KATEDRA INFORMATYKI
Wydział IEiT AGH

AGH

AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA
W KRAKOWIE

Projekt Systemy Wbudowane 2016 pt. “Prosty system monitoringu na platformie Raspberry Pi”

Dokumentacja techniczna

Karol Kozak
Marcin Lasoń

Spis treści

I. Dokumentacja techniczna

II. Dokumentacja użytkownika

I. Dokumentacja techniczna

1. Wstęp

1.1. Opis projektu

Projekt realizowany jest na platformie Raspberry PI. Prace wykonywane są na urządzeniu wersji 2, lecz założeniem projektu jest pełna kompatybilność z najnowszą wersją 3. Głównym celem jest prawidłowa obsługa kamery Raspberry Pi Camera HD v2 8MPx oraz modułu Bluetooth. Projekt zakłada kilka funkcjonalności:

- możliwość robienia zdjęć oraz filmów za pomocą zainstalowanego na platformie przełącznika typu Tact Switch,
- możliwość robienia zdjęć oraz filmów za pomocą prostej aplikacji mobilnej, komunikującej się z platformą poprzez Bluetooth,
- możliwość wprowadzenia platformy w tryb wykonywania zdjęć przy wykryciu ruchu (wykorzystany zostanie czujnik ruchu PIR HC-SR501),
- dodanie modułu wykrywającego twarze na zrobionych zdjęciach.

1.2. Potrzebne elementy

Do zbudowania projektu zostały wykorzystane następujące elementy:

- minikomputer Raspberry Pi w wersji 2.,
- zasilacz microUSB 5V 2A,
- karta microSDHC 16GB,
- Raspberry Pi Camera HD v2 8MPx,
- Moduł Bluetooth,
- płytka stykowa,
- przewody żeńsko-męskie, męsko-męskie, żeńsko-żeńskie,
- diody LED 5mm,
- rezystory THT 1/4 W wartości: od 47Ω,
- czujnik ruchu PIR HC-SR501,
- Tact Switch 6x6, 4.3mm THT,
- moduł Bluetooth USB (dla Raspberry Pi2),
- ...

1.3. Szczegółowy opis projektu

Podstawową funkcjonalnością jest wykonywanie zdjęć oraz filmów używając przełącznika Tact Switch. Za jego pomocą użytkownik będzie mógł kręcić dowolnej długości filmy (przytrzymanie przełącznika) lub wykonać zdjęcie (przyciśnięcie przełącznika). Pliki zostaną zapisane w pamięci urządzenia w postaci [photo/video]_rok_miesiąc_dzień_godzina:minuta:sekunda.[jpg/h264].

Następną rzeczą jest obsługa platformy z poziomu aplikacji mobilnej dzięki komunikacji poprzez Bluetooth. Rozbudowa aplikacji nie jest głównym celem projektu, dlatego zakłada minimalną funkcjonalność, zbliżoną do Tact Switcha, tzn. wykonanie pojedynczego zdjęcia lub nakręcenie filmu. Długość filmu również zależna będzie od tego jak długo użytkownik dotykać będzie przycisku nagrywania. Głównym zadaniem tego elementu projektu jest wykorzystanie modułu Bluetooth w Raspberry PI.

Aplikacja przewiduje także trzecią możliwość, czyli wprowadzenie platformy w tryb wykrywania ruchu. Bardzo czuły moduł PIR HC-SR501 pozwoli nam na wykrywanie nawet najmniejszych zmian w otoczeniu kamery. Gdy takowy zostanie wykryty kamera wykona zdjęcie. Jest to funkcjonalność, która ma sprawić, że platforma będzie mogła stanowić bardzo prymitywną wersję monitoringu. Wysoka jakość kamery pozwoliłaby na wykonanie zdjęcia, na którym zdecydowanie można by rozpoznać sprawcę np. włamania do mieszkania. Niewielkie rozmiary kamery powodują, że można ją świetnie ukryć.

Aby zwiększyć użyteczność platformy jako narzędzia do monitoringu zaimplementowany zostanie moduł wykrywania twarzy. Do tego celu użyta zostanie istniejąca już biblioteka open source dostępna w języku Python. Jej możliwość pozwala na znalezienie twarzy na wykonanym już zdjęciu, dlatego finalnym działaniem systemu będzie wykonywanie zdjęć, gdy tylko czujnik ruchu wykryje jakiś ruch. Zdjęcie to będzie od razu analizowane, w przypadku, gdy nie znajduje się na nim dobrze widoczna twarz będzie ono usuwane. W przeciwnym razie zdjęcie zostanie umieszczone w specjalnym folderze.

