możliwość wystąpienia chwilowych przepięć, postanowiliśmy wykorzystać rozwiązanie ESC (ang. electronic speed control), dostępne na rynku azjatyckim - HK-SS70A (rysunek 21). Jest to rozwiązanie przeznaczone do silników bezszczotkowych, jednak po odpowiednim przeprogramowaniu udało się je zaimplementować do naszych silników.

Nazwa:	HK-SS70A	**Max. stałe obciążenie:**	60 A
Waga:	63g	**Max. chwilowe obciążenie:**	70 A
PWM:	8 / 16 K	**Zakres pracy napięć:**	14,4 V – 28,8V
Programowalne:	TAK	**BEC**	Nie

Tabela 7 - specyfikacja HK-SS70A

Źródło: Opracowanie własne

Rysunek 21 - HK-SS70A

Źródło: www.hobbyking.com

3.2.6 Zasilanie

3.2.6.1 Akumulatory

Rysunek 22 - akumulator HPG 060120

Źródło: www.hpgbatteries.com

Za zasilanie całego układu odpowiadają cztery bezobsługowe akumulatory VRLA (ang. Valve Regulated Lead-Acid Battery), nazywane również SLA (ang. Sealed Lead-Acid Battery). Akumulatory tego typu posiadają szereg zalet w porównaniu do akumulatorów tradycyjnych. Podstawową różnicą to zastosowany elektrolit. W akumulatorach żelowych ma on konsystencje żelu, przez co jest bardziej odporny na wstrząsy i wibracje, co daje możliwość niestandardowej pozycji pracy. W naszej konstrukcji, akumulatory są przymocowane w pozycji poziomej, aby zmniejszyć gabaryty pojazdu. Co więcej, akumulatory VRLA są odporne na głębokie rozładowanie, dlatego są zalecane do pracy w reżimie cyklicznym (rozładuj – ładuj). Kolejnymi ważnymi zaletami są małe samorozładowanie oraz lepsze odprowadzanie ciepła niż w przypadku pracy zwykłych akumulatorów. Wszystkie te cechy zdecydowały o wyborze tego typu źródła zasilania do naszego pojazdu. Akumulatory zostały połączone szeregowo, dzięki czemu uzyskaliśmy na wyjściu napięcie o wartości 24V oraz 12 Ah pojemności.

Nazwa:	HPG 060120	**Pojemność:**	12 Ah	**Długość:**	150 mm	**Waga:**	2.10 kg
Napięcie:	6 V	**Wysokość:**	94 mm	**Szerokość:**	50 mm		

Tabela 8 - specyfikacja HPG 060120

Źródło: Opracowanie własne

3.2.6.2 Ładowarka

Rysunek 23 - ładowarka akumulatorowa

Źródło: opracowanie własne

Do zastosowanego źródła zasilania – akumulatorów żelowych, zadbaliśmy o ich prawidłowy proces ładowania, aby pełniły swoją funkcję jak najdłużej. Zgodnie z zaleceniami producenta, użyliśmy specjalnej ładowarki, która przeprowadza proces ładowania metodą stałonapięciową z ograniczeniem prądowym. W pierwszej fazie akumulator ładowany jest stałym prądem wraz ze zwiększającym się napięciem. W momencie osiągnięcia napięcia 2,27 V na ogniwo, ładowanie przechodzi do drugiej fazy – stałym napięciem. Zastosowany przez nas model charakteryzuje się poniższymi parametrami.

Nazwa:	AC3017	**Napięcie ładowania:**	10-15 V DC
Zasilanie:	100-240V 50/60Hz	**Max. moc wyjściowa:**	30W
Temperatura pracy:	0 – 40 ° C	**Prąd ładowania:**	0,8+/-0,1A

Tabela 9 - specyfikacja AC3017

Źródło: Opracowanie własne

3.3 Bezpieczeństwo

Rozważając potencjalne zagrożenia, jakie może sprawić robot, wyróżniliśmy dwa główne obszary: bezpieczeństwo prowadzącego i bezpieczeństwo osób trzecich.

