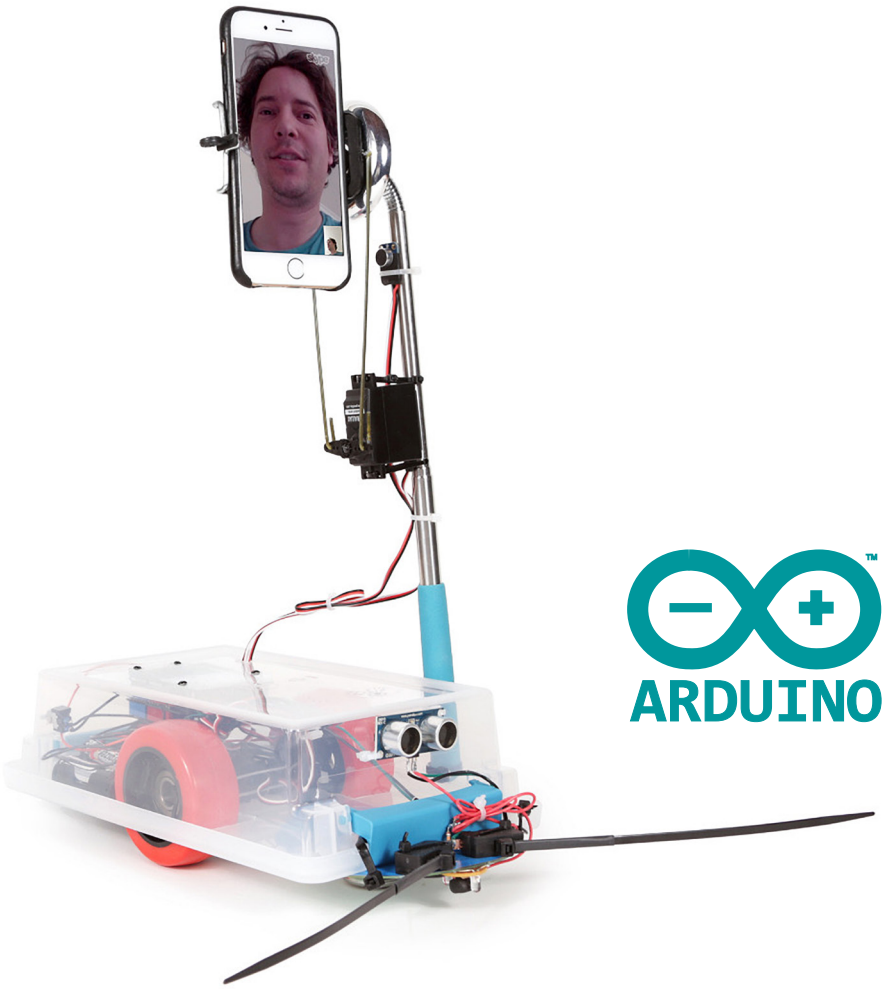
Robot do teleprezencji

Teleprezencja, nazywana również teleobecnością lub po prostu zdalną obecnością, nabrała w ostatnim czasie rozgłosu. Jest podstawą utrzymywania dystansu społecznego, który pozwala m.in. na ograniczenie przenoszenia się niebezpiecznych chorób.



Praca i nauka zdalna stały się jednym z podstawowych narzędzi ograniczania pandemii. Siedząc w domu, nie można wszak od nikogo się zarazić, więc wirus nie ma możliwości przenosić się dalej. W wielu wypadkach do zdalnej pracy czy nauki wystarczy zwykły komputer, smartfon lub tablet, istnieje jednak szereg aplikacji, gdzie potrzebne są bardziej funkcjonalne systemy teleobecności.

Robot do teleobecności to rodzaj robota, którym można sterować zdalnie i który może funkcjonować jako zastępca naszej fizycznej obecności w danym miejscu. Na przykład, jeśli jesteś w Warszawie, ale chcesz fizycznie wchodzić w interakcję z zespołem ludzi w Gdańsku, możesz połączyć się z robotem do zdalnej obecności i użyć go jako swojego zastępcy.

Podstawą stworzenia bardziej funkcjonalnego systemu teleprezencji jest możliwość poruszania się. Omówiona poniżej konstrukcja to prosty i tani robot wideo, którym można sterować przez Internet – może być uruchamiany wszędzie tam, gdzie istnieje do niego dostęp. Jest to elektromechaniczna platforma, wzbogacona o szereg czujników i elektronikę sterującą. System ten może się poruszać i przekazywać w obie strony sygnał wideo oraz audio.

