

**Politechnika Warszawska**  
**Wydział Mechaniczny Energetyki i Lotnictwa**



**Praca przejściowa inżynierska  
Projekt prostego robota mobilnego  
sterowanego przy użyciu GUI**

Paweł Marton 304515

rok akademicki 2023/2024



---

## Contents

<b>1 Cel pracy</b>	<b>2</b>
<b>2 Wykorzystane narzędzia</b>	<b>2</b>
2.1 Arduino . . . . .	2
2.2 Qt . . . . .	2
<b>3 Opis warstwy sprzętowej</b>	<b>3</b>
3.1 Wykorzystane moduły . . . . .	3
3.2 Robot po złożeniu . . . . .	3
<b>4 Opis oprogramowania</b>	<b>4</b>
4.1 Qt . . . . .	4
4.1.1 Interfejs graficzny aplikacji . . . . .	4
4.1.2 Budowa i powiązanie działania obiektów . . . . .	5
4.1.3 Omówienie sygnałów i slotów klas . . . . .	6
4.1.4 Obsługa programu . . . . .	7
4.1.5 Komunikacja między aplikacją mobilną a Arduino . . . . .	9
4.2 Arduino . . . . .	10
<b>5 Napotkane problemy oraz sposoby ich rozwiązywania</b>	<b>11</b>
5.1 Konfiguracja środowiska . . . . .	11
5.2 Nieprawidłowa kolejność bajtów . . . . .	11
<b>6 Subiektywna ocena projektu</b>	<b>11</b>
<b>7 Podsumowanie i dalsze możliwości rozwoju projektu</b>	<b>12</b>



---

## 1 Cel pracy

Celem niniejszej pracy było zbudowanie robota mobilnego, wykorzystującego płytę Arduino Uno, a także napisanie oprogramowania na urządzenie z systemem operacyjnym Android, umożliwiającego zdalne sterowanie robotem przy pomocy Bluetooth. Obiekt powinien być w stanie poprawnie reagować na polecenia zadawane przez operatora, natomiast sposób sterowania za pomocą GUI powinien pozwalać operatorowi poprowadzić robota według wyznaczonej trasy.

## 2 Wykorzystane narzędzia

W celu wykonania zadania związanego z budową warstwy sprzętowej robota oraz napisania oprogramowania do jego sterowania wykorzystano dwie platformy, jakimi są Arduino oraz zestaw bibliotek z modułu Qt.

### 2.1 Arduino

Arduino jest otwartą platformą programistyczno-elektroniczną, której głównym celem jest szybkie i proste prototypowanie. Powodem wyboru tej platformy, oprócz jej prostoty, jest jej popularność oraz duża dostępność materiałów pomocniczych w internecie. Platforma ta została wykorzystana w projekcie do komunikacji między urządzeniem mobilnym a robotem za pomocą modułu Bluetooth HC-06 oraz do sterowania silnikami robota mobilnego w zależności od otrzymywanych sygnałów z urządzenia mobilnego.

### 2.2 Qt

Qt jest znaną platformą przeznaczoną do tworzenia graficznych interfejsów użytkownika. Narzędzie to łączy w sobie wiele bibliotek, które umożliwiają tworzenie wizualnie atrakcyjnych GUI w relatywnie prosty sposób. Dodatkowym atutem Qt, jest fakt, że napisane na tej platformie aplikacje są łatwo przenośne między urządzeniami posiadającymi różne systemy operacyjne – w przypadku opisanego projektu między komputerem, na którym będzie pisany program (Windows), a urządzeniem mobilnym (Android).

Edytorem kodu, który posłuży do pisania oraz komplikacji programu, został Qt Creator. Środowisko to daje możliwość budowy interfejsu w uproszczony sposób poprzez przeciąganie dodanych przez producenta widgetów w trybie Design, a następnie zaprogramowanie akcji, które mają się wydarzyć jako rezultat zdarzenia. Przykładowo, naciśnięcie przycisku ma spowodować zmianę tekstu na przycisku z "Jeszcze nie kliknięty" na "Już kliknięty". Kolejnym atutem wykorzystania tego narzędzia jest automatyczne generowanie pliku o rozszerzeniu .pro, będącego odpowiednikiem makefile, za pomocą którego nie ma potrzeby samodzielnej kontroli procesu komplikacji projektu. Ostatnią zaletą Qt jest jego intuicyjny sposób programowania zdarzeń, przedstawiony na rysunku 1.



**Figure 1:** Schemat obrazujący działanie obiektów w QT

Każdy obiekt klas wbudowanych w Qt posiada zdefiniowane sygnały, które są emitowane przy wystąpieniu określonych zdarzeń (np. naciśnięcia przycisku). Sygnał ten jest następnie przekazywany do połączonego slotu, który może należeć do tej samej lub innej klasy. Slotem nazywamy zdefiniowaną metodę danego obiektu, która zawiera instrukcje, które powinny się wykonać w momencie, gdy zostanie wyemitowany dany sygnał (np. po wyemitowaniu sygnału z przycisku informującego o jego naciśnięciu,

---

przechodzimy do połączonego z nim slotu widgetu ComboBox, który powinien w konsekwencji załadować do niego tekst "Przycisk kliknięty").