Na bezpieczeństwo użytkownika głównie wpływa maksymalna prędkość pojazdu. Teoretyczna, maksymalna prędkość wynikająca z parametrów dostępnych silników, przekładni i kół to w przybliżeniu 25 km/h. Jednak w fazie testów na idealnej nawierzchni prędkość jaką uznaliśmy za bezpieczną to maksymalnie 15 km/h. Drugą istotną kwestią, mającą znaczenie na zapewnienie bezpieczeństwa kierującego, było ograniczenie dostępu do wszystkich ruchomych części takich jak koła, paski zębate i silniki.

Rysunek 24 - wyłącznik bezpieczeństwa i stacyjka

Źródło: opracowanie własne

Bezpośrednio za bezpieczeństwo osób trzecich odpowiedzialne są dwa czynniki: niezawodność człowieka i niezawodność maszyny. Dlatego postanowiliśmy umieścić na kierownicy awaryjny wyłącznik, który pozwala w każdym momencie zatrzymać pojazd, przez odcięcie zasilania. Przycisk w fazie testów okazał się bardzo przydatny w przypadku błędnego działania oprogramowania sterującego. Natomiast niemożliwe jest wyeliminowanie czynnika ludzkiego, dlatego zalecamy rozważne korzystanie z pojazdu, stosując się do przepisów ruchu drogowego.

4 OPROGRAMOWANIE

4.1 Środowisko programistyczne

Arduino oprócz nazwy płytki z mikrokontrolerem to również zintegrowane środowisko programistyczne dla systemów wbudowanych. Celem projektu było stworzenie środowiska, które będzie przyjazne w użytkowaniu dla hobbystów i osób niezajmujących się programowaniem. Platforma została oparta na licencji Open Hardware. Jest to aplikacja napisana w języku Java, dzięki której możliwa jest edycja, a następnie kompilacja gotowego programu. Programowanie płyty odbywa się w prosty sposób, wystarczy wybrać posiadaną płytkę, port COM, jaki został do niej przypisany oraz nacisnąć przycisk „Upload".

Aby program został poprawnie skompilowany, wystarczy zaimplementować w kodzie dwie funkcje:

- setup() – jest wykonywana raz, po uruchomieniu programu, służy między innymi do skonfigurowania ustawień
- loop() – funkcja zapętlona, wykonywana przez cały czas, aż do odłączenia zasilania

Pozostała część wymaganego kodu jest dodawana automatycznie podczas kompilacji.

Rysunek 25 - interfejs Arduino

Źródło: opracowanie własne

4.2 Filtracja danych

Akcelerometr jest urządzeniem bardzo czułym na wszelkie siły pochodzące z otoczenia. Każdy, nawet delikatny wstrząs przekłada się na zniekształcenie, więc odczyt danych będzie wiarygodny jedynie w długim okresie czasu.

Cechą charakterystyczną żyroskopu jest dryft, czyli błąd pomiarowy kumulujący się z czasem. Jeżeli czujnik będzie pozostawał w spoczynku, mierzona wartość będzie stopniowo oddalać się od zera, więc dane można wykorzystać tylko w małych odstępach czasowych. Przetestowaliśmy kilka możliwych rozwiązań powyższego problemu, opisując je poniżej.