Podstawa platformy robota

Podstawa robota skonstruowana jest z użyciem plastikowego pudełka, które zapewnia zarówno miejsce do montażu całej konstrukcji, jak i zamyka elektronikę i silniki w obudowie. Konstrukcja zawiera dwa koła napędowe przymocowane do serwomechanizmów z ciągłymi obrotami, które pozwalają im poruszać się do przodu i do tyłu, dzięki czemu robot może jeździć do przodu i do tyłu, a także obracać się w miejscu. Aby nie przechylać się na boki, podwozie ma dwa metalowe ślizgacze. Całość systemu kontrolowana jest poprzez Arduino.

Do budowy podwozia robota potrzebne będą:

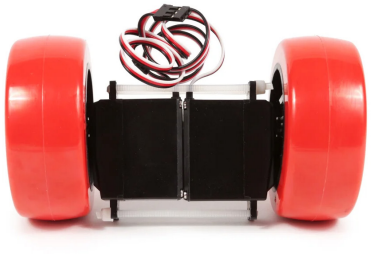
- serwa o ciągłej rotacji (×2),
- moduł Arduino,
- uchwyt baterii 4×AA,
- uchwyt baterii 2×AA,
- bateria AA (×6),
- wtyczka zasilania pasująca do Arduino,
- kółka (×2),
- plastikowe pudełko,
- zestaw śrub i nakrętek,
- przewody i rurki termokurczliwe,
- opaski zaciskowej („trytytki”) do montażu elementów – różne rozmiary.

Montaż należy rozpocząć od budowy napędu. Składa się on z dwóch serwomotorów o nieskończonych obrotach. Należy

a)

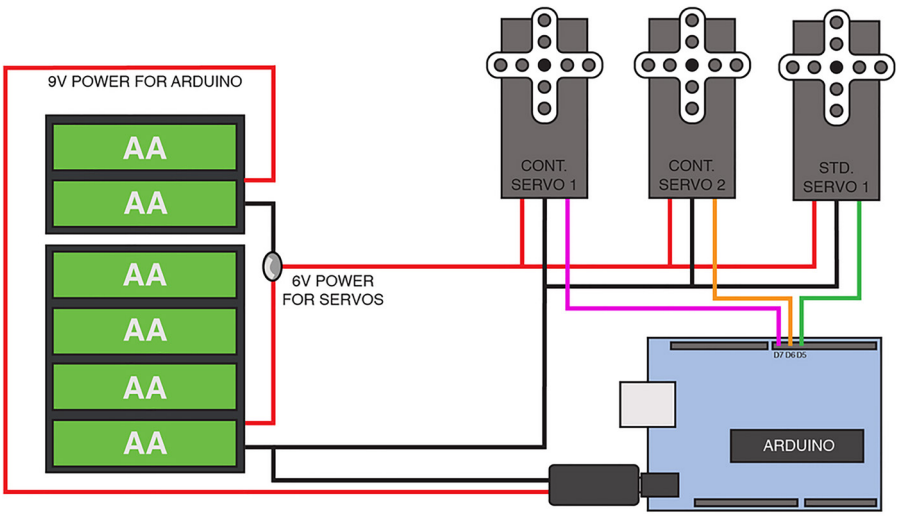
b)

c)



Fotografia 1. a) sposób rozwiercenia otworów montażowych w krzyżakach napędowych serwosilników; b) serwosilniki z zamontowanymi kołami; c) Kompletny moduł napędowy złożony z dwóch silników

Fotografia 2. Gotowa mechaniczna część podwozia



Rysunek 1. Schemat połączeń poszczególnych elementów w podwoziu

rozwiercić otwory montażowe w ramionach krzyżaków serwosilników (**fotografia 1a**) i przymocować do nich odpowiednio nawiercone kółka (**fotografia 1b**). Następnie operację należy powtórzyć z drugim serwosilnikiem i połączyć obydwa w lustrzanym odbiciu, co daje kompletną jednostkę napędową (**fotografia 1c**).

Moduł napędowy można następnie zainstalować w pokrywce plastikowego opakowania (takiego, jak do przechowywania żywności np. w lodówce). Przed przymocowaniem silników należy jedynie wykonać w wieku otwory na koła, aby mogły one wystawać na drugą stronę wieka. Następnie z przodu i z tyłu podwozia montowany jest metalowy ślizgacz, zapobiegający przewróceniu się robota. Gotową mechaniczną część podwozia pokazano na **fotografii 2**. Następnym krokiem jest zamontowanie w niej dwóch koszyków na baterie. W ten sposób generowane są dwa napięcia – 6 V do zasilania serwosilników oraz 9 V do zasilania modułu Arduino. Na **rysunku 1** zostało pokazane połączenie poszczególnych elementów podwozia.