W Qt istnieje możliwość definiowania również własnych widgetów. Proces ten polega na napisaniu klasy dziedziczącej po klasie czysto wirtualnej QWidget, a następnie nadpisaniu jej metod oraz zdefiniowaniu dodatkowych metod, slotów i sygnałów.

W kontekście napisania własnego widgeta (Joysticka), nie było możliwości skorzystania z udogodnienia, jakim jest możliwość przeciągania i oprogramowania zdarzeń w trybie Design. Oczywiście istnieje możliwość użycia własnej klasy widgeta w trybie Design, jednak wiąże się to z koniecznością skonfigurowania całego środowiska. Ze względu na pracochność tej akcji zdecydowano się na ręczne zaprogramowanie interfejsu wewnętrz klasy głównego okna programu, jakim jest MainWindow.

### 3 Opis warstwy sprzętowej

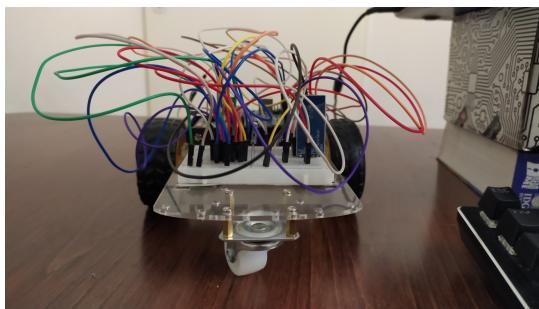
#### 3.1 Wykorzystane moduły

Do budowy prostego robota mobilnego wykorzystałem następujące elementy.

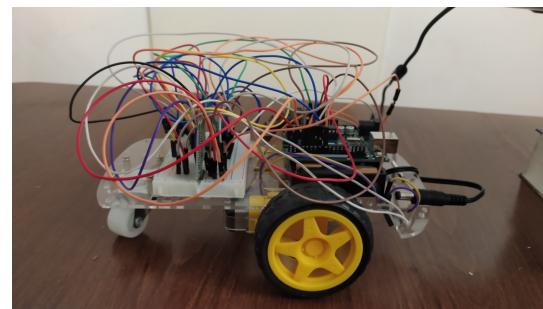
- Płytkę Arduino Uno.
- 2 Silniki DC.
- Płytkę stykową.
- Moduł Bluetooth HC-06.
- Przewody połączeniowe.
- Podwozie robota.
- Koło obrotowe.
- 2 Koła z oponami.
- Koszyk na 6 baterii 1,5 V.

#### 3.2 Robot po złożeniu

Robot po złożeniu został zaprezentowany na rysunku 2.



(a) Widok robota z przodu



(b) Widok robota z boku

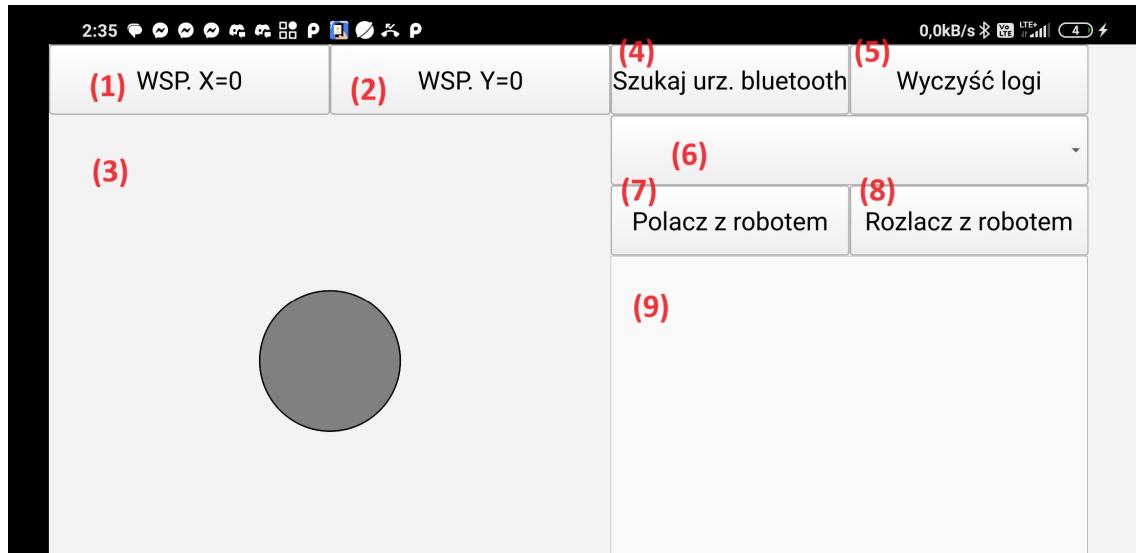
**Figure 2:** Robot po złożeniu

## 4 Opis oprogramowania

### 4.1 Qt

#### 4.1.1 Interfejs graficzny aplikacji

Interfejs graficzny użytkownika pokazano na rys. 3



**Figure 3:** Interfejs aplikacji mobilnej

Poszczególne cyfry naniesione na obraz są odniesieniem do widgetów okna aplikacji.