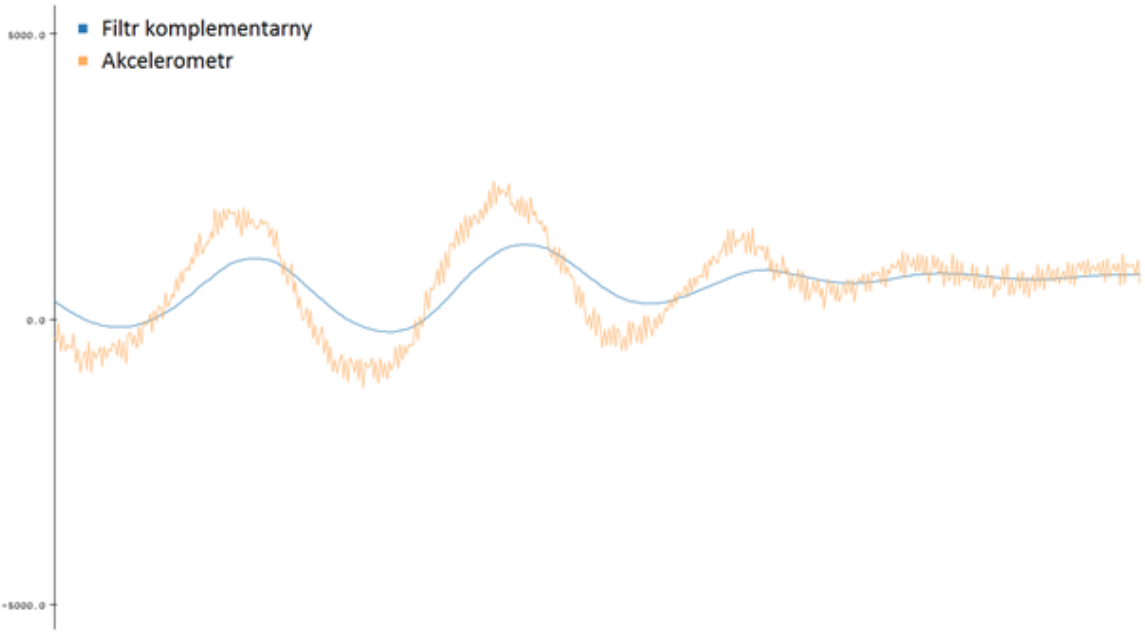
4.2.1 Filtr komplementarny

Filtr komplementarny jest najprostszą fuzją danych pochodzących z obu czujników (akcelerometru i żyroskopu). Akcelerometr, którego odczyt jest podatny na zakłócenia, odfiltrowujemy dolnoprzepustowym filtrem. Wprowadza to do mierzonego sygnału opóźnienie. Drugi czujnik - żyroskop posiada błąd wolnozmienny (dryft), który w tym wypadku filtrujemy za pomocą górnoprzepustowego filtru. Jednak zostają przy tym usunięte pożyteczne składowe wolnozmienne. Połączenie obu odfiltrowań pozwala osiągnąć zadowalające wyniki. Implementacja filtru wygląda w następujący sposób:

$$angle = 0{,}98 \times (angle + gyrData \times dt) + 0{,}02 \times accData$$

Równanie 1

Dane z żyroskopu są dodawane z każdym krokiem czasowym do poprzedniej odczytanej wartości kąta, po czym są łączone z aktualną wartością akcelerometru. Jak widać na rysunku 26, wartość kąta z poprzedniej iteracji ma dużo większe znaczenie niż aktualny odczyt, dzięki temu szumy zostają częściowo zniwelowane.



Rysunek 26 - zastosowanie filtru komplementarnego

Źródło: opracowanie własne

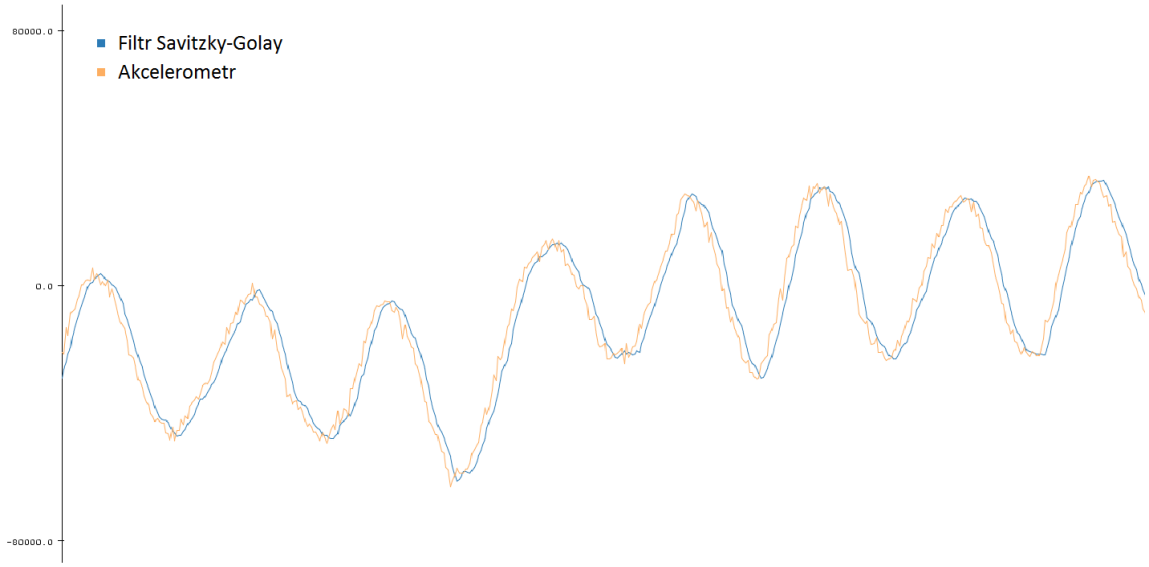
Wykonując jazdy próbne oraz analizując dane doszliśmy do wniosku, iż filtr działa zbyt wolno i nie możemy go zastosować w naszym pojedzie. W przydatku dynamicznych ruchów platformą, gdy dochodziło do 20 % przechyłu, dane po filtracji osiągały wynik jedynie około 10 – 12 %. Całość była obarczona dodatkowo około półsekundowym opóźnieniem, co widać na wykresie 26, po przesunięciu poziomym w prawo linii wykresu zaznaczonego kolorem granatowym względem linii pomarańczowej.