Po drugiej stronie koszyków na baterie, na drugiej części plastikowego pojemnika, montowany jest moduł Arduino, sterujący konstrukcją. Aby sprawdzić poprawność montażu, do Arduino wgrać należy szkic z **listingu 1**. Pozwala on na sprawdzenie podstawowych funkcjonalności – robot porusza się losowo w różnych kierunkach. W kodzie znajdują się podstawowe funkcje do sterowania robotem. Po dodaniu kolejnych elementów robota program należy poszerzyć o ich obsługę.

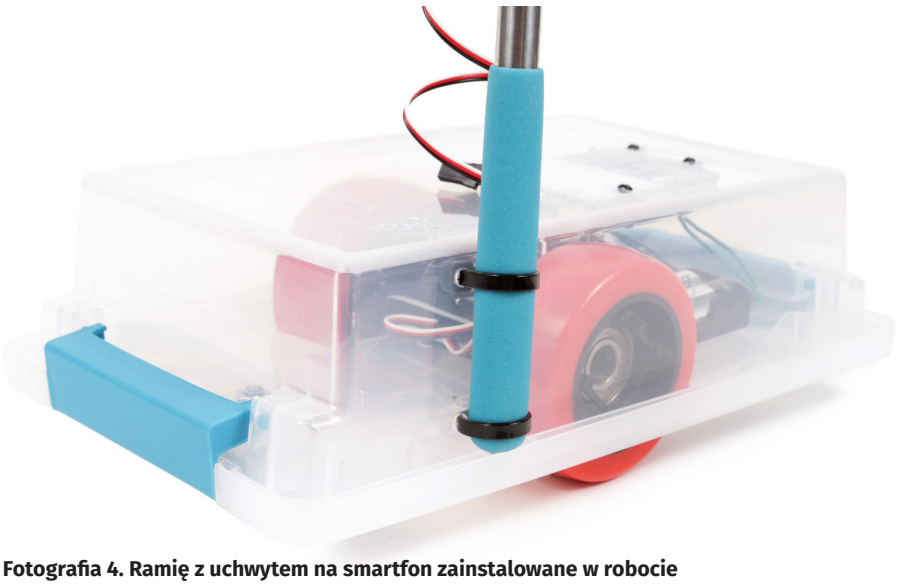
Fotografia 3. Sposób montażu smartfona w uchwycie z cięgnami

Ramię ze smartfonem

Kolejnym krokiem jest zamontowanie na jeżdżącej platformie ramienia, w którym zainstalowany zostanie smartfon. Do budowy ramienia potrzebne będą:

- selfie-stick, z uchwytem na smartfon,
- drut,
- standardowe serwo,
- metalowa blaszka (autor wykorzystał tutaj podkładkę do wykończania rur w ścianach),
- zestaw śrub i nakrętek.

Uchwyt do smartfona montujemy na metalowym elemencie, w którym nawiercamy odpowiednie otwory. Po połączeniu tych elementów do podstawki (blaszki) dołączamy dwa fragmenty drutu, które łączymy z serwomotorem i umieszczamy całość na selfie-sticku, jak pokazano na **fotografii 3**. Cięgna i serwo pozwolą na dostosowywanie pozycji telefonu w uchwycie.