1. Przycisk podający informację o pierwszej współrzędnej położenia uchwytu od joysticka.
2. Przycisk podający informację o drugiej współrzędnej położenia uchwytu od joysticka.
3. Joystick, widget służący do sterowania robotem.
4. Przycisk rozpoczynający poszukiwanie urządzeń Bluetooth w otoczeniu.
5. Przycisk służący do czyszczenia tablicy logów zebranych w tablicy.
6. Rozwijane pole wyboru to element, w którym zapisane zostają informacje o wszystkich dostępnych urządzeniach Bluetooth.
7. Przycisk służący do połączenia się z robotem.
8. Przycisk służący do rozłączenia się z robotem.
9. Tablica logów.



#### 4.1.2 Budowa i powiązanie działania obiektów

Na diagramie 4 przedstawiono wpływ zdarzeń z widgetów na zachowanie innych obiektów (mechanizm łączenia sygnałów ze slotami). Ze względu na poziom skomplikowania programu, w diagramie ograniczono się do przedstawienia klas posiadających własne sloty i sygnały, które mają istotny wpływ na logikę działania całego programu.

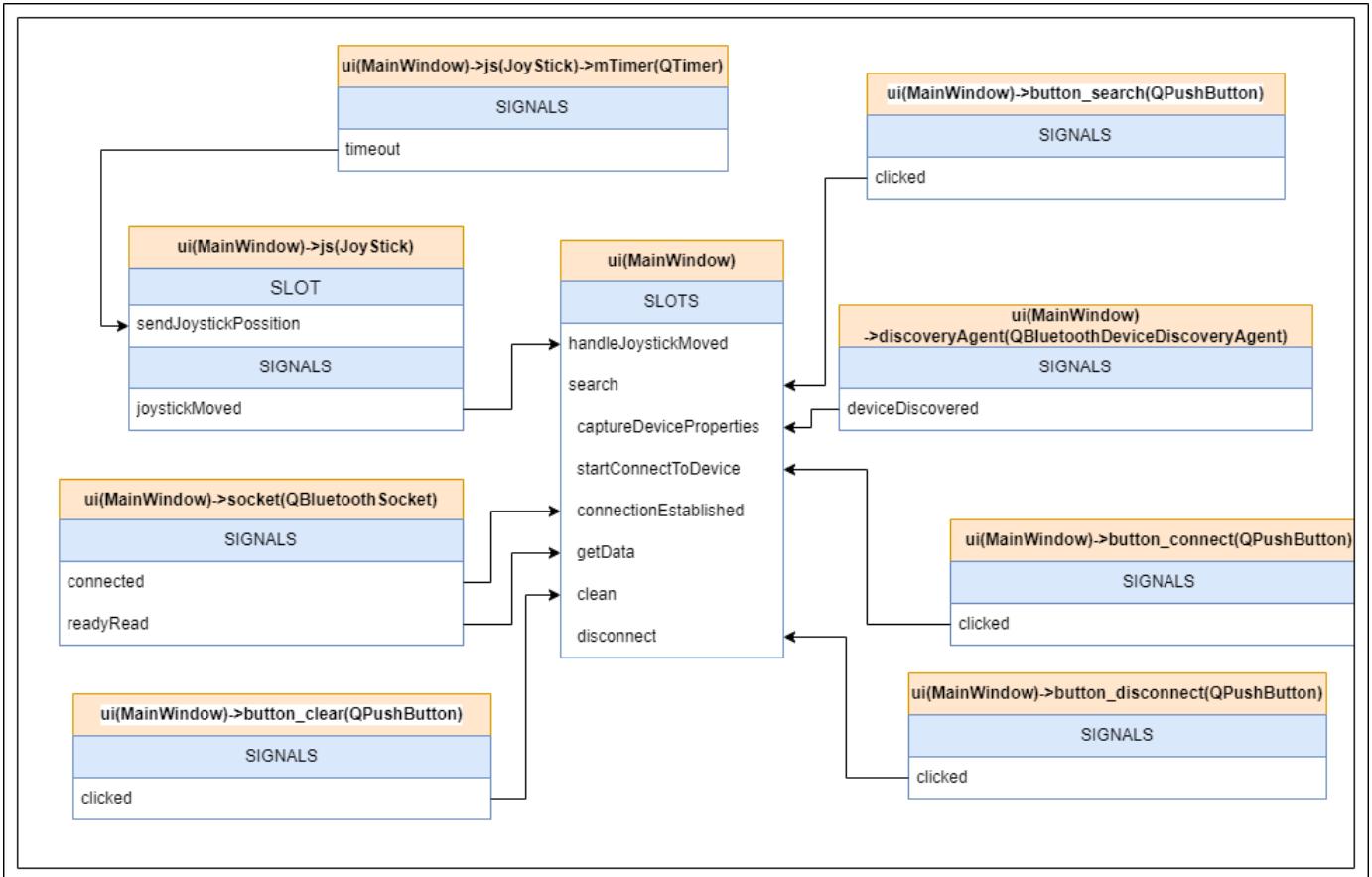


Figure 4: Diagram oddziaływania zdefiniowanych obiektów na siebie



#### 4.1.3 Omówienie sygnałów i slotów klas

W tabeli 1 przedstawione zostanie dokładniejszy opis działań slotów z diagramu 4

**Table 1:** Opis działania slotów obiektów zawartych w programie

Nazwa obiektu	Nazwa Slotu	Instrukcje wykonane przez dany Slot
js	sendJoystickPosition	Emituje sygnał joystickMoved wraz z aktualnymi współrzędnymi Joystika.
ui	handleJoystickMoved	1) Koreguje aktualną pozycję Joystika w ten sposób, aby początek układu współrzędnych względem, którego wyznaczana jest pozycja uchwytu znajdował się w środku Widgetu. 2) Zmienia napisy na przyciskach informujących o położeniu uchwytu od Joystika. 3) Na podstawie funkcji getModule i getDirection wyznacza jakie wartości bajtów należy przesyłać do robota. 4) Wysyła bajty do robota.