2. Podstawowa konfiguracja

2.1 Przygotowanie nośnika z systemem operacyjnym

Pierwszą rzeczą jaką musimy wykonać jest przygotowanie systemu operacyjnego na karcie microSDHC. Jest on możliwy do pobrania z oficjalnej strony Raspberry Pi: https://www.raspberrypi.org/downloads/raspbian/. Znajduje się tam zawsze najświeższa wersja systemu operacyjnego. W trakcie wykonywania projektu aktualną wersją był RASPBIAN JESSIE WITH PIXEL (wersja jądra 4.4), który ukazał się 23.09.2016. Pobieramy ze strony obraz, następnie wypakowujemy z archiwum obraz .iso. Pamiętajmy, by użyta karta microSDHC miała pojemność minimum 16GB (RASPBIAN JESSIE WITH PIXEL ma ok. 4GB, jednak ze względu na używaną w dalszej części projektu bibliotekę openCV karta o pojemności mniejszej niż 16GB jest niewystarczająca). Na stronie znajdują się także instrukcje tego, jak zapisać na

karcie microSDHC obraz systemu Raspbian. My wykorzystaliśmy do tego komputer z systemem operacyjnym Windows oraz program Win32 Disk Imager.
Kolejne czynności pokazują jak to zrobić:

1. Uruchamiamy program Win32 Disk Imager. W polu „Plik Obrazu” wskazujemy obraz, który uprzednio został wypakowany z archiwum .zip.
2. Wkładamy kartę microSDHC do czytnika kart pamięci, sprawdzamy pod jaką literą karta jest widoczna w systemie. Wskazujemy tę literę w programie Win32 Disk Imager.
3. Klikamy w “Zapisz”, zatwierdzamy komunikat przyciskiem “OK”.
4. Czekamy na zakończenie operacji.

2.2. Uruchomienie systemu na platformie Raspberry Pi

Po wykonaniu poprzedniego kroku, możemy przejść do uruchomienia systemu na Raspberry Pi. Wkładamy kartę microSDHC do slotu na odwrocie minikomputera, następnie podłączamy monitor/telewizor, klawiaturę, myszkę, interfejs Ethernet lub kartę sieciową USB (w przypadku Raspberry Pi 3 nie jest ona potrzebna, gdyż platforma posiada wbudowany odbiornik WI-FI). Na końcu podłączamy zasilanie.
Na ekranie zobaczymy startujący system operacyjny. Początkowo Raspbian jest skonfigurowany tak, że startuje od razu do wersji desktopowej bez żądania hasła. Możemy tak ustawić, aby hasło przy uruchamianiu było wymagane oraz aby system nie uruchamiał GUI. Możemy tego dokonać po wpisaniu do terminala polecenia:

~$ sudo raspi-config

Dane do logowania przy starcie systemu:

login: pi
pass: raspberry

Do przejścia z wiersza poleceń do wersji desktopowej służy polecenie:

~$ startx

Natomiast, aby wyjść z wersji desktopowej:

~$ pkill x

Pierwszą rzeczą jaką powinniśmy zrobić jest rozszerzenie systemu plików na całą kartę microSDHC. Uruchamiamy w terminalu tryb konfiguracji (sudo raspi-config), a następnie wybieramy pierwszą opcję:

Expand filesystem. Ensures that all of the SD card storage is available to the OS

Po wprowadzeniu zmian system zostanie uruchomiony ponownie.