Fotografia 4. Ramię z uchwytem na smartfon zainstalowane w robocie

Listing 1. Szkic do testowania działania ruchomej platformy robota

```
#include <Servo.h>          // Biblioteka do obsługi serwomotorów

Servo ContinuousServo1;     // Inicjalizacja serwomotorów z ruchem ciągłym
Servo ContinuousServo2;

void setup() {
    // Podłączenie serwomotorów z ciągłymi obrotami do pinów 6 i 7
    ContinuousServo1.attach(6);
    ContinuousServo2.attach(7);
    // Uruchom serwa jako spauzowane. Zmień wartość liczbową, jeśli się nie zatrzymują
    ContinuousServo1.write(94);
    ContinuousServo2.write(94);
}

void loop() {
    int range = random(4);  // Wybierz losową liczbę pomiędzy 0 a 3
    switch (range) {        // Przełączanie trybów oparte o wylosowaną liczbę
    case 0:                 // Jeśli wybrano 0, skręć w prawo i spauzuj na sekundę
        right();
        delay(500);
        stopDriving();
        delay(1000);
        break;
    case 1:                 // Jeśli wybrano 1, skręć w lewo i spauzuj na sekundę
        left();
        delay(500);
        stopDriving();
        delay(1000);
        break;
    case 2:                 // Jeśli wybrano 2, jedź prosto i spauzuj na sekundę
        forward();
        delay(500);
        stopDriving();
        delay(1000);
        break;
    case 3:                 // Jeśli wybrano 3, skręć w jedź wstecz i spauzuj na sekundę
        backward();
        delay(500);
        stopDriving();
        delay(1000);
        break;
    }
    delay(1);               // Pauza 1 ms dla poprawy stabilności kodu
}

void stopDriving() {        // Funkcja do zatrzymania się
    ContinuousServo1.write(94);
    ContinuousServo2.write(94);
}
void forward(){             // Funkcja do jechania naprzód
    ContinuousServo1.write(84);
    ContinuousServo2.write(104);
}
void backward(){            // Funkcja do jechania wstecz
    ContinuousServo1.write(104);
    ContinuousServo2.write(84);
}
void right(){               // Funkcja do jechania w prawo
    ContinuousServo1.write(104);
    ContinuousServo2.write(104);
}
void left(){                // Funkcja do jechania w lewo
    ContinuousServo1.write(84);
    ContinuousServo2.write(84);
}
```

Następnie dolną część selfie-sticku montujemy do plastikowego pudełka – bazy naszego robota, jak pokazano na **fotografii 4**. Można przetestować działanie trzeciego serwosilnika, korzystając z kodu na **listingu 2**. Inicjalizuje on trzecie serwo i rusza nim, przekręcając smartfon w górę, na środek, w dół, na środek i tak dalej – w pętli.

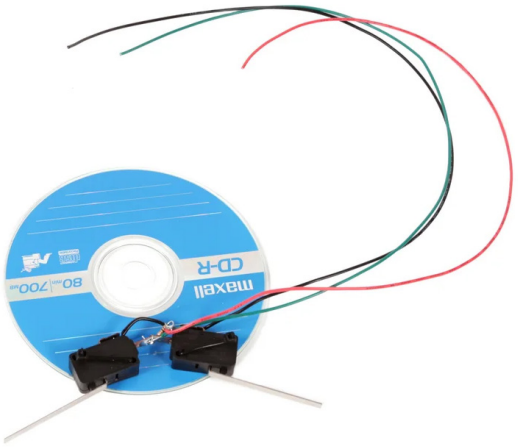
Detektory kolizji

Kolejnym elementem robota jest prosty detektor kolizji. Wykrywa on najechanie przez robota przednią częścią w obiekty. Do złożenia tego modułu potrzebne będą następujące elementy:

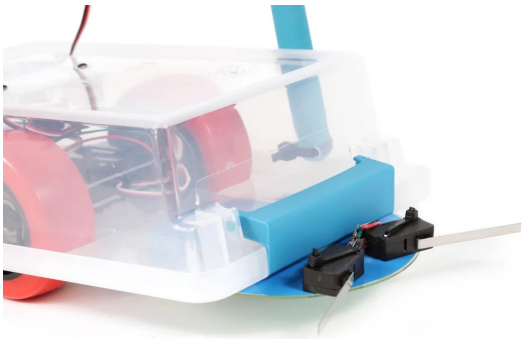
- przełączniki chwilowe SPST (×2) – patrz fotografie,

Listing 2. Szkic do testowania działania ramienia dla smartfona

```
#include <Servo.h>              // Biblioteka do obsługi serwomotorów

Servo StandardServo1;           // Inicjalizacja standardowego serwomotoru

void setup() {
    StandardServo1.attach(5);   // Podłączenie serwa do inu 5
    StandardServo1.write(90);   // Ustawienie serwa w pozycji neutralnej
    delay(2000);
}

void loop() {
    StandardServo1.write(135); //Przekręć uchwyt w górę i spauzuj
    delay(2000);
    StandardServo1.write(90);   // Przekręć uchwyt do środka i spauzuj
    delay(2000);
    StandardServo1.write(45);   // Przekręć uchwyt w dół i spauzuj
    delay(2000);
    StandardServo1.write(90);   // Wróć na środek i spauzuj
    delay(2000);
}
```



Fotografia 5. Moduł detektora zmontowany na płycie CD



Fotografia 6. Moduł detektora zamontowany w jeżdżącej podstawie

- podstawka pod detektory (autor wykorzystał płytę CD),
- oporniki 10 kΩ (×2),
- okrągłe PCB o średnicy ok. 25 mm,
- podkładki, śrubki, opaski zaciskowe i rurki termokurczliwe.

