Asystent głosowy do zastosowań w inteligentnych budynkach

METODYKI MODELOWANIA I PROJEKTOWANIA SYSTEMÓW

Dawid Ilba (296481), Dominik Starzyk (296513)
Elektronika i telekomunikacja 1 rok, II stopień
Wydział Informatyki Elektroniki i Telekomunikacji
Akademia Górniczo-Hutnicza
Kraków 2021

Spis treści

1. Wstęp

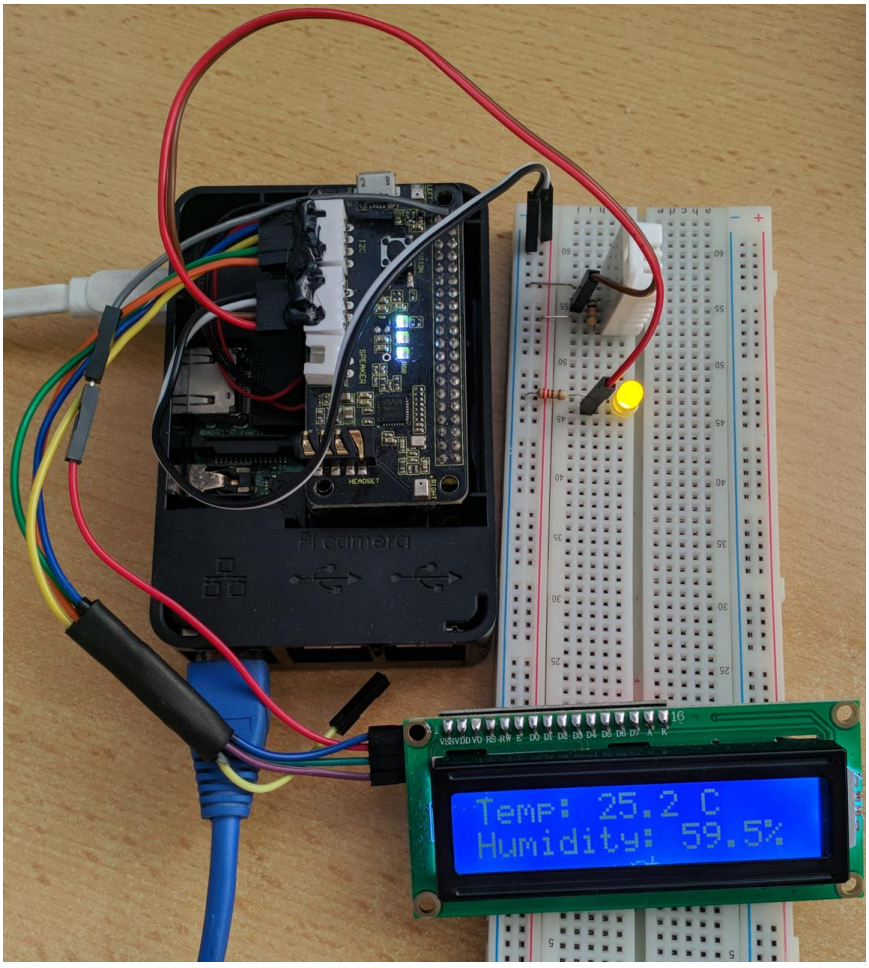
Asystent głosowy to narzędzie, które jest w stanie wykonać określone czynności poprzez wypowiedzenie odpowiedniej komendy. Jest w stanie znacznie ułatwić życie, poprawić komfort i wygodę życia, a także zaoszczędzić czas. Zalet takiego urządzenia jest mnóstwo, dlatego powinno ono znaleźć zastosowanie w większości inteligentnych budynków.

Przeglądając urządzenia dostępne na rynku zauważyliśmy, że większość z nich do poprawnej pracy wymaga dostępu do internetu. Uznaliśmy, że w naszym projekcie wyeliminowanie tej wady, będzie bardzo wielką zaletą. Zdecydowaliśmy przedstawić podstawowe funkcje takiego urządzenia. Do projektu wykorzystaliśmy minikomputer Raspberry Pi 3B, moduł (shield) Respeakera 2-Mic Pi Hat V1.0, wyświetlacz LCD HD44780 z interfejsem i2c, czujnik temperatury i wilgotności DHT22 oraz diodę LED.