2.3. Aktualizacja systemu - w przypadku starszych wersji Raspbiana

Jeśli posiadamy na karcie już system Raspbian w starszej wersji, możemy go w prosty sposób zaktualizować do najnowszej wersji. Pamiętajmy jednak, że nakładka PIXEL wymaga karty minimum 8GB (starsze wersje Raspbiana mogły być spokojnie umieszczone na karcie o pojemności 4GB). Do zrealizowania tego zadania posłużą nam polecenia:

~$ sudo apt-get update
~$ sudo apt-get dist-upgrade
~$ sudo apt-get install -y rpi-chromium-mods
~$ sudo apt-get install -y python-sense-emu python3-sense-emu
~$ sudo apt-get install -y python-sense-emu-doc realvnc-vnc-viewer
~$ sudo apt-get install -y realvnc-vnc-server
~$ sudo reboot

Powyższe polecenia zaktualizują nam Raspbiana do najnowszej wersji. Otrzymamy nakładkę PIXEL, podstawową przeglądarkę Chromium, serwer VNC. Ostatnim poleceniem uruchomimy ponownie system.

2.4. Podłączenie i podstawowa obsługa dedykowanej kamery

Na tym etapie system jest już gotowy do podłączenia kamery. Przy wyłączonym zasilaniu podpinamy taśmę połączeniową od kamery do portu kamery CSI na płytce Raspberry Pi. Następnie uruchamiamy minikomputer. Przechodzimy w terminalu do trybu konfiguracji urządzenia (sudo raspi-config) i włączamy obsługę kamery opcją:

Enable Camera. Enable this Pi to work with the Raspberry Pi Camera

Po każdej zmianie konfiguracyjnej nasze Raspberry Pi zostanie uruchomione ponownie. Przejdźmy teraz do podstawowej obsługi kamery.

Do robienia zdjęć służy polecenie: **raspistill.**
Po wpisaniu w konsoli powyższego polecenia bez żadnych argumentów pokaże nam listę dostępnych opcji. Ponadto w pierwszej wypisanej linii dostaniemy informację o podpiętej kamerze, pod warunkiem, że została dobrze podpięta. W przeciwnym razie jest to informacja dla nas, by sprawdzić połączenie.

Podstawowym poleceniem, aby wykonać zdjęcie jest:

~$ raspistill -o picture_name.jpg

Parametr ‘-o’ ustawia nazwę, pod którą zostanie zapisane zdjęcie. W przypadku gdyby zdjęcie zostało odwrócone (efekt odwrócenia płytki kamery), możemy to skorygować odwracając obraz horyzontalnie i wertykalnie:

~$ raspistill -hf -vf -o picture_name.jpg

Powyższe polecenia wykonują zdjęcie w najwyższej rozdzielczości wspieraną przez kamerę. Można zmienić rozdzielczość jak też inne parametry robionego zdjęcia. Wszystkie dostępne opcje zobaczymy po wykonaniu polecenia:

~$ raspistill

Przejdźmy teraz do opcji nagrywania filmów. Podobnie jak z poprzednim poleceniem wpisanie poniższego bez podania argumentów wyświetli nam listę dostępnych opcji:

~$ raspivid

Podstawowym poleceniem wykonania filmu jest:

~$ raspivid -o video_name.h264

Domyślną długością kręconego filmu jest 5 sekund.

W obu przypadkach obraz, zarówno zrobione zdjęcie jak i nakręcony film, jeśli podamy tylko nazwę przy parametrze **-o** zostanie zapisany w bieżącym katalogu. Możemy jednak podać ścieżkę docelową, w której zostanie zapisany obraz.

2.5. Prosty skrypt, pozwalający na wykonywanie zdjęć

Stwórzmy skrypt ułatwiający robienie zdjęć. Zadbamy o to, by nazwa robionego zdjęcia była unikalna, zapobiegnie nam to nadpisywaniu wcześniej wykonanych. Przykładowo zapiszmy go pod nazwa: **photo.sh**

```
#!/bin/bash
DATE=$(date +"%Y-%m-%d_%H:%M:%S")
raspistill -o /home/pi/desktop/camera/$DATE.jpg
```

Nazwą naszego zdjęcia będzie data i godzina, co zapewni nam unikalność nazw. Dokładność wynosi co do sekundy. Musimy zadbać o to, by ścieżka do zapisywania zdjęć istniała. Możemy też wprowadzić modyfikację, by docelowa ścieżka była przekazywana przez parametr wykonywanego skryptu:

```
#!/bin/bash
if [ $# -eq 1 ]
then
   PATH=$1
else
   PATH="./"
DATE=$(date +"%Y-%m-%d_%H:%M:%S")
raspistill -o $PATH/$DATE.jpg
```