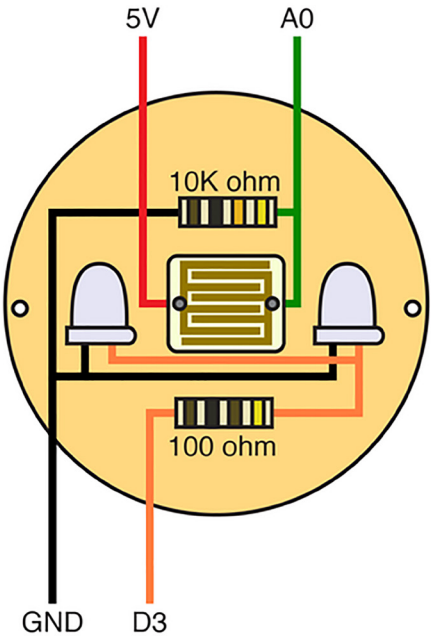
Przełączniki chwilowe montujemy na podstawce, w sposób pokazany na **fotografii 5**, a do jednego z wyprowadzeń lutujemy zasilanie poprzez opornik 10 kΩ, a do drugiego masę. Sygnał, który podłączamy do Arduino, pobieramy z drugiej nóżki opornika (tej, która nie jest podłączona do zasilania). Sygnał z detektora podłączany jest do wejścia cyfrowego na pinie 2 modułu Arduino. Moduł detektora dołączany jest do jeżdżącej podstawy tak, jak pokazano na **fotografii 6**.

Do przetestowania działania przycisku wykorzystać można prosty program, którego najistotniejsze fragmenty pokazano na **listingu 3**. Wykorzystuje on funkcje do sterowania silnikami z listingu 1 oraz **listingu 2**.

Sensor krawędzi

Kolejnym sensorem instalowanym w urządzeniu jest sensor krawędzi, który zabezpiecza robota przed np. spadnięciem ze schodów. Jest to prosty sensor odbiciowy, sprawdzający, czy pod przednią krawędzią robota jest podłoga, czy nie. Do zestawienia takiego układu potrzebne są:

- fotooporniki,
- diody LED (×2),



Rysunek 2. Schemat sensora krawędzi

- opornik 10 kΩ,
- opornik 100 Ω,
- uniwersalna płytka PCB.

Schemat sensora krawędzi pokazano na **rysunku 2**. Moduł składa się z dwóch niezależnych obwodów. Pierwszym z nich są diody LED, które są podłączane równolegle do cyfrowego pinu wyjściowego z Arduino (w tym przypadki pin D3), przez opornik 100 Ω. Pozwala to na włączanie i wyłączanie oświetlenia w sensorze. Arduino będzie faktycznie wykonywało dwa pomiary jasności – pierwszy będzie wykonywany przy zapalonych diodach, a drugi przy zgaszonych. Aby to zrobić, mikrokontroler musi mieć możliwość przełączania diod LED.

Drugą częścią czujnika jest fotorezystor, podłączany do analogowego pinu wejściowego. Jest on czujnikiem rezystancyjnym, co oznacza, że jego rezystancja zmienia się w zależności od tego, ile pada na niego światła. Fotooporник jest częścią dzielnika napięcia, złożonego jeszcze z opornika 10 kΩ. Dzielnik zasilany jest z napięcia 5 V, a zmiana rezystancji powoduje zmianę napięcia mierzonego przez wejście A0. Kod Arduino, potrzebny do wykrycia krawędzi za pomocą wejścia analogowego, pokazany jest na **listingu 4**.

Listing 3. Kod testujący detektory kolizji (fragment)

```
int buttonState = HIGH;             // Zmienna do przechowywania stanu przycisków

void setup() {
    pinMode(2, INPUT);              // Inicjalizacja pinu podłączonego do przycisku
                                    // (pin 2) jako wejścia
}

void loop() {
    buttonState = digitalRead(2);   // Odczytaj stan detektora zderzenia
    if (buttonState == HIGH) {      // Jeśli stan jest wysoki - brak zderzenia
        forward();                  // Jazda naprzód
    } else {
        backOff();                  // Wycofanie się po zderzeniu
    }
}

// A function which stops the robot, backs off, and changes direction.
void backOff(){                     // Funkcja do wycofania się
    stopDriving();                  // Zatrzymaj się
    delay(1000);
    StandardServo1.write(30);       // Spójrz w dół
    delay(2000);
    StandardServo1.write(70);       // Spójrz z powrotem w górę
    delay(2000);
    backward();                     // Wycofaj robota
    delay(1000);
    stopDriving();                  // Zatrzymaj się
    delay(1000);
    right();                        // Skręć w prawo
    delay(1000);
    stopDriving();                  // Zatrzymaj się
    delay(1000);
}
```