ui	search	Rozpoczyna poszukiwania urządzeń Bluetooth.
ui	captureDeviceProperties	Dodaje do ComboBox informacje dotyczące urządzeń Bluetooth znajdujących się w zasięgu urządzenia mobilnego.
ui	startConnectToDevice	Urządzenie mobilne rozpoczyna próbę połączenia się do wybranego w ComboBoxie urządzenia.
ui	connectionEstablished	Dodanie informacji o udanej próbie połączenia się do tablicy logów.
ui	getData	Dodanie do tablicy logów komunikatów przesyłanych z robota.
ui	clean	Czyści zawartość tablicy logów.
ui	disconnect	Urywa połączenie między urządzeniem mobilnym a robotem.

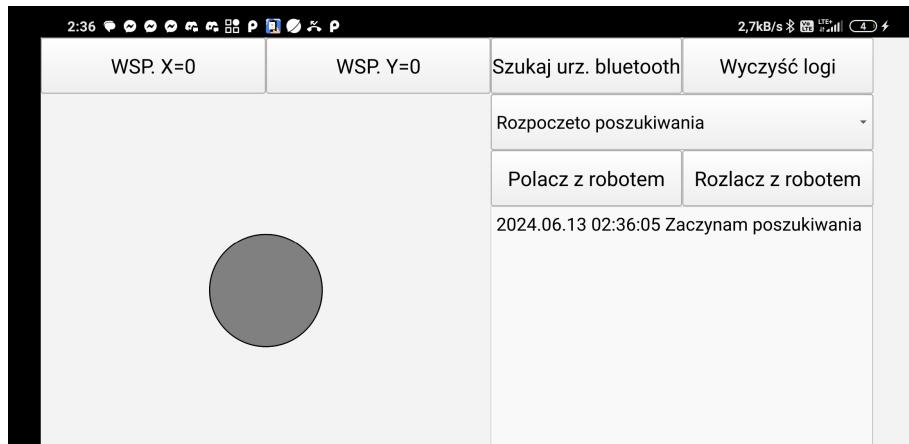
W tabeli 2 przedstawione zostaną opisy zdarzeń, na podstawie których z obiektów wyemitowane zostaną sygnały z diagramu 4

**Table 2:** Przyporządkowanie sygnałom emitujących je zdarzeń

Nazwa obiektu	Nazwa sygnału	Zdarzenie emitujące sygnał z obiektem
mTimer	timeout	Miniecie 0,5 sekundy od poprzednio wyemitowanego sygnału.
js	joystickMoved	Odebranie przez obiekt wyemitowanego przez Timer sygnału.
socket	connected	Urządzenie połączyło się z robotem.
button_clear	clicked	Naciśnięcie przycisku oznaczonego napisem "Wyczyść logi".
button_search	clicked	Naciśnięcie przycisku oznaczonego napisem "Szukaj urz. Bluetooth".
button_connect	clicked	Naciśnięcie przycisku oznaczonego napisem "Połącz z robotem".
button_disconnect	clicked	Naciśnięcie przycisku oznaczonego napisem "Rozłącz z robotem".
discoveryAgent	deviceDiscovered	Znalezienie w obrębie zasięgu urządzenia mobilnego innego urządzenia z włączoną usługą Bluetooth.

