



AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Inteligentny karmnik dla chomików

Autor: Waldemar Świder
Kierunek: Informatyka techniczna
Przedmiot: Inteligentne systemy pomiarowe
Nr albumu: 303174

Kraków, 18.05.2023r

Spis treści

1. Cel projektu	3
2. Wykorzystane komponenty i urządzenia.....	3
3. Elementy wydrukowane	3
4. Oprogramowanie	7
4.1. Podsumowanie wykorzystywanych technologii.....	7
4.2. Skrypt w Arduino.....	8
4.3. API (Python)	11
5. Funkcjonalności	12
5.1. Pomiar nacisku	12
5.2. Przesypywanie karmy	13
5.3. Powiadomienie o niskim poziomie karmy w zbiorniku	15
5.4. Zdalne sterowanie mechanizmem	16

1. Cel projektu

Inteligentny karmnik dla chomików to łączy zaawansowane funkcje technologiczne z troską o dobrostan małych gryzoni. Dzięki temu rozwiązaniu można zapewnić chomikom odpowiednią ilość pożywienia oraz monitorować ich dietę celem dostosowania jej do indywidualnych potrzeb pupili. Karmnik ma za zadanie automatyczne podawanie odpowiednich porcji jedzenia. System kontrolujący ilość karmy pozwala na precyzyjne dostarczenie odpowiedniej ilości pokarmu w regularnych interwałach czasowych. Jest to o tyle istotne, że chomiki mają tendencję do przejadania się, co prowadzi do otyłości i powiązanych problemów zdrowotnych. Inteligentny karmnik pozwala uniknąć tego zagrożenia.

Ponadto, karmnik jest wyposażony w czujnik nacisku. Dzięki niemu można śledzić poziom karmy w zbiorniku na górze. W razie niskiego poziomu karmy w zbiorniku odpowiednia informacja zostanie przesłana do połączanego API. To umożliwia przekazanie tej informacji np. potencjalnej aplikacji mobilnej. Proponowane rozwiązanie dzięki automatycznemu podawaniu odpowiednio obranych porcji jedzenia, monitorowaniu nawyków żywieniowych i zdalnemu dostępowi do mechanizmu przesypującego karmę, zapewnia ułatwienie w dbanie o wszelkie gryzonie domowe.

2. Wykorzystane komponenty i urządzenia

Do wykonania projektu wykorzystano następujące komponenty:

- Płytką rozwojową ESP-WROOM-32 (Wifi + Bluetooth)
- Silnik krokowy 28BYJ-48
- Gniazdo USB-A
- Czujnik nacisku SEN0297
- Filament PLA Devil Design (została z niego wykonana obudowa oraz mechanizm przesypujący karmę w technologii druku 3D)

3. Elementy wydrukowane

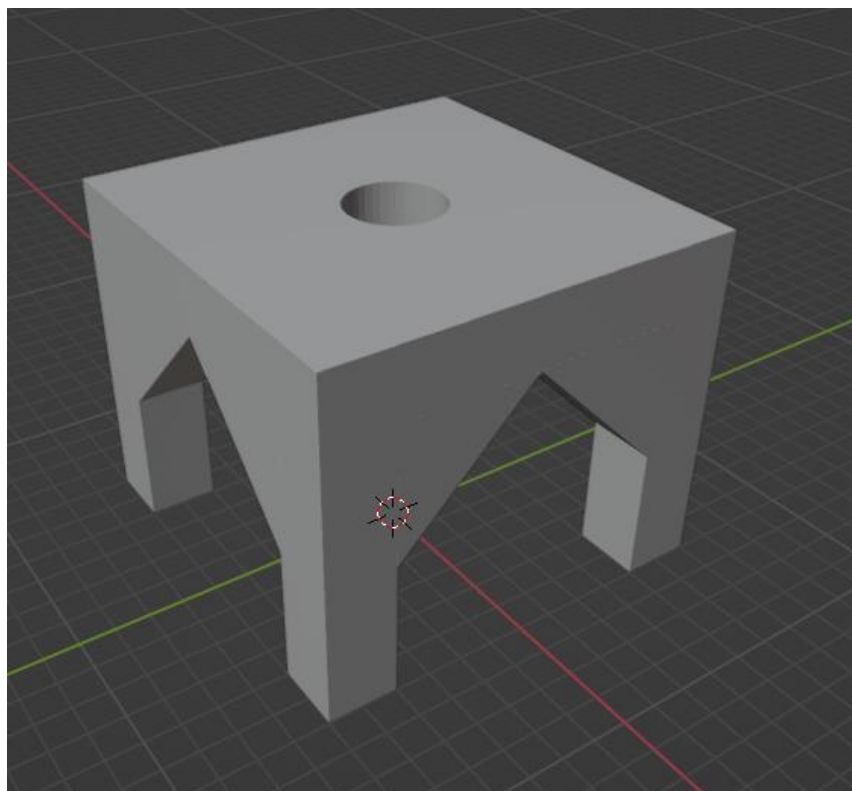
Biorąc pod uwagę specyfikację proponowanego rozwiązania problemu, elementy składające się na działające urządzenie muszą znajdować się w specjalnie zaprojektowanej obudowie. W tym celu samodzielnie zaprojektowano oraz wydrukowano niezbędne elementy karmnika. Część projektowa została wykonana w programie Blender.