Listing 4. Kod odpowiedzialny za odczyt wejścia analogowego w module

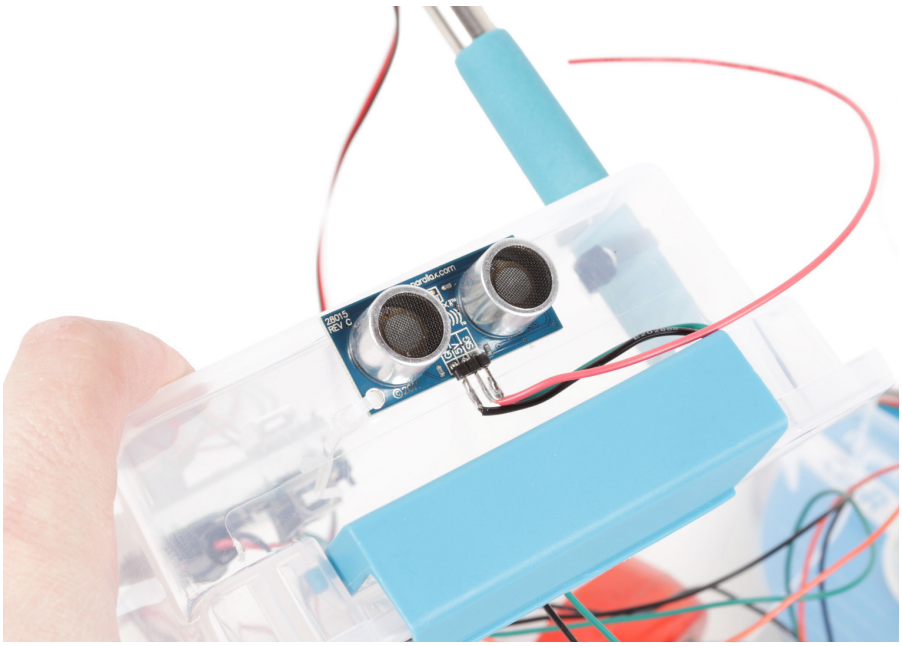
```
int sensorLEDoff;                   // Pomiar przy wyłączonych LED-ach
int sensorLEDon;                    // Pomiar przy włączonych LED-ach
int compareValue;                   // Zmienna do obliczania różnicy pomiędzy pomiarami

void loop() {
    edgeDetect();
    if (compareValue > 100) {
        forward();
    }
    else {
        backOff();                  // Funkcja do wycofywania się
    }
}

void edgeDetect(){                  // Funkcja detekcji krawędzi
    sensorLEDoff = analogRead(A0);  // Pomiar napięcia na pinie A0 przy wyłączonych LED-ach
    delay(10);
    digitalWrite(3, HIGH);          // Zapalenie LED-ów
    delay(5);
    sensorLEDon = analogRead(A0);   // Pomiar napięcia na pinie A0 przy włączonych LED-ach
    digitalWrite(3, LOW);           // Wyłączenie LED-ów
    compareValue = abs(sensorLEDon - sensorLEDoff);
                                    // Wyliczenie różnicy pomiędzy pomiarami
    delay(50);
}
```