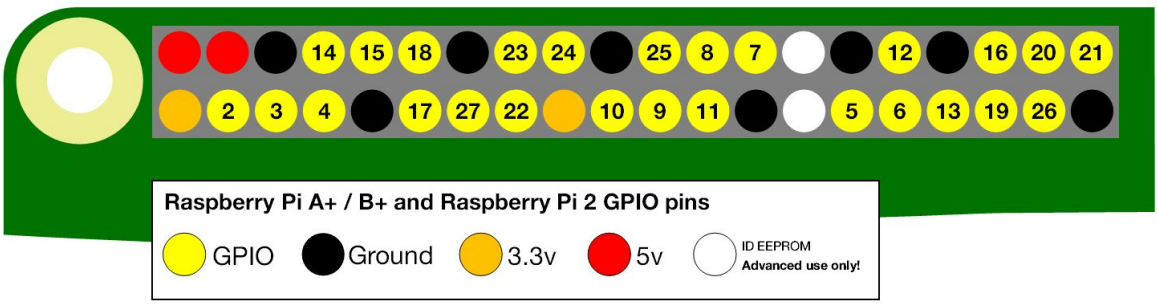
Powyższy skrypt wykonujemy w konsoli poleceniem

```
~$ photo.sh path_to_folder
```

3. Złącze GPIO i podłączenia

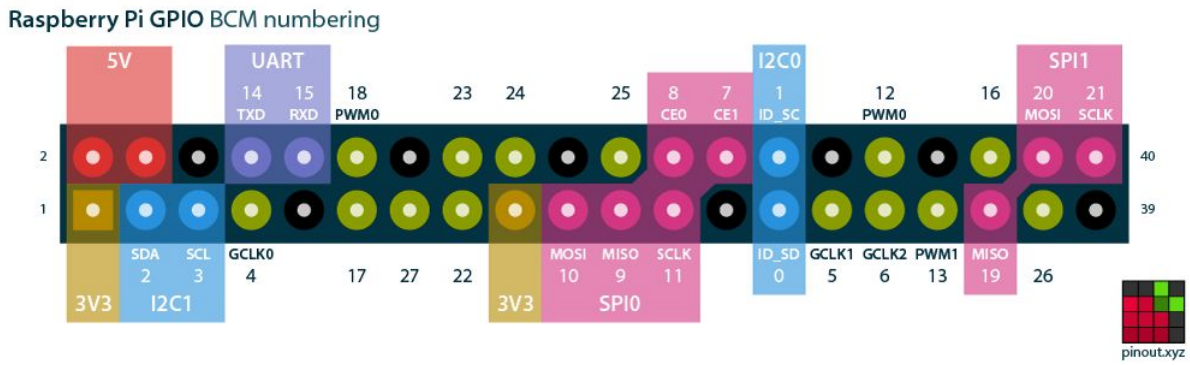
3.1. Złącze GPIO na Raspberry Pi

Raspberry Pi udostępnia nam interfejs do komunikacji w postaci pinów GPIO. Ich liczba to 40 i zarówno w Raspberry Pi 2 jak i Raspberry Pi 3 ich poszczególna funkcjonalność jest taka sama. Obrazek poniżej pokazuje za co odpowiada poszczególny pin.



źródło: raspberrypi.org

Powyższa numeracja jest jedynie fizyczna. Drugą numeracją jest numeracja GPIO; nie wygląda ona tak łatwo jak ta pierwsza, więc by jej używać potrzebny nam jest schemat numeracji GPIO.