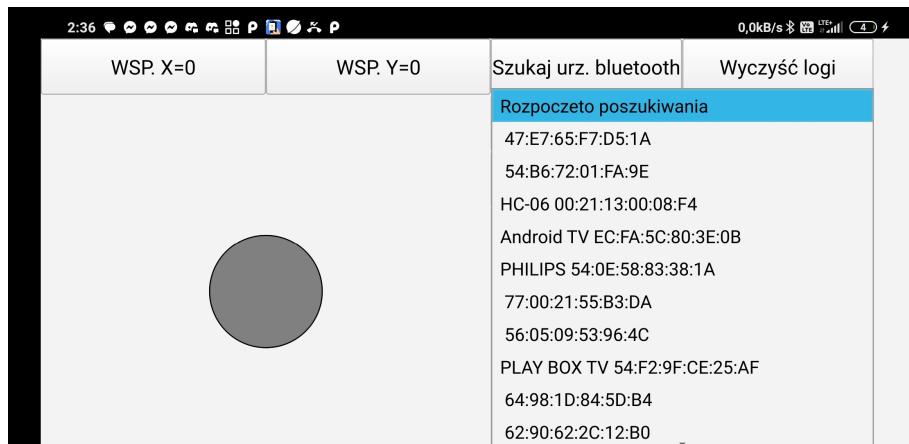
#### 4.1.4 Obsługa programu

W celu skorzystania z programu należy w pierwszej kolejności uruchomić aplikację i nacisnąć przycisk "Szukaj urządzeń Bluetooth". Wówczas w tablicy logów powinna pojawić się następująca informacja dotycząca rozpoczęcia poszukiwania urządzeń z włączoną usługą Bluetooth.



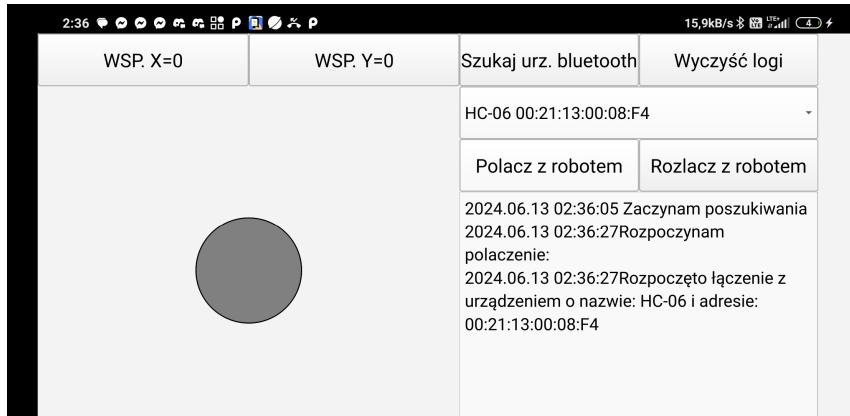
**Figure 5:** Komunikat po rozpoczęciu wyszukiwania urządzeń Bluetooth

Po krótkim czasie oczekiwania na liście dostępnych urządzeń w ComboBoxie pojawi się moduł HC-06 z robota, który należy wybrać.



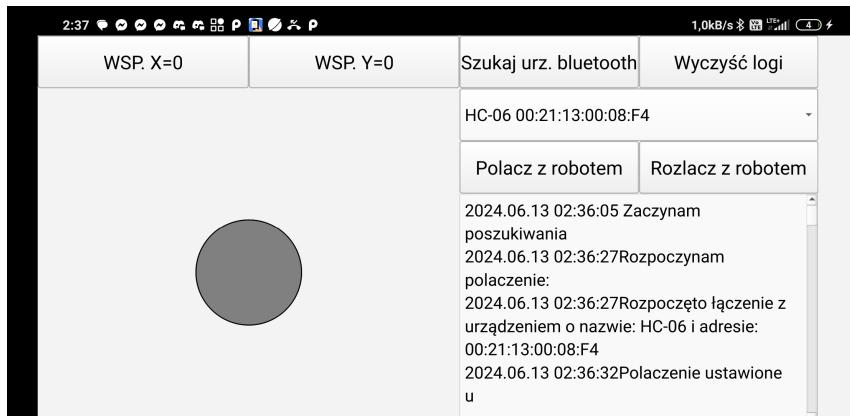
**Figure 6:** Lista dostępnych urządzeń z usługą Bluetooth w zasięgu urządzenia mobilnego

Po wyborze urządzenia należy nacisnąć przycisk "Połącz z robotem". Wówczas na monitorze zdarzeń powinna pojawić się informacja dotycząca rozpoczęcia procedury połączenia z modułem HC-06.