Fotografia 7. Zmontowany sensor krawędzi

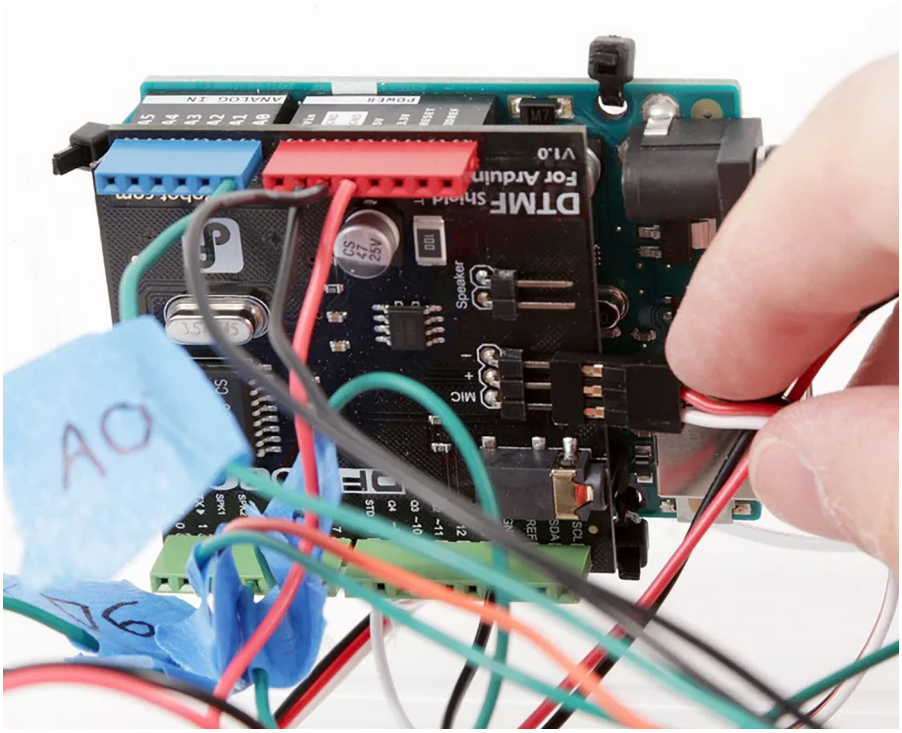
Fotografia 8. Sensor krawędzi zamontowany w robocie

Listing 5. Kod do obsługi sensora odległości

```
long duration;                      // Czas przelotu impulsu
long inches;                        // Odległość w calach

void distanceMeasurement(){         // Funkcja do pomiaru odległości
    // Sensor jest uruchamiany stanem wysokim na wejściu przez dłużej niż 2 mikrosekundy
    pinMode(4, OUTPUT);             // Ustawienie pinu 4 jako wyjście
    digitalWrite(4, LOW);           // Ustawienie stanu niskiego
    delayMicroseconds(2);
    digitalWrite(4, HIGH);          // Ustawienie stanu wysokiego
    delayMicroseconds(5);           // Opóźnienie co najmniej 2 mikrosekund
    digitalWrite(4, LOW);
    // Ten sam pin używany jest do odczytu.
    // Sensor wystawia impuls o czasie trwania przelotu
    pinMode(4, INPUT);              // Ustawienie pinu 4 jako wejście
    duration = pulseIn(4, HIGH);    //
    inches = microsecondsToInches(duration); // Konwersja czasu do odległości w calach
}

long microsecondsToInches(long microseconds) {
    // Zgodnie z kartą katalogową modułu sygnał potrzebuje 73,746 mikrosekundy, aby
    // przebyć jeden cal. W ten sposób wyznaczana jest droga impulsu. Należy podzielić
    // ją na pół, gdyż droga impulsu jest dwukrotnością odległości od przeszkody,
    return microseconds / 74 / 2;
}
```



Fotografia 9. Sensor odległości zainstalowany na platformie robota



Fotografia 10. Moduł Shield do odbierania kodów DTMF do sterowania robotem

Gotowy sensor zaprezentowano na **fotografii 7**. Należy umieścić go pod robotem na płycie CD, pełniącej funkcję wspornika detektora kolizji, jak pokazano na **fotografii 8**.

Czujnik odległości

Większość elementów, które do tej pory zamontowano w robocie, obejmuje nawigację zabezpieczającą robota przed bezpośrednim niebezpieczeństwem, takim jak zderzenie czy upadek z krawędzi. Czujnik odległości ma inne zastosowanie, gdyż służy do wykrywania rzeczy w większej odległości, dzięki czemu robot może podejmować świadome decyzje o tym, jak poruszać się dalej.

Czujnikiem, który zastosowano w tym projekcie, jest ultradźwiękowy sensor odbiciowy. Działa na podobnej zasadzie jak czujnik wykrywania krawędzi, ponieważ wysyła sygnał – w tym przypadku ultradźwięki, a nie światło z diody LED – a następnie mierzy, ile czasu potrzeba impulsowi, aby się odbić i wrócić do sensora. Obliczając, ile czasu zajmuje powrót i znając prędkość dźwięku w powietrzu, można obliczyć, jaka odległość dzieli sensor od obiektu, od którego dźwięk się odbił.