4.2.2 Filtr Savitzky-Golay

Filtr Savitzky-Golay jest powszechnie używany do wygładzania wykresu danych poprzez uśrednianie wyniku, biorąc pod uwagę sąsiednie wartości z określoną wagą [13]. Jego matematyczna postać została pokazana w równaniu 2.

$$Y_j = \frac{1}{35}(-3y_{j-2} + 12y_{j-1} + 17y_j + 12y_{j+1} - 3y_{j+2})$$

Równanie 2



Rysunek 27 - zastosowanie algorytmu Savitzky-Golay

Źródło: opracowanie własne

Analizując wykresy doszliśmy do wniosku, iż filtr działa zaskakująco dobrze. Niestety, podczas jazd próbnych ten algorytm nie sprawdza się dobrze. Pomimo, iż szumy są częściowo niwelowane, nadal istnieją oraz powstaje dodatkowo - tak samo jak w filtrze komplementarnym - delikatne opóźnienie w przeliczaniu danych. Całość przekłada się na niezbyt stabilną jazdę.

4.2.3 Filtr Kalmana

Filtr Kalmana jest algorytmem rekurencyjnym, minimalizującym średniokwadratowy błąd estymacji [10]. Obliczenia dokonywane są na podstawie wejścia i wyjścia układu. Jeśli odczytane dane będą bardzo zaszumione, filtr będzie zdawał się na wartości z predykcji. Jednak, gdy dane będą posiadały mały szum, filtr oprze się na danych pomiarowych. Jego matematyczna implementacja została przedstawiona w dwóch równaniach.

$$x_k = Ax_{(k-1)} + Bu_{(k-1)} + w_{(k-1)}$$

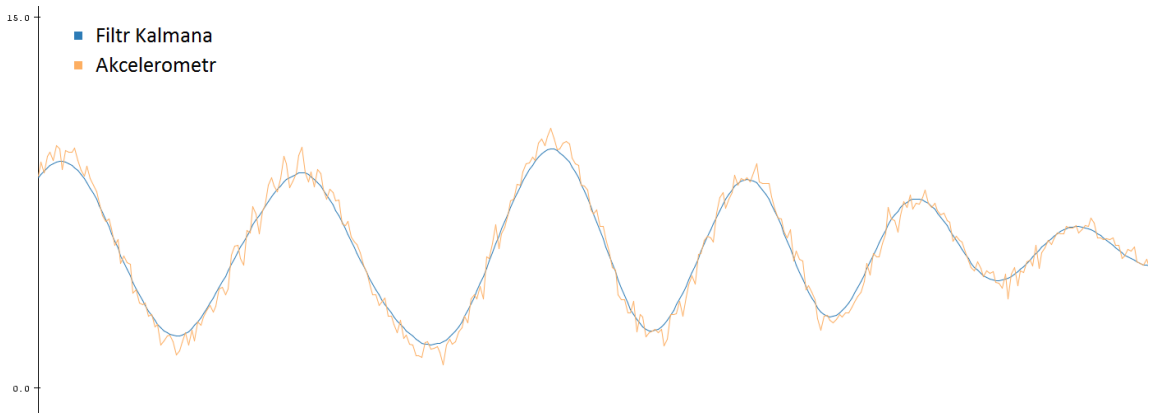
Równanie 3

$$z_k = Hx_k + v_k$$

Równanie 4

Pierwsze równanie nazywane jest równaniem czasu, a drugie to model pomiaru. Zmienne A, B, H są to odpowiednio macierz przejścia, wejścia i wyjścia. Zmienna W to tak zwane szumy procesu (cześć losowa), a V to zakłócenia. Zarówno W jak i V to białe szumy Gausa.