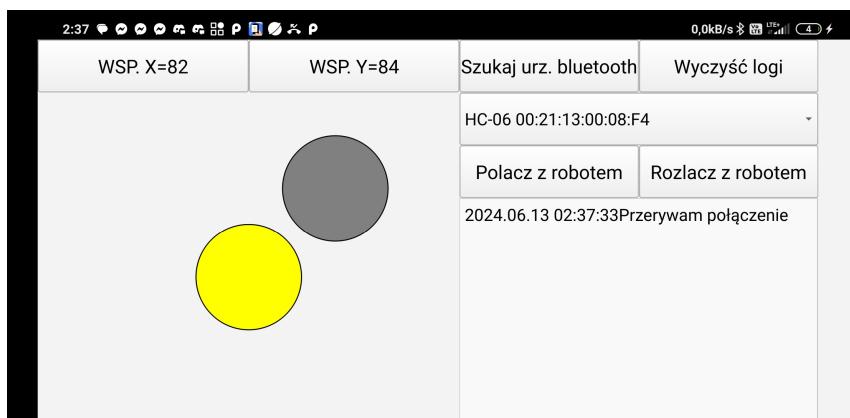
**Figure 7:** Rozpoczęcie procesu łączenia z modułem HC-06

Po pewnym czasie w tablicy logów powinna pojawić się wiadomość informująca o ustanowieniu połączenia między urządzeniami.



**Figure 8:** Ustawienie połączenia między urządzeniami

Po ustanowieniu połączenia między robotem a urządzeniem mobilnym można sterować robotem poprzez przeciąganie joysticka palcem od centrum widgetu. W momencie puszczenia ekranu kółko wróci do centrum, a robot się zatrzyma. Dodatkowo na rys. 9 zaprezentowani działań przycisku oznaczonego "Wyczyszc logi".

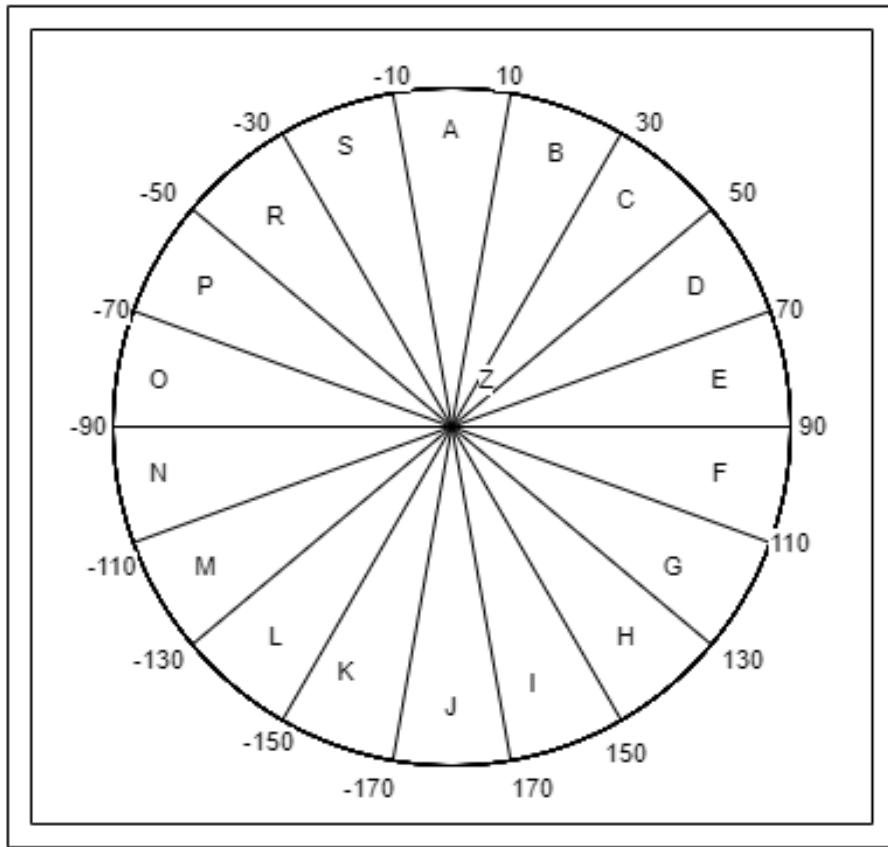


**Figure 9:** Sterowanie robota za pomocą Joysticka

#### 4.1.5 Komunikacja między aplikacją mobilną a Arduino

Komunikacja między urządzeniem mobilnym a robotem polega na przesłaniu z aplikacji dwóch bajtów, z których pierwszy wyznacza kierunek poruszania się, natomiast drugi szybkość z jaką dany robot się porusza.