Obudowa inteligentnego karmnika dla chomiczków powinna być zaprojektowana tak, aby zapewnić wygodę, bezpieczeństwo i funkcjonalność zarówno dla zwierzęcia, jak i dla właściciela. Oto kilka aspektów, które uwzględniono przy projektowaniu obudowy:

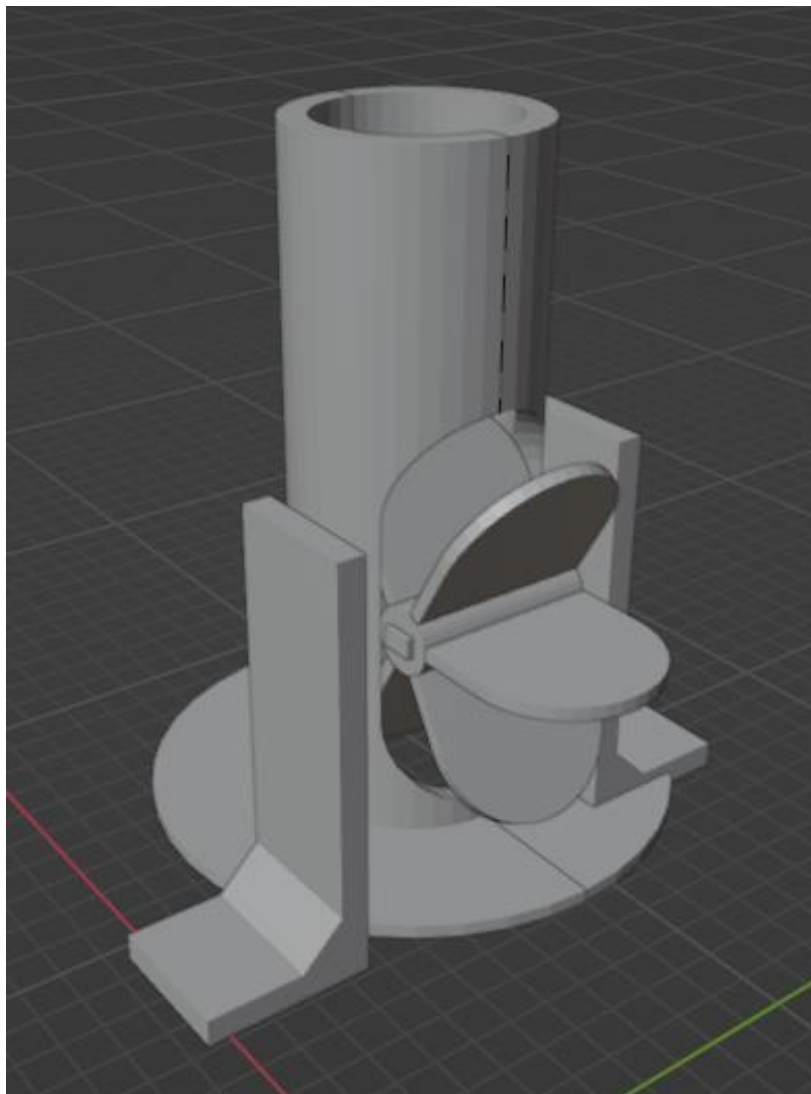
1. Rozmiar i przestrzeń: Obudowa powinna być odpowiednio przemyślana pod względem rozmiaru i przestrzeni, aby zapewnić wystarczające miejsce dla chomika do swobodnego poruszania się i wygodnej konsumpcji karmy.
2. Bezpieczeństwo: Powinna być wykonana z materiałów nietoksycznych i łatwych do czyszczenia, aby zapewnić higienę i zapobiec przypadkowemu zatruciu. Wszystkie elementy obudowy muszą być solidne i trwałe, aby zapobiec dostępowi do niebezpiecznych przedmiotów.
3. Łatwość użycia: Obudowa powinna być zaprojektowana w sposób, który ułatwia użytkowanie zarówno dla właściciela, jak i dla chomika. Dostęp do pojemnika na pokarm powinien być wygodny, umożliwiając łatwe napełnianie i wymianę jedzenia.

Na wydrukowane elementy w technologii 3D składają się:

- Stół podtrzymujący całą konstrukcję. Ma wymiary 12cmx12cm, wysokość 10cm. Służy do podniesienia ważnych elementów konstrukcji ponad zasięg chomików aby były dla nich niedostępne. Stół posiada dziurę w centralnej części blatu, przez którą przesypywać ma się karmę.