Wyniki tego filtru przedstawione na rysunku 28 w porównaniu do wyników poprzednich filtrów wypadły najkorzystniej. Metoda ta charakteryzuje się bardzo małym opóźnieniem, które było widoczne w przypadku użycia wcześniejszych rozwiązań. Filtr ten zachowuje również szybkość i sprawność działania, to znaczy, że zachowuje dynamiczną zmianę wartości.



Rysunek 28 - zastosowanie filtru Kalmana

Źródło: opracowanie własne

4.3 Logika pojazdu

W poniższym rozdziale przedstawiliśmy oprogramowanie jakie zostało zaimplementowane przez nas w kontrolerze Arduino UNO R3 do sterowania pojazdu.

4.3.1 Definicje

Na początku kodu implementujemy potrzebne biblioteki takie jak: Wire.h – podstawowa biblioteka do Arduino, MPU6050.h – zasób do czytników, KalmanFilter.h – filtr Kalmana i I2Cdev.h – biblioteka do komunikacji po magistrali I^2C. Następnie definiujemy dwa czujniki MPU6050 zamontowane w różnych miejscach. Pozostałe procedury to określenie potrzebnych zmiennych, które zostały opisane na rysunku 29.

```
#include <Wire.h>
#include <MPU6050.h>
#include <KalmanFilter.h>
#include <I2Cdev.h>

MPU6050 mpu1;   // Modul na palatformie
MPU6050 mpu2;   // Modul na kierownicy

// Konfiguracja filtru Kalmana dla osi X i Y (kat, odchylka, pomiar)
KalmanFilter kalmanX1(0.001, 0.003, 0.03);
KalmanFilter kalmanY1(0.001, 0.003, 0.03);
KalmanFilter kalmanX2(0.001, 0.003, 0.03);
KalmanFilter kalmanY2(0.001, 0.003, 0.03);

// Inicjalizacja zmiennych Pitch i Roll tylko z akcelerometru
float accPitch1 = 0;
float accRoll1 = 0;
float accPitch2 = 0;
float accRoll2 = 0;

// Inicjalizacja zmiennych Pitch i Roll
// z uwzglednieniem filtru Kalmana i zyroskopu
float kalPitch1 = 0;
float kalRoll1 = 0;
float kalPitch2 = 0;
float kalRoll2 = 0;

// Inicjalizacja zmiennych dla sterownikow silnikow
float levelp = 0;
float levell = 0;

// Piny sterownikow silników
int l = 11;
int p = 3;

```

Rysunek 29 - logika, definicje

Źródło: opracowanie własne

4.3.2 Funkcja setup

Część kodu przedstawiona na rysunku 30 jest wykonywana tylko jeden raz podczas uruchomienia pojazdu. Zawarliśmy tutaj procesy takie jak wykrywanie poprawnie podłączonych czujników oraz ich kalibracje. W tej części również ustawiliśmy odpowiednie częstotliwości modulacji sygnału PWM oraz zdefiniowaliśmy piny połączeniowe do sterowników silników.

```
36  void setup()
37  {
38    // Inicjalizacja komunikacji
39    Serial.begin(115200);
40
41    // Inicjalizacja MPU6050 na palatformie
42    //Serial.println("Inicjalizacja MPU6050 na palatformie");
43    while (!mpu1.begin(MPU6050_SCALE_2000DPS, MPU6050_RANGE_2G, (0x69)))
44    {
45      //Serial.println("Nie znaleziono MPU6050 na palatformie!");
46      //delay(500);
47    }
48
49    // Kalibracja zyroskopu na palatformie
50    mpu1.calibrateGyro();
51
52    // Inicjalizacja MPU6050 na kierownicy
53    //Serial.println("Inicjalizacja MPU6050 na kierownicy");
54    while (!mpu2.begin(MPU6050_SCALE_2000DPS, MPU6050_RANGE_2G, (0x68)))
55    {
56      //Serial.println("Nie znaleziono MPU6050 na kierownicy!");
57      //delay(500);
58    }
59
60    // Kalibracja zyroskopu na kierownicy
61    mpu2.calibrateGyro();
62
63    // Ustawienie czestotliwosci PWM na pinie 3 i 12
64    // TCCR2B = TCCR2B & 0b11111000 | 0x05;
65    TCCR2B = TCCR2B & B11111000 | B00000101;
66
67    // Ustawienie pinow silnikow jako wyjscie
68    pinMode(p, OUTPUT);
69    pinMode(l, OUTPUT);
70  }
71
```