Wartość pierwszego bajtu jest określana przez funkcję `getDirection`, która jest uzależniona od stosunku współrzędnych położenia uchwytu oraz znaku pierwszej koordynaty. W przypadku, gdy drążek sterujący znajdujący się w położeniu początkowym, wysyłanym bajtem będzie 'Z'. Dokładniejszy sposób wyznaczenia znaku zobrazowano na diagramie 10.



**Figure 10:** Sterowanie robotem-przekazywanie bajtów

Wartość drugiego bajtu jest określana przez funkcję `getModule`. Funkcja ta na podstawie wartości  $x^2 + y^2$  podejmuje decyzję o wartości znaku. Wartości przekazywanego bajtu są z zakresu 0-9. Sposób wyznaczenia wartości bajtu zostanie przedstawiony w tabeli 1.



## 4.2 Arduino

Program napisany w środowisku Arduino IDE służy do odbierania przesyłanych z urządzenia mobilnego sygnałów oraz ustawienia odpowiednich wartości napięć na silnikach. W tym celu w programie użyty został interfejs szeregowy "Serial", służący do komunikacji za pomocą portu szeregowego. Moduł ten początkowo sprawdza, czy na porcie znajdują się dwa bajty do odczytania, a następnie sprawdza kolejność przesłanych bajtów, czy jest zgodna z zaplanowaną, tj. czy pierwszy bajt zawiera informację o kierunku jazdy robota, a drugi o prędkości robota. Na podstawie wartości przesłanych bajtów program ustawia odpowiednie wartości na silnikach. W celu doboru wartości prędkości napędów skorzystałem z modulacji szerokości impulsu (PWM), która pozwala na ustawienie nieciągłej wartości prędkości. Wartości podawane na silniki w zależności od przesłanych bajtów z urządzenia mobilnego zostały przedstawione w tabelach 3 oraz 4.

**Table 3:** Tabela sterowań silników-kierunek ruchu

Bajt kierunku	Ustawienie kierunku silników	Ustawienie wartości prędkości na silnikach
'A'	Prawy: Przód Lewy: Przód	Prawy: value Lewy: value
'B'	Prawy: Przód Lewy: Przód	Prawy: 0,9 value Lewy: value
'C'	Prawy: Przód Lewy: Przód	Prawy: 0,8 value Lewy: value
'D'	Prawy: Przód Lewy: Przód	Prawy: 0,7 value Lewy: value
'E'	Prawy: Tył Lewy: Tył	Prawy: 0,6 value Lewy: value
'F'	Prawy: Tył Lewy: Tył	Prawy: 0,6 value Lewy: value
'G'	Prawy: Tył Lewy: Tył	Prawy: 0,7 value Lewy: value
'H'	Prawy: Tył Lewy: Tył	Prawy: 0,8 value Lewy: value
'I'	Prawy: Tył Lewy: Tył	Prawy: 0,9 value Lewy: value
'J'	Prawy: Tył Lewy: Tył	Prawy: value Lewy: value
'K'	Prawy: Tył Lewy: Tył	Prawy: value Lewy: 0,9 value
'L'	Prawy: Tył Lewy: Tył	Prawy: value Lewy: 0,8 value
'M'	Prawy: Tył Lewy: Tył	Prawy: value Lewy: 0,7 value
'N'	Prawy: Tył Lewy: Tył	Prawy: value Lewy: 0,6 value
'O'	Prawy: Tył Lewy: Tył	Prawy: value Lewy: 0,6 value
'P'	Prawy: Przód Lewy: Przód	Prawy: value Lewy: 0,7 value
'R'	Prawy: Przód Lewy: Przód	Prawy: value Lewy: 0,8 value
'S'	Prawy: Przód Lewy: Przód	Prawy: value Lewy: 0,9 value
'Z'	Prawy: Stop Lewy: Stop	Prawy: 0 Lewy: 0



**Table 4:** Tabela sterowań silników-wartość prędkości

Bajt prędkości	Zakres wartości zmiennej distance ( $x^2 + y^2$ )	Wartość zmiennej value (PWM)
'0'	0	0
'1'	(0,100)	100
'2'	[100,400)	120
'3'	[400, 900)	140
'4'	[900,1600)	160
'5'	[1600,2500)	180
'6'	[2500,3600)	200
'7'	[3600,4900)	220
'8'	[4900,6400)	240
'9'	[6400, $\infty$ )	255

## 5 Napotkane problemy oraz sposoby ich rozwiązania

### 5.1 Konfiguracja środowiska

Największą trudnością związaną z wykorzystaniem framework'a Qt w projekcie była konfiguracja niezbędnych narzędzi do budowania aplikacji na Androide, takich jak Java Developer Kit (JDK), Android Software Development Kit (SDK) oraz Android Native Development Kit (NDK) w Qt Creatorze. Pomimo wykorzystania dostarczonej i polecanej przez Qt możliwości instalacji tych elementów w prosty i automatyczny sposób za pomocą zakładki "Urządzenia" w opcjach środowiska podczas procesu budowy i instalacji projektu na urządzeniu mobilnym z systemem Android, wystąpił błąd związany z wersją systemu operacyjnego na urządzeniu.