Respeaker to urządzenie, które posiada zestaw bardzo czułych mikrofonów, przystosowanych do rozpoznawania mowy. Jest rozwiązaniem zdecydowanie lepszym niż wykorzystanie karty USB i standardowego mikrofonu. Na początku nie mieliśmy w ogóle testować takiego rozwiązania i polegać na opiniach innych konstruktorów narzędzia do rozpoznawania mowy, jednak mieliśmy problem z terminową dostawą Respeakera. Niestety w momencie składania zamówienia, nie znaleźliśmy żadnego modułu (w rozsądnej cenie) dostępnego do kupienia w Polsce i zamówiliśmy układ z Chin. Wpłynęło to w dość dużym stopniu na terminowość naszego projektu, i żeby nie tracić czasu przetestowaliśmy rozpoznawanie mowy z wykorzystaniem karty USB i mikrofonu. Skuteczność rozpoznawania była niska, a odległość mówcy od mikrofonu musiała być bardzo mała. Dodatkowo straciliśmy mnóstwo czasu na poprawną konfigurację wszystkiego na Raspberry. Zastosowania shieldu Respeakera poprawiło skuteczność rozpoznawania mowy, a w szczególności zwiększyło odległość mówcy od mikrofonu. Dodatkowo konfiguracja jest zdecydowanie prostsza i szybsza.