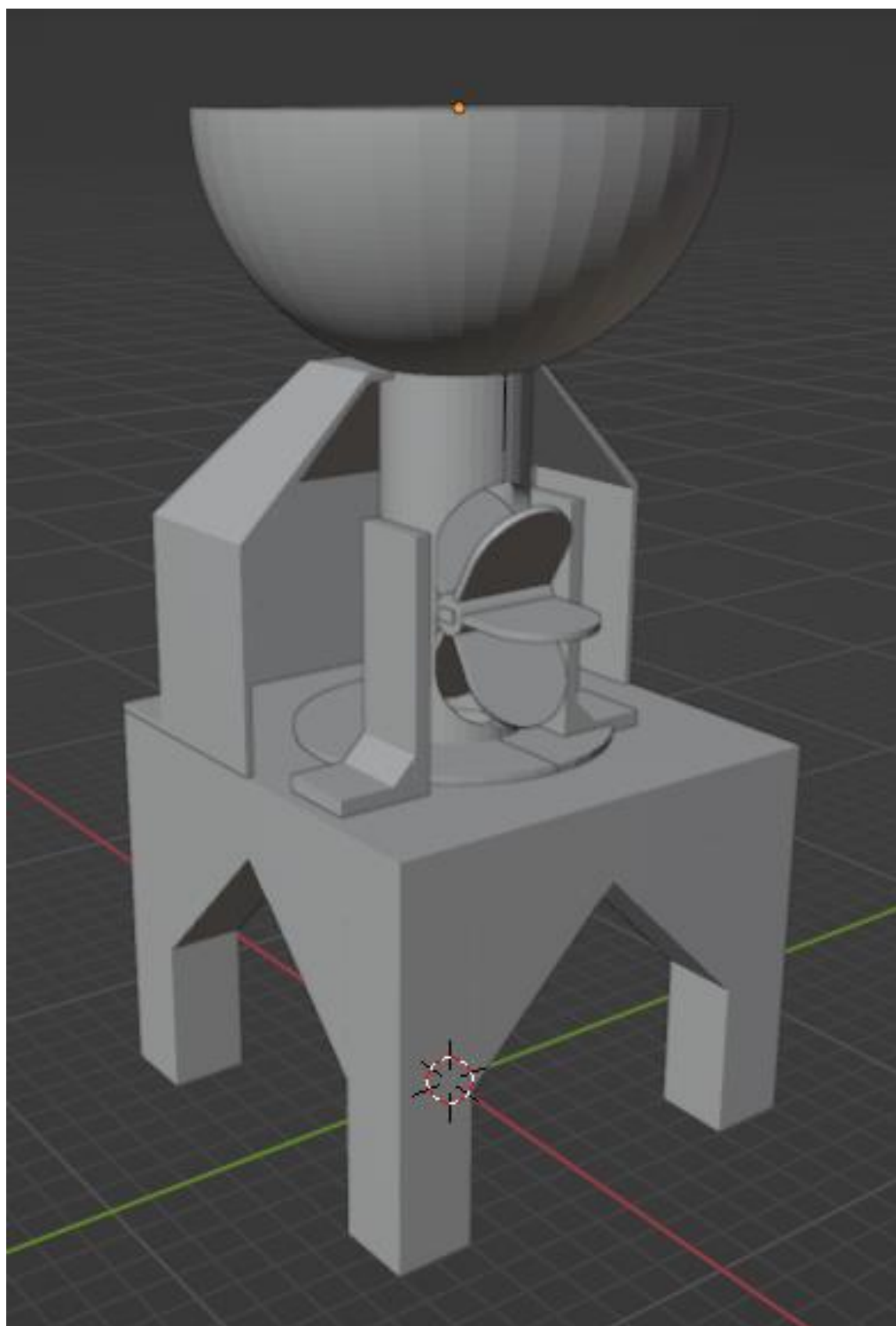


- Mechanizm przesypywania karmy. Składa się z obrotowych łopatek, rurki z wyciętym zasięgiem łopatek oraz elementów podtrzymujących łopatki wraz z silnikiem krokowym. Karma dostaje się do mechanizmu ze zbiornika umieszczonego nad rurką.