Błąd ten, mimo komunikatu, nie był związany z wersją Androide, lecz z wersją JDK. Okazało się, że najnowsze dostępne narzędzia, które pobierają się za pośrednictwem tej zakładki, nie są kompatybilne z wersją IDE. Problem ten nie był szczegółowo opisany na forum użytkowników Qt. Rozwiązaniem okazała się zmiana ustawień w pliku build.gradle, gdzie konieczne było zmienienie 'androidx.core:core:1.10.1' na wcześniejszą wersję 'androidx.core:core:1.8.0'.

Kolejną trudnością było zainstalowanie programu na urządzeniu mobilnym. Użycie obiektów klasy QBluetoothDeviceDiscoveryAgent do wyszukiwania urządzeń Bluetooth w zasięgu urządzenia mobilnego, do wersji Qt 5.13 nie wymagało uprawnień do lokalizacji. Jednak zmiany w wymaganiach nie zostały w pełni zaktualizowane we wszystkich źródłach, w tym również w tych, z których korzystano. Brak wymaganych uprawnień do lokalizacji nie powodował błędu, jednakże moduł odpowiedzialny za wyszukiwanie urządzeń nie działał poprawnie, co utrudniało diagnozę problemu. Dopiero sprawdzenie wymagań w najnowszej wersji dokumentacji Qt pozwoliło rozwiązać ten problem.

### 5.2 Nieprawidłowa kolejność bajtów

W fazie testów została zaobserwowana również nieprawidłowość związana z przesyłaniem bajtów. W wyniku tej nieprawidłowości w pierwszej kolejności zamiast bajtu kierunku wysyłał się bajt prędkości. Podany problem został rozwiązyany w ten sposób, że przesyłane znaki zostały podzielone na bajty prędkości (0-9) oraz kierunku (A-S, Z). Wówczas przed każdą iteracją programu zapisanego na płytce Arduino sprawdzane jest, czy sekwencja znaków w trakcie przesyłania nie została zmieniona i jeśli wystąpiłaby taka sytuacja program zamienia kolejność znaków.

## 6 Subiektywna ocena projektu

Testy aplikacji mobilnej oraz robota potwierdziły, że sterowanie robotem działa poprawnie i obiekt sterowany ręcznie przez operatora jest w stanie poruszać się po wyznaczonej trasie. Robot nie rozmazuje się w trakcie pracy z urządzeniem mobilnym, co często zdarzało się w popularnej aplikacji MIT App Inventor, stworzonej do wykonywania podobnych zadań. W trakcie testów zaobserwowano jednak, że



---

z perspektywy operatora robota sterowanie nie jest intuicyjne i przed dobrym opanowaniem ruchów robota wymagany jest trening. Sterowanie robota było także obarczone niewielkim opóźnieniem między przesłaniem sygnału a rozpoczęciem ruchu robota będącego najprawdopodobniej wynikiem zdefiniowanego czasu emitowania sygnału z obiektu mTimer.

## 7 Podsumowanie i dalsze możliwości rozwoju projektu

Wykonany projekt spełnia założenia początkowe, umożliwiając niezakłóconą komunikację między pojazdem a urządzeniem mobilnym oraz umożliwiając sterowanie robotem wzdłuż wybranego toru za pomocą aplikacji mobilnej. Jednakże istnieje wiele aspektów, które mogłyby być ulepszone lub zmienione w dalszym rozwoju projektu.

Pierwszą kluczową poprawką, która mogłyby zostać wdrożona, jest dodanie do robota silników wyposażonych w enkodery. Umożliwiłyby one dokładne zliczanie obrotów kół, co pozwoliłoby na precyzyjne kontrolowanie kierunku i prędkości robota przy użyciu metod teorii sterowania.

Kolejnym ulepszeniem projektu powinno być przeskalowanie wartości napędowych dla robota. Aktualne wartości czasami nie wystarczają do ruszenia robota z miejsca przy niewielkim wychyleniu uchwytu Joysticksa z położenia początkowego, ze względu na opory ruchu.

Kolejną modyfikacją mogłyby być zastąpienie przewodów połączeniowych z płytka stykowej na płytkę drukowaną. Zmiana ta zmniejszyłaby ryzyko przypadkowego wypadnięcia przewodu z odpowiedniego miejsca na płytce stykowej, co mogłyby prowadzić do problemów z działaniem robota.

Ostatnim usprawnieniem mogłyby być zmienienie sposobu komunikacji między pojazdem a urządzeniem nadzorującym tak, aby wartości podawane na silniki były ciągłe, a nie zdyskretyzowane, jak to jest obecnie.

W ramach pracy, razem z raportem dołączone zostały pliki programu aplikacji mobilnej oraz skrytu działającego na robocie, które stanowią element rozwiązania.