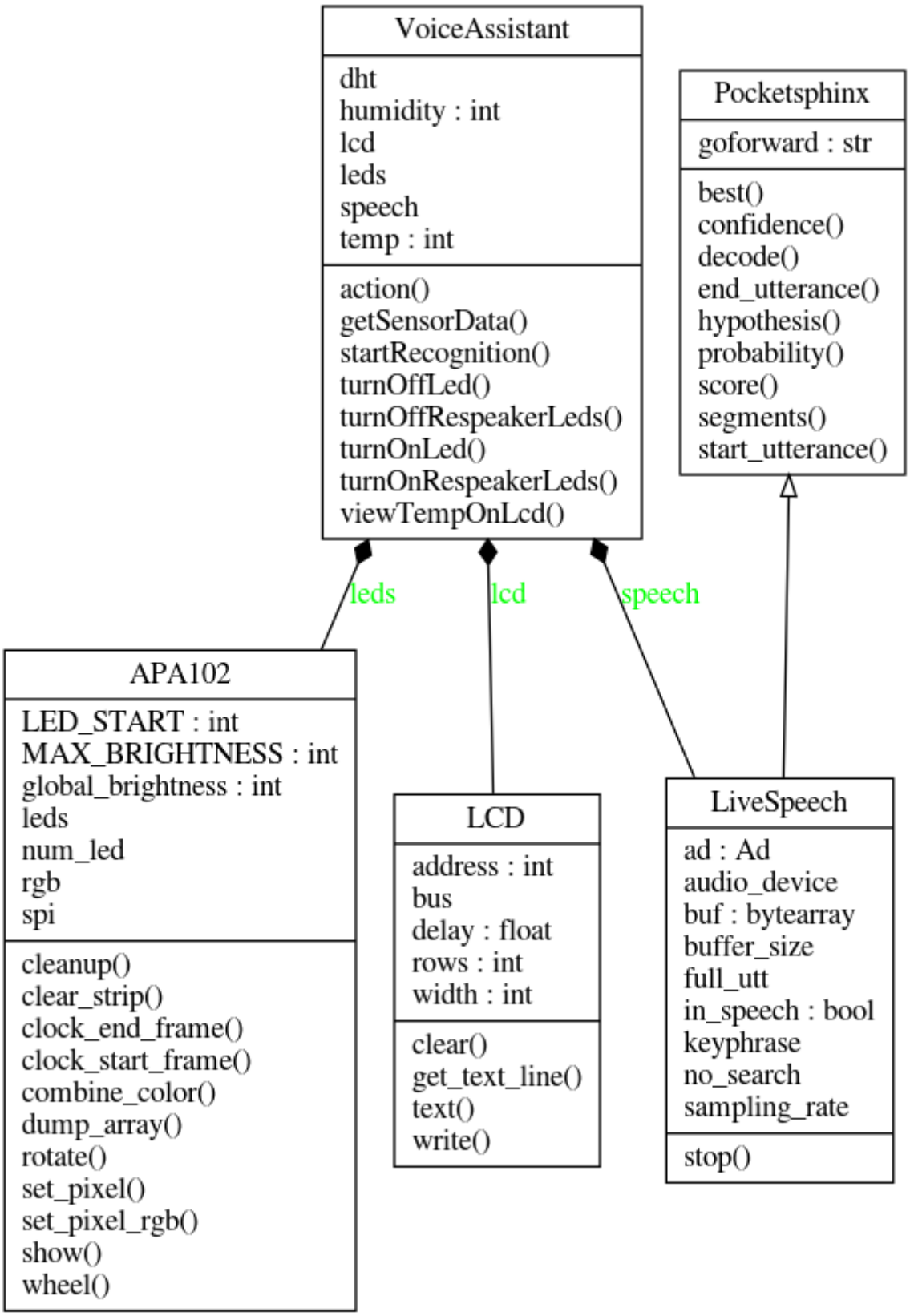
2. Założenia projektu

- Projekt „asystenta głosowego” z wykorzystaniem platformy Raspberry Pi
- Przetwarzanie mowy w języku angielskim na tekst
- Dodatkowo, żeby zwiększyć atrakcyjność naszego projektu zdecydowaliśmy się, aby przetwarzanie mowy odbywało się bez dostępu do internetu
- Stworzenie zakresu komend ilustrujących przykładowe wykorzystanie asystenta głosowego w inteligentnym budynku
- Użytkownik głosowo będzie mógł obsługiwać takie peryferia jak : czujnik wilgotności oraz temperatury, wyświetlacz LCD (interfejs I2C), na którym pojawiać się będą rozpoznawane słowa, a także informacje pobrane z czujnika, diodę LED zamodelowaną w projekcie jako „lampę”, 3 diody RGB połączone interfejsem SPI występujące na pokładzie modułu Respeakera
- Język programowania Python
- Metodologia pracy SCRUM – szczegóły znajdują się w pliku „SCRUM.xls” zamieszczonego w repozytorium



Rys 2.1 Wygląd opracowanego asystenta głosowego

3. Diagram klas

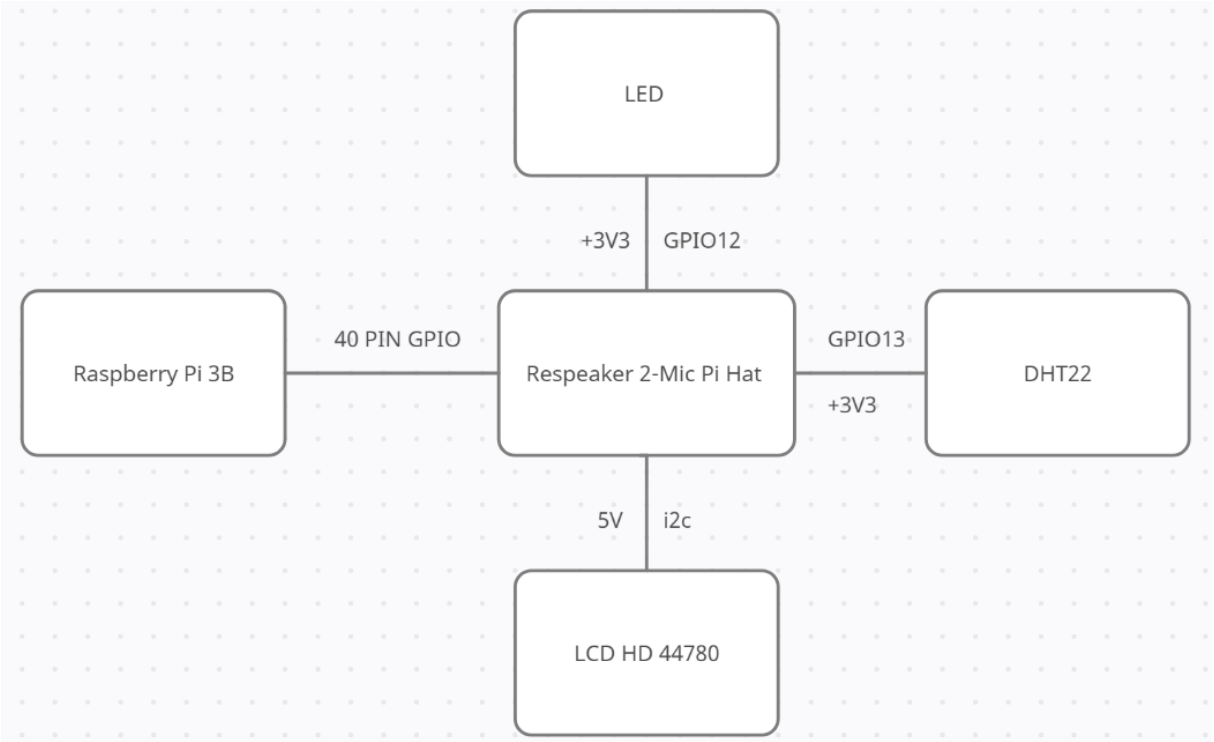


4. Instrukcja

4.1 Podłączenie i konieczne biblioteki

Schemat poprawnego podłączenia wszystkich komponentów wykorzystanych w projekcie został przedstawiony na poniższym rysunku. Dodatkowo na minikomputerze Raspberry Pi 3B konieczne jest posiadanie następujących bibliotek i sterowników:

- System operacyjny Raspbian
- Sterowniki dla modułu Respeakera firmy Seeed model ReSpeaker 2-Mics Pi Hat V1.0. Poradnik dostępny jest na stronie producenta[1].
- Biblioteka CMU **pocketsphinx**, do rozpoznawania mowy offline. [2]
- Biblioteka **rpi_lcd** dla wyświetlacza LCD HD44780 i2c [3]
- Biblioteka **RPi.GPIO** dla układów peryferyjnych [4]
- Biblioteka **board [5]** oraz **adafruit_dht [6]** dla poprawnego działania czujnika wilgotności i temperatury DHT22.



