TECHNICAL DOCUMENTATION OF THE “EMERGENCY RESCUE BOT”

AUTHORS: PAWEŁ SOBCZYŃSKI, SZYMON ZIAJEWSKI, WIKTOR ROSÓŁ

COORDINATORS: MATEUSZ STEFANIK, MAŁGORZATA ZIAJEWSKA

1. Components:
   1. Logical unit with a Bluetooth
   2. Camera
   3. Gripper motor controller
   4. Wheel motor controller
   5. Base servo
   6. Arm servo 1
   7. Arm servo 2
   8. Gripper rotation servo
   9. Gripper opening/closing motor
   10. Wheel motor 1
   11. Wheel motor 2
   12. 5V Power module
2. Robot:
   1. {zdjęcie robota z podpisami}
3. Operating principles of components:
   1. Servo:

The servo is connected using 3 cables: 5V power, ground, and PWM signal. PWM stands for pulse-width modulation. It is a method of regulating current or voltage signal with a constant amplitude and frequency. Full cycle time of this servo system is 20 ms (milliseconds). The duration of the current signal varies from 0.5 ms to 2.2 ms.

* 1. Motor controller:

The motor controller is connected using 4 cables: 5V power supply, ground, and two signal cables. The purpose of the controller is to rotate the motor in both directions and to stop it. To achieve this, two data cables are needed. The operating principle is as follows: if both signal cables carry a signal of 0 or 1, the motor stops. If the first cable carries a signal of 1 and the second carries 0, the motor rotates in one direction. If the first cable carries 0 and the second carries 1, the motor rotates in the opposite direction. Specifying the direction of rotation is not possible, because it depends on how the motor is connected to the controller..

* 1. Power module:

Power module consists lithium-ion battery 18650, that supplies the robot with current.

1. Control instructions:
   1. Left – rotate the base to the left
   2. Right – rotate the base to the right
   3. Up – first joint upwards
   4. Down – first joint downwards
   5. Square – second joint downwards
   6. Circle – second joint upwards
   7. Triangle – rotate gripper to the right
   8. Cross – rotate gripper to the left
   9. L1 – open gripper
   10. R1 – close gripper
   11. Left analog to the left – rotate robot to the left
   12. Left analog to the right– rotate robot to the right
   13. Right analog up – move forward
   14. Right analog down – move backward



Clicking more than one button simultaneously not recommended!

1. Code documentation:
   1. Technology:

Logic circuit has been programmed int the C++ language.

* 1. Imported libraries:

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Program uses two libraries. ESP32Servo - allows the ESP32 circuit to control the servo. The interface is identical to that for regular Arduino boards.

Ps3 Controller Host – emulates Ps3 console, allowing the connection of a DualShock 3 controller to the ESP32 circuit.

* 1. Pins in usage:

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Every used pin has been defined as a compiler constant.

* 1. Class MyServo:

To control the robot's servos effectively, it's necessary to keep track of the current servo position. For clarity and ease of use, a class called MyServo has been created. This class will retain the current position, control the servo, and upon program start, set the servo to its initial position.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Constructor MyServo():

Initializes the object, all variables, and sets the servo to the initial position.

The variable 'servo' points to an object of the Servo() class from the ESP32Servo library. The following methods are utilized:

.attach(int pin, int min, int max) - associates the pin with the servo and sets the minimum and maximum pulse width of the PWM signal in microseconds

.writeMicroseconds(int value) - positions the servo at a specific angle, where 'value' represents the pulse width in microseconds.

. write(int degree) - sets the angle at which the servo should be positioned (ranging from 0 to 180 degrees).

* 1. Class Engine:

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

The 'Engine' class is an implementation of the motor controller's functionality, which was explained in subsection 3.2.

* 1. Constants:

Obraz zawierający tekst, Czcionka, zrzut ekranu, biały

Opis wygenerowany automatycznie

Constants representing the default positions of the servos in the arm, set during the robot's startup.

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Constants representing the neutral position of the servos in the arm. This is a folded arm position that makes the movement and shutdown of the robot safer.

* 1. Inicjalizacja obiektów:

Obraz zawierający tekst, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

Initialisation of all necessary objects for controlling the robot.

* 1. Callback functions:

When a library invokes a certain function, users might want to add additional operations to it. This capability is provided by the Ps3 Controller Host library. To the Ps3 object, which has its own implementation for managing the DualShock 3 controller, it is possible to pass a callback function to assign custom behaviors to specific button or event.

* + 1. notify():

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

notify() function implements the robot control.

* + 1. connected():

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

A function primarily used for debugging, to indicate that the controller has been connected.

* + 1. disconnected():

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

A function primarily used for debugging, to indicate that the controller has been disconnected.

* 1. setup():

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

One of the two main functions in the Arduino language. It is inoked once at the beginning of the program.

* 1. loop():

Obraz zawierający tekst, Czcionka, pismo odręczne, biały

Opis wygenerowany automatycznie

The second main function in the Arduino language. It executes code in a loop. In this case, it's empty and it is defined to avoid compilation errors.