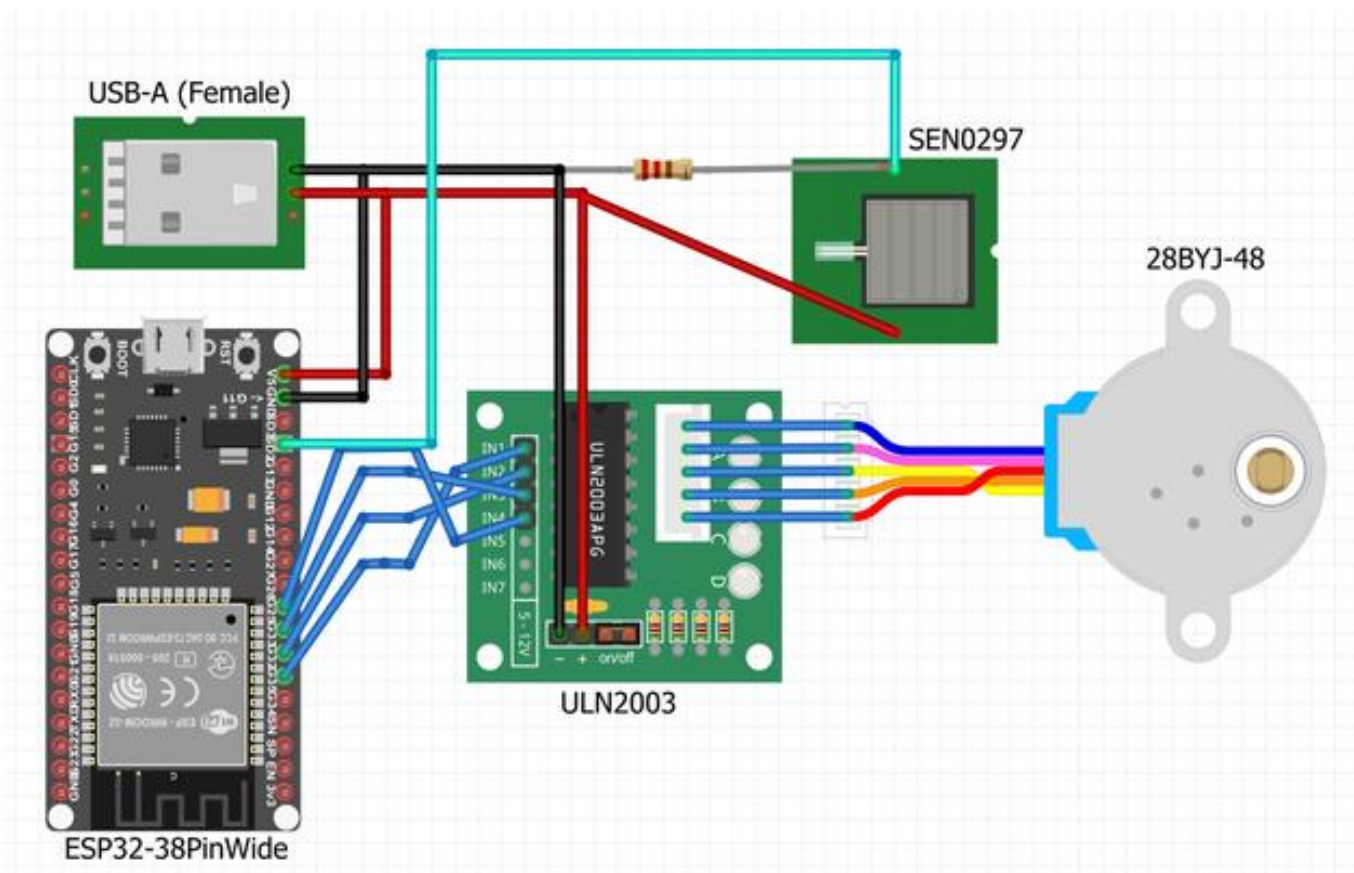
- Zbiornik na karmę zapewniający mechanizmowi przesypującemu dostateczny zapas karmy.
- Obudowa mająca na celu ochronę elementów układu przed chomikami.

Całość zaprojektowanych części karmnika prezentuje się następująco:

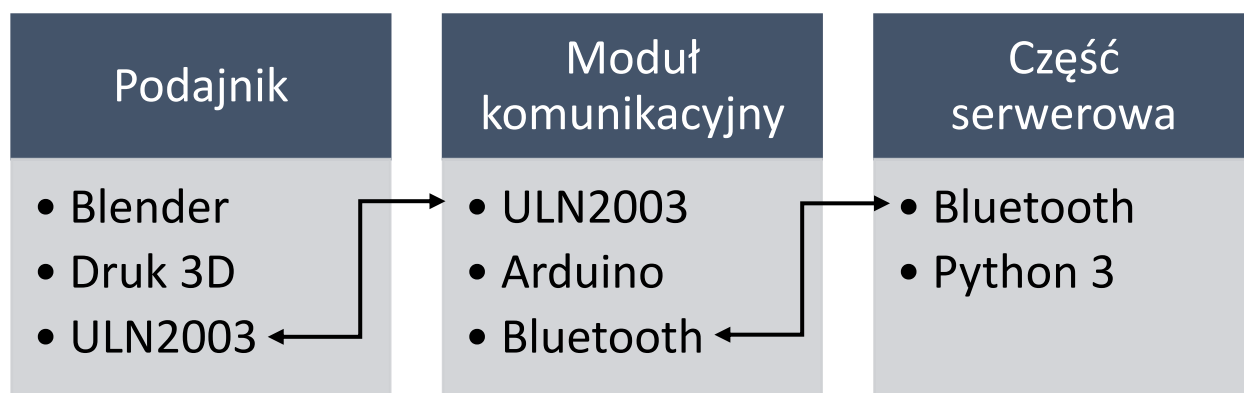


4. Oprogramowanie

Uproszczony schemat działania systemu zawierający najważniejsze moduły widnieje na poniższym schemacie. Oprogramowanie zarządzające całym systemem to samodzielnie napisany skrypt w Arduino.



4.1. Podsumowanie wykorzystywanych technologii



System będzie składa się z trzech części głównych:

- podajnika,
- modułu komunikacyjnego,
- części odpowiedzialnej za logikę dozowania.