Rys 4.1 Schemat blokowy

4.2 Obsługa i wygląd

Napisany przez nas skrypt w Pythonie ma ogromne możliwości rozwoju, a projekt prezentuje jedynie mały odsetek możliwości opracowanego asystenta głosowego. Rozbudowa systemu odbywa się poprzez proste dodanie instrukcji warunkowych „if" w klasie „VoiceAssistant" w metodzie „action". Dodatkowo konieczne jest uwzględnienie nowych słów w pliku „key.list"., aby system potrafił je rozpoznawać. Jako, że w projekcie wykorzystujemy przetwarzanie mowy na tekst w formie offline, dokładność takiego systemu jest zdecydowanie mniejsza niż narzędzi online (np. asystent głosowy Google). W pracy zdecydowaliśmy się wykorzystać darmową open source'ową bibliotekę CMU Sphinx (pocketsphinx). Oprogramowanie to korzysta z wyuczonych wcześniej modeli językowych i akustycznych. Niestety w swojej bazie nie zawiera ona języka polskiego. Oczywiście możliwe jest stworzenie takiego modelu od podstaw, jednak potrzebna jest ogromna liczba danych treningowych (nagrania głosowe wielu mówców) oraz poprawnie przygotowany słownik fonetyczny. Stworzenie skutecznego modelu językowego asystenta głosowego rozpoznającego mowę offline zajęło by przynajmniej kilka miesięcy. My zdecydowaliśmy się na wykorzystanie angielskiego modelu językowego z ograniczeniem słownika do wyrazów które ma on rozpoznawać. Takie rozwiązanie daje znaczną poprawę skuteczności poprawnego rozpoznawania mowy. Możliwe są także inne sposoby na poprawę jakości przetwarzania mowy, np. jeżeli system będzie wykorzystywał jeden i ten sam mówca, może on dodać nagrania słów wypowiadanych przez siebie, co zagwarantuje najlepsze rezultaty.

Przetwarzanie mowy niewymagające dostępu do internetu jest czynnikiem zdecydowanie wyróżniającym się na tle komercyjnych rozwiązań, który my zdecydowaliśmy się wykorzystać w projekcie. Poza wieloma zaletami, niestety skutecznością nie dorównuje on komercyjnym rozwiązaniom, gdzie za minimalny próg błędów przetwarzania przyjmuje się 5%.

Opracowany przez nas skrypt rozpoznaje następujące komendy:

- „Lights on/off" – włączenie/wyłączenie 3 diod RGB Respeakera połączonych interfejsem SPI. Diody świecą w kolorze białym
- „Lamp on/off" – włączenie/wyłączenie diody LED
- „Show temperature" lub „show humidity" – wyświetlenie na wyświetlaczu danych pobranych z czujnika DHT22.
- „Set light color to red" – ustawienie koloru diod RGB na kolor czerwony
- „Set light color to green" – ustawienie koloru diod RGB na kolor zielony
- „Set light color to blue" – ustawienie koloru diod RGB na kolor niebieski
- Na wyświetlaczu poza komendą „show temperature" oraz „show humidity" zawsze pokazuje się rozpoznana fraza. Dzięki temu możemy łatwo zweryfikować, że system się pomylił i ponownie wypowiadamy komendę

Asystent głosowy jest w stanie rozpoznawać multum komend, które tylko użytkownik zdefiniuje, a dodanie tych komend zostało wyjaśnione na początku tego podrozdziału.

5. Prezentacja projektu – linki

Link do głównego repozytorium :

https://github.com/dawidilba/voice-assistant-for-smart-buildings/

Link do filmiku przedstawiającego działanie projektu:

https://www.youtube.com/watch?v=_gPBFD0bUVg

6. Bibliografia

[1] Strona producenta modułu Respeakera

https://wiki.seeedstudio.com/ReSpeaker_2_Mics_Pi_HAT/

[2] Biblioteka do przetwarzania mowy na tekst pocketsphinx

https://pypi.org/project/pocketsphinx/

[3] Biblioteka wyświetlacza LCD HD44780 i2c

https://pypi.org/project/rpi-lcd/

[4] Biblioteka dla układów peryferyjnych

https://pypi.org/project/RPi.GPIO/

[5] Biblioteka board

https://pypi.org/project/board/

[6] Biblioteka adafruit_dht dla czujnika temperatury i wilgotności

https://github.com/adafruit/Adafruit_Python_DHT