Rysunek 30 - logika, setup

Źródło: opracowanie własne

4.3.3 Funkcja loop

Ten zestaw instrukcji jest wykonywany nieskończoną ilość razy i odpowiada za aktywną regulacje pozycji pojazdu. W każdej iteracji odczytywane jest aktualne położenie w postaci wektorów i na podstawie których wyznaczany jest obrót. Następnie te dane są przefiltrowywane przez filtr Kalmana. Dodatkowo uwzględniając nieidealne zamontowanie czujników, do zmiennych kalPitch1 i kalPitch2 zostały dodane odpowiednie wartości, aby w wartości początkowe wynosiły zero. Tą operacje nazywa się programową kalibracją. Następnie wartość obrotu jest mapowana. Proces ten polega na zamianie wartości obrotu do postaci skwantowanej (dyskretnej), czyli zastąpieniu wartości zmieniających się płynnie do wartości zmieniających się skokowo. Kolejnym blokiem kodu zawartego w instrukcjach warunkowych to uwzględnienie wszystkich możliwych ruchów takich jak jazda do przodu, do tyłu oraz skręcanie. Na końcu odpowiednio wyliczone wartości są przesyłane do silników.

```
void loop()
{
  // Odczytanie wektorow z czujnikow
  Vector acc1 = mpu1.readNormalizeAccel();
  Vector gyr1 = mpu1.readNormalizeGyro();
  Vector acc2 = mpu2.readNormalizeAccel();
  Vector gyr2 = mpu2.readNormalizeGyro();

  // Kalukacja Pitch &amp;amp; Roll z akcelerometru
  accPitch1 = -(atan2(acc1.XAxis, sqrt(acc1.YAxis * acc1.YAxis
                + acc1.ZAxis * acc1.ZAxis)) * 180.0) / M_PI;
  accRoll1  = (atan2(acc1.YAxis, acc1.ZAxis) * 180.0) / M_PI;
  accPitch2 = -(atan2(acc2.XAxis, sqrt(acc2.YAxis * acc2.YAxis
                + acc2.ZAxis * acc2.ZAxis)) * 180.0) / M_PI;
  accRoll2  = (atan2(acc2.YAxis, acc2.ZAxis) * 180.0) / M_PI;

  // Kalman - dane z akcelerometru i zyroskopu
  kalPitch1 = kalmanY1.update(accPitch1, gyr1.YAxis);
  kalRoll1  = kalmanX1.update(accRoll1, gyr1.XAxis);
  kalPitch2 = kalmanY2.update(accPitch2, gyr2.YAxis);
  kalRoll2  = kalmanX2.update(accRoll2, gyr2.XAxis);

  // Programowa kalibracja odczytów do 0
  kalPitch1 = kalPitch1 - 6;
  kalPitch2 = kalPitch2 - 6;

  // Mapowanie wartości (wartość 0 dla sterownikow to 95)
  levelp = map (
             kalPitch1,
             10,
             -10,
             105,
             85
           );
  levell = map (
             kalPitch1,
             10,
             -10,
             105,
             85
           );
  if ( kalPitch1 > 0)
  {
      if( kalPitch2 > 0)
      {
        levelp = levelp - kalPitch2;
        levell = levell + kalPitch2;
      }
      if( kalPitch2 < 0)
      {
        levelp = levelp - kalPitch2;
        levell = levell + kalPitch2;
      }
      if( kalPitch2 == 0)
      {
        levelp = levelp;
        levell = levell;
      }
  }
  if ( kalPitch1 <= 0)
      {
        levelp = levelp;
        levell = levell;
      }
  // Wysłanie sygnału do sterownikow
  analogWrite(p, levelp);
  analogWrite(l, levell);

}
```

Rysunek 31 - logika, loop

Źródło: opracowanie własne

5 PODSUMOWANIE

Celem naszej pracy inżynierskiej było zbudowanie pojazdu samobalansującego z ogólnodostępnych części na rynku, zachowując przy tym wszystkie zalety mobilnego robota do transportu ludzi. Poza samą budową i wprawieniem pojazdu w ruch udało nam się go ładnie wykończyć, dzięki czemu nie odstaje zbyt wiele od wzorca.