Terrarium dla którego przygotowano urządzenie, jest terrarium szklanym składających się z pięciu pełnych ścianach ze szkła, oraz otwartej części górnej z nakładaną "pokrywką", o wymiarach zewnętrznych 35x85x45[cm] (głębokość, szerokość, wysokość) i ściankach o grubości 0.5cm. W terrarium mieszkają dwa chomiki dzungarskie. Karmione są one okrągłym, spłaszczonym granulatem o średnicy około 8mm. Dzielne zużycie karmy na chomika waha się pomiędzy 15, a 20 gramów.

Pojemnik na karmę powinien przechowywać ilość karmy wystarczającą na co najmniej tydzień, a więc około 250 gramów, które zajmuje około 700ml objętości. Podajnik składa się z silnika krokowego, oraz części zaprojektowanych z pomocą programu Blender i wykonanych w technologii druku 3D. Jako dodatkową funkcjonalność urządzenia, dodane zostały czujniki nacisku celem pomiaru ilości pozostałej karmy, a także warunków jej przechowywania.

4.2. Skrypt w Arduino

Arduino to platforma elektroniczna, składająca się z mikrokontrolera oraz środowiska programistycznego, która umożliwia tworzenie interaktywnych projektów i prototypów. Mikrokontroler Arduino jest małym układem elektronicznym, który można programować w celu sterowania różnymi urządzeniami i czujnikami.

Arduino oferuje prosty interfejs programowania, który jest dostępny dla osób bez specjalistycznej wiedzy z dziedziny elektroniki i programowania. Środowisko programistyczne Arduino umożliwia pisanie kodu w języku C/C++, a następnie wgrywanie go na mikrokontroler Arduino za pomocą kabla USB.

Skrypt Arduino, który jest wgrany na moduł komunikacyjny silnika krokowego, jest odpowiedzialny za odbieranie komunikatów z API napisanego w Pythonie i sterowanie silnikiem na podstawie tych komunikatów. Poniżej przedstawiam ogólny opis działania takiego skryptu:

1. Inicjalizacja: Skrypt Arduino rozpoczyna się od inicjalizacji niezbędnych komponentów. Obejmuje to konfigurację portów komunikacyjnych, ustawienie parametrów silnika krokowego i inicjalizację połączenia z API.
2. Odbieranie komunikatów: Arduino czeka na nadejście komunikatów z modułu komunikacyjnego, który otrzymuje je poprzez odpowiedni interfejs (Serial port/Bluetooth).

Komunikaty mogą zawierać informacje dotyczące kierunku ruchu, prędkości, liczby kroków itp., które zostaną przekazane do silnika krokowego.







3. Dekodowanie komunikatów: Po otrzymaniu komunikatu Arduino dekoduje go, aby zrozumieć polecenia wysłane z API napisanego w Pythonie. W zależności od struktury komunikatu i zawartych w nim danych, Arduino podejmuje odpowiednie działania.
4. Sterowanie silnikiem krokowym: Na podstawie odczytanych informacji z komunikatu, Arduino generuje odpowiednie sekwencje impulsów dla silnika krokowego. Impulsy te sterują ruchem silnika, określając kierunek i liczbę kroków do wykonania. Arduino może również kontrolować prędkość silnika, manipulując częstotliwością generowanych impulsów.
5. Odpowiedź na API: Po wykonaniu odpowiednich operacji sterujących silnikiem, Arduino może wysłać odpowiedź z powrotem do API, informując o statusie wykonania danego polecenia. To umożliwia API śledzenie postępu i dostosowanie dalszych działań na podstawie otrzymanych informacji.

```
1  #include "BluetoothSerial.h"
2  #include "Stepper.h"
3
4  #if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
5  #error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
6  #endif
7
8  BluetoothSerial SerialBT;
9  const int stepsPerRevolution = 2037;
10 const int feedStep = stepsPerRevolution / 6;
11 Stepper myStepper = Stepper(stepsPerRevolution, 32, 33, 25, 26);
12 int res;
13 int feedCount;
14
15 void setup()
16 {
17     analogReadResolution(10);
18     myStepper.setSpeed(10);
19     Serial.begin(115200);
20     SerialBT.begin("PetFeeder");
21     Serial.println("The device started, now you can pair it with bluetooth!");
22 }
23
24 void loop()
25 {
26     if (SerialBT.available())
27     {
28         char command = SerialBT.read();
29         Serial.println(command);
30         switch (command)
31         {
32             case 's':
33                 res = analogRead(34) >> 2;
34                 SerialBT.write(res);
35                 Serial.println(res);
36                 break;
37             default:
38                 feedCount = command;
39                 Serial.println(command);
40                 myStepper.step(feedStep * feedCount);
41                 SerialBT.write(0);
42         }
43     }
44     delay(1000);
45 }
```

4.3. API (Python)

Działanie serwera napisanego w Pythonie, który komunikuje się poprzez Bluetooth z modułem komunikacyjnym Arduino obejmuje następujące elementy:

1. Inicjalizacja serwera: Serwer napisany w Pythonie rozpoczyna się od inicjalizacji odpowiednich bibliotek do obsługi komunikacji Bluetooth. Następnie serwer tworzy gniazdo nasłuchujące na określonym porcie Bluetooth, oczekujące na połączenie od modułu komunikacyjnego Arduino.
2. Połączenie z Arduino: Moduł komunikacyjny Arduino, wyposażony w moduł Bluetooth, nawiązuje połączenie z serwerem Pythona poprzez adres Bluetooth. Po nawiązaniu połączenia, serwer i Arduino mogą wymieniać dane.
3. Odbieranie żądań API: Serwer nasłuchuje na gnieździe Bluetooth i oczekuje na żądania API wysyłane przez użytkownika. Żądania API mogą zawierać informacje, takie jak kierunek obrotu, liczba kroków do wykonania lub inna kontrola ruchu dla silnika krokowego.
4. Przetwarzanie żądań: Po otrzymaniu żądania API serwer przetwarza je, analizując dane przesłane przez użytkownika. Może to obejmować walidację żądania, weryfikację poprawności danych i konwersję ich na odpowiednie formaty zrozumiałe dla Arduino.
5. Komunikacja z Arduino: Serwer wysyła przetworzone dane do modułu komunikacyjnego Arduino poprzez połączenie Bluetooth. Dane te mogą zawierać instrukcje dotyczące kierunku obrotu, liczby kroków, prędkości itp.
6. Odbiór danych przez Arduino: Moduł komunikacyjny Arduino odbiera przesłane dane od serwera i przetwarza je na instrukcje dla układu ULN2003. Arduino generuje odpowiednie sekwencje impulsów dla silnika krokowego, kontrolując jego obrót zgodnie z żądaniami API.
7. Odpowiedź serwera: Po wykonaniu żądanej operacji, Arduino może wysłać odpowiedź zwrótną do serwera, potwierdzając wykonanie zadania lub przekazując informacje o stanie silnika. Serwer odbiera odpowiedź i może ją przekazać z powrotem do klienta, który wywołał żądanie API.

 alert.py	init
 bt_api.py	init
 consts.py	init
 pet_feeder_controller.log	init
 pet_feeder_controller.py	init
 pyserial.py	init

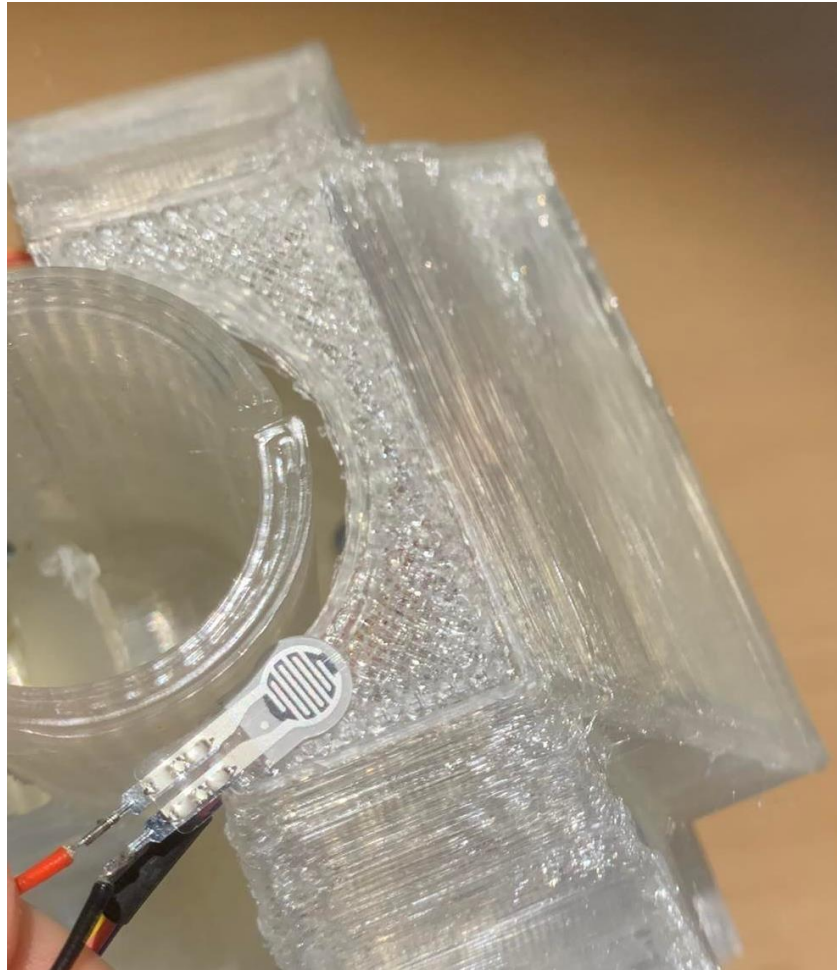
5. Funkcjonalności

5.1. Pomiar nacisku

Pomiar nacisku na zbiorniku z automatycznym karmnikiem dla chomików jest realizowany z wykorzystaniem czujnika nacisku umieszczonego pod zbiornikiem z karmą dla chomików. Jest czujnikiem oporu, który reaguje na zmianę nacisku. Czujnik nacisku jest podłączony do mikrokontrolera poprzez wejście analogowe i jako element dzielnika napięciowego. Mikrokontroler służy jako pośrednik między czujnikiem a automatem karmnika.