Czujnik, pokazany na **fotografii 9,** działa na odległość do około 4 metrów i zapewnia rozsądną dokładność pomiaru przy tak dużym obszarze. Zastosowanie go pomaga nie tylko w unikaniu obiektów, może również służyć do uzyskania lepszej orientacji w przestrzeni. Ma za zadanie zapobiegać aktywacji detektorów kolizyjnych – mają one być bardziej niezawodnym i awaryjnym elementem, a nie podstawowymi środkami interakcji ze światem.

Sensor ma trzy piny. Dwa z nich wykorzystane są do zasilania (5 V oraz GND), a trzeci należy podłączyć do pinu cyfrowego Arduino – *w* tym systemie podłączany jest do pinu 4. Funkcja mierząca odległość za pomocą tego sensora zamieszczona jest na **listingu 5**.

Odbiornik kodów DTMF

Ostatnim elementem jest moduł do odbioru kodów DTMF, w postaci modułu Arduino Shield. Jest to rozszerzenie do Arduino, wykorzystuje zewnętrzny mikrofon do odbierania i analizowania kodów DTMF, jakie można przesyłać np. przez telefon lub Skype.

W omawianym projekcie moduł ten używany jest do sterowania robotem poprzez Skype. Po naciśnięciu na klawiaturę odpowiedniego klawisza generowany i przesyłany jest kod DTMF. Robot odbiera ten dźwięk i reaguje na przesłany sygnał, odpowiednio się poruszając.

Aby podłączyć moduł Shield, należy odłączyć wszystkie dotychczasowo podłączone sygnały (pamiętaj, aby je oznaczyć) i umieścić moduł na swoim miejscu, jak pokazano na **fotografii 10.** Widoczny jest także sposób,

Listing 6. Sposób wykorzystania biblioteki dtmf.h w Arduino

```
#include "dtmf.h"               // Biblioteka do obsługi modułu DTMF

DTMF dtmf;                      // Inicjalizacja modułu DTMF
int myDtmf;                     // Zmienna do przechowywania kodów DTMF

void loop(){
    myDtmf = dtmf.getDTMF();    // Odczytanie kodu DTMF
    // W zależności od otrzymanego kodu, uruchamiane są różne funkcje,
    // takie jak poruszanie się, ruch smartfonem etc.

    switch(myDtmf){
        case 2:
            forward();
        break;
        case 8:
            backward();
        break;
        case 4:
            left();
        break;
        case 6:
            right();
        break;
        case 7:
            tiltUp();
        break;
        case 9:
            tiltDown();
        break;
        default:
            stopDriving();
        break;
    }
    // Pauza na końcu pętli pomaga szkicowi działać płynnie
    delay(10);
 }
```

w jaki podłączone są pozostałe sygnały. Do modułu DTMF dołączony jest mikrofon, który odbiera głos ze smartfona.

Jeśli chodzi o oprogramowanie modułu, jest ono oparte na dedykowanej bibliotece *dtmf.h*, która jest do pobrania ze strony projektu (patrz link na końcu artykułu). Sposób zastosowania jej w kodzie pokazano na **listingu 6**.

Oprogramowanie

Na **listingach 1…6** pokazano różne elementy, które składają się na w pełni funkcjonalnego robota. Kompletny szkic Arduino dla robota dostępny jest na stronie projektu oraz w materiałach dodatkowych do artykułu (**listing 7**).

Obsługa robota jest bardzo prosta. Wystarczy umieścić w nim smartfon z dostępem do Internetu i zainstalowanym Skype (na przykład). Po połączeniu się z komunikatorem można rozpocząć działanie – Skype przekazuje obraz i dźwięk w obie strony, a sterowanie ruchem robota zrealizowano za pomocą kodów DTMF. Robot reaguje na następujące kody:

- 2: jazda w przód,
- 4: skręt w lewo,
- 6: skręt w prawo,
- 8: jazda do tyłu,
- 7: pochylenie smartfona w dół,
- 9: pochylenie smartfona w górę.

Podsumowanie

Omówiony robot umożliwia teleobecność. Aplikacja na smartfonie odpowiedzialna jest za dwukierunkową transmisję obrazu i dźwięku, a możliwość poruszania się dodatkowo poprawia wrażenie zdalnej obecności. System taki idealnie nadaje się do zastosowań, np. przy zdalnej inspekcji pomieszczeń itp. Tego rodzaju maszyny chętnie stosowane są obecnie np. w hospicjach czy szpitalach, aby ograniczać przenoszenie chorób.

Nikodem Czechowski, EP

Źrodło: http://bit.ly/3pue7hm