Znaczna część pracy jest poświęcona poszczególnym etapom budowy, ponieważ to ona zajęła najwięcej czasu. Dużo mniej czasochłonne okazało się programowanie pojazdu, które w przeciwieństwie do początkowych przewidywań okazało się mało skomplikowane. Dużym zaskoczeniem okazały się akumulatory, które nie potrzebowały doładowania podczas łącznie około 25 godzin testów.

Przedstawiony transporter stanowi finalną wersję testową. Dynamiczny rozwój technologii sprawia, że cały czas pojawiają się nowe możliwości ulepszenia pojazdu, dlatego w dalszych planach jest zmodyfikowanie pojazdu poprzez usunięcie kierownicy i zastosowanie platformy podzielonej na dwa ruchome elementy, jako źródło sterowania.

Nasz robot to nowoczesny i przyciągający uwagę pojazd, który można wykorzystać do celów rekreacyjnych lub do pracy. Wszystko to dzięki prostej i intuicyjnej obsłudze. Jest niedrogi w użytkowaniu, cichy i ekologiczny. Pojazd porusza się z łatwością po każdej powierzchni, a tam gdzie się pojawi stanowi niewątpliwą atrakcję. Dzięki swoim nietypowym zdolnościom poruszania się wzbudza zainteresowanie, a użytkownikowi przynosi wielką przyjemność z jazdy.

6 BIBLIOGRAFIA

[1] KĘDZIERSKI J. Robot balansujący - model i sterowanie. Wrocław 2008

[2] TCHOŃ K., MAZUR A., DULĘBA I., HOSSA R., MUSZYŃSKI R. Manipulatory i roboty mobilne: modele, planowanie ruchu, sterowanie, Akademicka Oficyna Wydawnicza PLJ, Warszawa, 2000

[3] KACZOREK T. Teoria sterowania i systemów Wydawnictwo Naukowe PWN

[4] WELCH G.,BISHOP G.: An Introduction to the Kalman Filter. University of North Carolina at Chapel Hill Department of Computer Science Chapel Hill, NC 27599-3175

[5] NEGENBORN R.: Robot Localization and Kalman Filters. On finding your position in a noisy world. Institute of Information and Computing Sciences in partial fulfilment of the requirements for the degree of Master of Science, specialized in Intelligent Systems

[6] CHI OOI R.Balancing a Two-Wheeled Autonomous Robot The University of Western Australia School of Mechanical Engineering, Final Year Thesis 2003

[7] WNUK M. SZLAWSKI R. Konstrukcja dwukołowego robota mobilnego RoBik Raport serii SPR nr 12/2004. Instytut Cybernetyki Technicznej Politechniki Wrocławskiej. Wrocław 2004

[8] Personal mobility vehicles and methods - http://www.google.com/patents/US6302230

[9] SZLAWSKI R. Konstrukcja podwójnego mostka H Raport 15/10/2004. Wrocław 2007

[10] KĘDZIERSKI J. Filtr Kalmana - zastosowania w prostych układach sensorycznych Raport Koła Naukowego Robotyków „KoNaR" 2008

[11] http://www.segway.com

[12] Segbot - Final project for the Introduction to Mechatronics class at the University of Illinois, 2004.

[13] William H. Press and Saul A. Teukolsky - Savitzky-Golay Smoothing Filters, American Institute of Physics, 1990

[14] http://www.arduino.cc

7 SPIS RYSUNKÓW

8 SPIS TABEL

9 SPIS RÓWNAŃ

Wyrażamy zgodę na udostępnienie naszej pracy w czytelniach Biblioteki SGGW.

.....

(czytelny podpis autora)

.....

(czytelny podpis autora)