Przed umieszczeniem karmnika z chomikami na czujniku, przeprowadzono proces kalibracji. Polega on na określeniu punktu odniesienia, tj. wartości nacisku, która odpowiada pustemu zbiornikowi. Kalibracja pozwala na dokładne mierzenie zmiany nacisku w zależności od ilości karmy w zbiorniku. Mikrokontroler odczytuje dane z czujnika nacisku, które są zwykle w postaci sygnału.

Na podstawie odczytanej wartości nacisku, mikrokontroler może podjąć odpowiednie działania. Na przykład, gdy ilość karmy w zbiorniku spada poniżej ustalonego progu, mikrokontroler może aktywować mechanizm dozujący, który dostarczy odpowiednią ilość karmy do zbiornika. W przeciwnym razie, jeśli zbiornik jest pełny, mikrokontroler może zignorować sygnał nacisku.

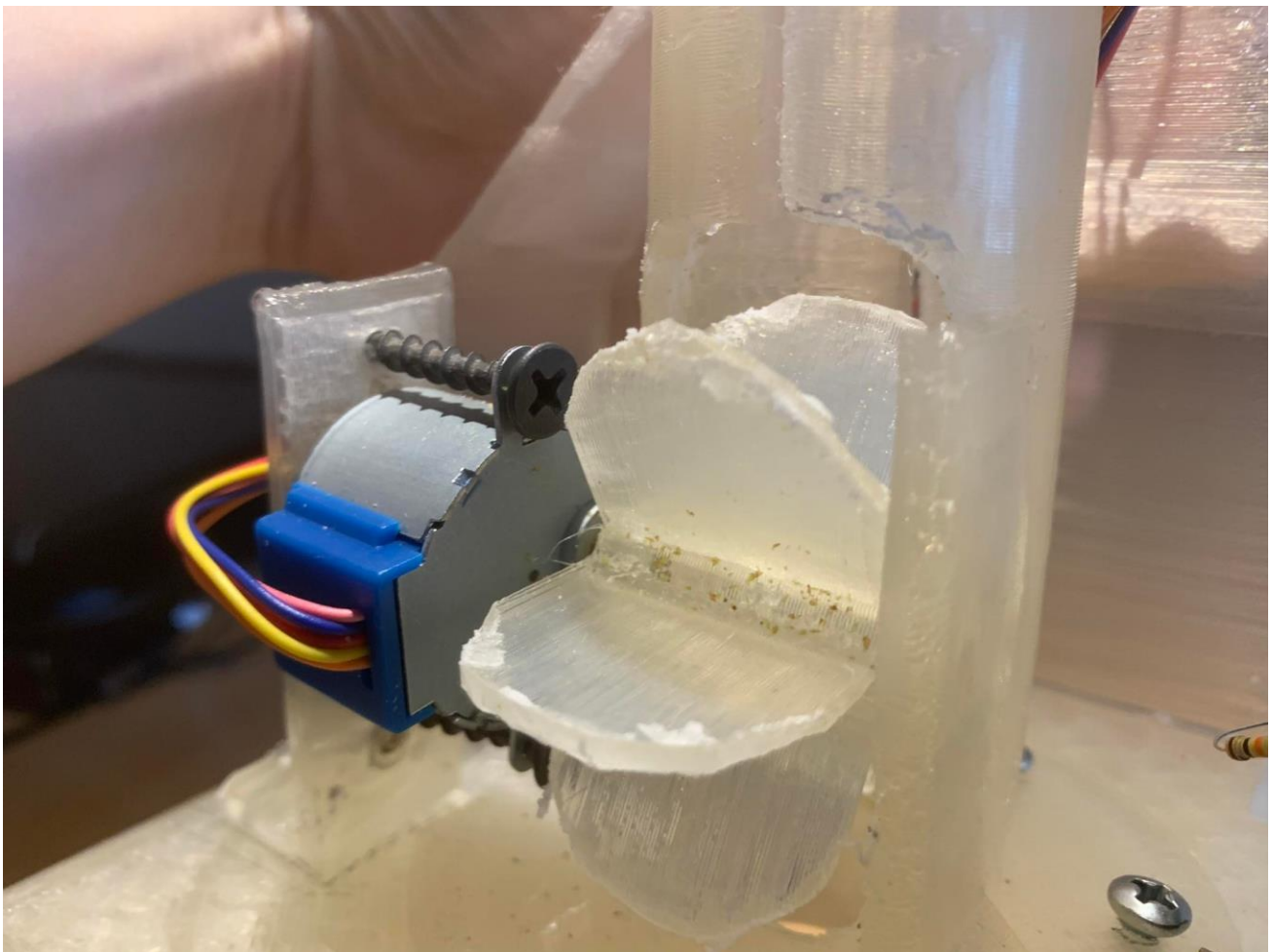


5.2. Przesypywanie karmy

Proces przesypywania się karmy w karmniku za pomocą mechanizmu z łopatkami sterowanego przez silnik krokowy przebiega w następujący sposób:

1. Konstrukcja mechanizmu: W inteligentnym karmniku znajduje się specjalny mechanizm składający się z łopatek, które służą do przesypywania karmy z zbiornika do miski. Łopatki są tak zaprojektowane, aby mogły obracać się wokół osi i przenosić karmę w odpowiednie miejsce.
2. Podłączenie silnika krokowego: Silnik krokowy jest połączony z mechanizmem łopatek. Jest on osadzony bezpośrednio na osi mechanizmu. Silnik krokowy umożliwia precyzyjne sterowanie obracaniem łopatek o określone kąty.
3. Sterowanie silnikiem krokowym: Do sterowania silnikiem krokowym służy komunikacyjny Arduino. Za pomocą odpowiedniego sterownika silnika krokowego, generowane są impulsy sterujące, które precyzyjnie określają kroki obracania się silnika.

4. Sterowanie czasowe lub na żądanie: Inteligentny karmnik działa na zasadzie ustalonego harmonogramu czasowego oraz dodatkowo na żądanie użytkownika. Mikrokontroler został skonfigurowany tak, aby uruchamiał proces przesypywania karmy w określonych interwałach czasowych lub na podstawie sygnałów przychodzących z interfejsu użytkownika.
5. Kontrola ilości karmy: W oprogramowaniu mikrokontrolera ustalono, że każde obroty łopatek będą przesypywać określoną ilość karmy do miski. Obrót o jedną wielokrotność 60 stopni odpowiada jednemu porcji karmy. W zależności od potrzeb i preferencji można dostosować wielkość porcji poprzez regulację liczby kroków obrót silnika.



```

while True:
    if last_timestamp is None or datetime.now() > last_timestamp + timedelta(hours=consts.FEEDING_INTERVAL_HOURS):
        print(f"{datetime.now().isoformat()} Feeding started")
        log_file.write(f"{datetime.now().isoformat()} Feeding started")
        log_file.flush()
        start_timestamp = datetime.now(tz=pytz.timezone("Europe/Warsaw"))

        food_left_at_start = bt_serial.get_food_left_in_grams()
        food_left = food_left_at_start
        while food_left_at_start - food_left <= consts.GRAMS_PER_FEEDING and not bt_serial.is_empty():
            bt_serial.rotate_stepper(1)
            food_left = bt_serial.get_food_left_in_grams()
        if bt_serial.is_empty():
            alert.empty_warning()
        elif bt_serial.get_food_left_in_grams <= consts.ALERT_LEVEL_GRAMS:
            alert.low_warning()

        seconds_passed = (datetime.now(tz=pytz.timezone("Europe/Warsaw")) - start_timestamp).total_seconds()
        log_file.write(f"Time passed: {seconds_passed}\n")
        log_file.write(f'{datetime.now().isoformat()}')
        print(f"Time passed: {seconds_passed}s\n")
        log_file.flush()

    sleep(consts.SCRIPT_INTERVAL_SECONDS)

```

5.3. Powiadomienie o niskim poziomie karmy w zbiorniku

System powiadamiania o niskim poziomie karmy w zbiorniku inteligentnego karmnika dla chomików może być zrealizowany na różne sposoby, ale zdecydowano się na powiadomienia w części

1. Karmnik posiada moduł komunikacji bezprzewodowej, dzięki któremu można było skonfigurować system powiadomień. Użytkownik aplikacji w Pythonie otrzymuje powiadomienie, kiedy poziom karmy spadnie poniżej ustalonego poziomu.
2. Karmnik umożliwia zdalny monitoring stanu karmy w czasie rzeczywistym. Za pomocą interfejsu, właściciel karmnika może sprawdzać aktualny poziom karmy w zbiorniku bezpośrednio na swoim urządzeniu.

```

1  def low_warning():
2      print("Low food level!")
3
4  def empty_warning():
5      print("Food container is empty!")

```


5.4. Zdalne sterowanie mechanizmem

Zdalne sterowanie mechanizmem przesypującym karmę za pomocą API w Pythonie może być zrealizowane w kilku krokach. Oto opis ogólnego procesu:

1. Komunikacja z mechanizmem: W implementacji obsługi żądań skonfigurowano połączenie z mechanizmem przesypującym karmę. Obejmuje to użycie biblioteki pySerial, która umożliwia komunikację z urządzeniami szeregowymi, takimi jak Arduino.
2. Przesyłanie komend do mechanizmu: Na podstawie komunikatów bluetooth, można przekazać odpowiednie komendy do mechanizmu sterującego. Na przykład, przy otrzymaniu żądania przesunięcia o określoną ilość kroków, należy przesłać tę informację do mechanizmu poprzez odpowiednią komunikację szeregową.
3. Obsługa odpowiedzi: W implementacji API należy uwzględniona została obsługa odpowiedzi. API może zwrócić odpowiedź z potwierdzeniem wykonania żądania lub zwrócić odpowiedni kod błędu w przypadku niepowodzenia.