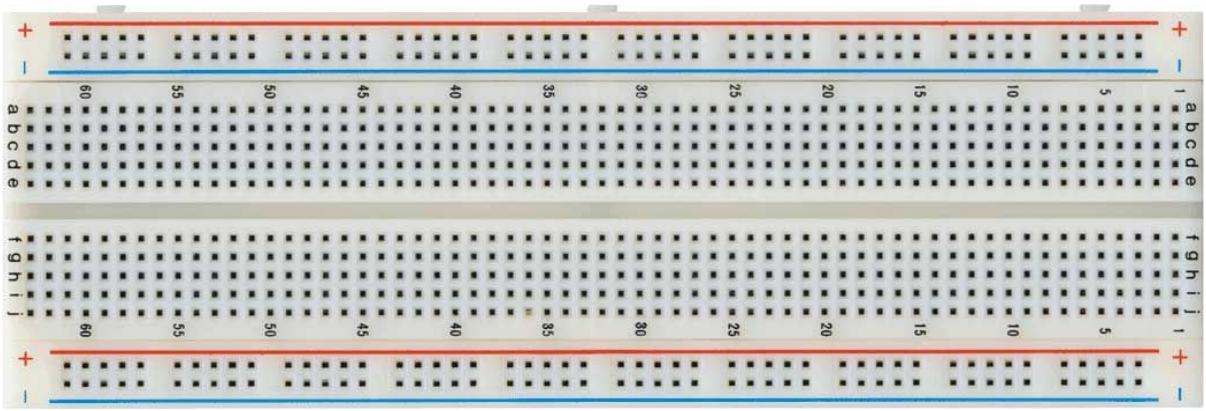


źródło: pinout.xyz

Dobrą praktyką jest używanie numeracji GPIO. Numeracja pinów w programie może też zależeć od języka programowania; niektóre z nich pozwalają na użycie tylko jednej z numeracji, a niektóre dają nam wybór. Przykładowo język Python pozwala nam na ustawienie, z której numeracji będziemy korzystać.

Każdy pin może być w stanie wysokim lub niskim. Jeśli pin jest ustawiony w stan wysoki, wtedy wypuszcza 3,3V (nie dotyczy to pinów zasilających jak i masy).

3.2. Płytka stykowa (płytka typu „breadboard”)



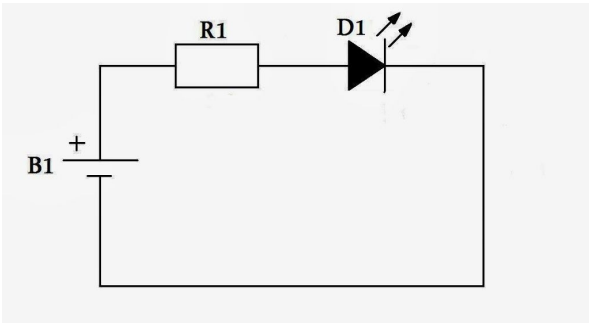
źródło: technovade.pl

Typowa płytka typu “breadboard ma wewnętrzne połączenia, które znacząco ułatwiają pracę przy niej. Najistotniejszym takim połączeniem jest przykładowo szereg poziomo ułożonych styków, opatrzonych po lewej i po prawej stronie znakiem ‘-’ (np rząd na dole). Możemy to wykorzystać, podpinając taki szereg jednym połączeniem do masy na Raspberry Pi. Będziemy wtedy mieli bezpośredni dostęp do masy na całej długości płytki stykowej.
Kolejnym przykładem wewnętrznego połączenia są pionowe rzędy (np rząd nr 60). Taki rząd jest podzielony na dwie części abcde - fghij. Każda z tych części jest połączona wewnętrznie osobno. Wykorzystamy te fakty do podłączania np diod LED.

3.3. Słowo o rezystorach

Podpinając diody LED występuje potrzeba użycia również rezystorów. Gdybyśmy nie podpięli rezystora, bądź podpięli, lecz o zbyt małej rezystancji w najlepszym przypadku doprowadziłoby to do skrócenia żywotności diody. W przypadku zbyt dużego prądu mogłoby dojść do wybuchu diody. Jakich zatem rezystorów użyć? Do obliczeń można użyć dostępnych w internecie kalkulatorów, jednakże warto wiedzieć na jakiej zasadzie następuje kalkulacja. Na przykładzie pokażemy jak obliczyć minimalną wartość rezystora.

Weźmy przykładowo czerwoną diodę LED o prądzie przewodzenia równym 20mA i napięciu przewodzenia 2,1V. Źródłem zasilania niech będzie wyprowadzenie z Raspberry Pi - pin dający napięcie 5V.



źródło: robotykadlapoczatkujacych.pl

Uwzględniając spadek napięcia na diodzie otrzymujemy napięcie na rezystorze

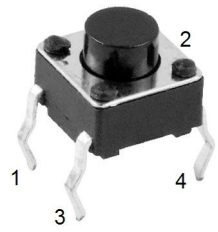
$$U_{R1} = U_{B1} – U_{D1}$$
$$U_{R1} = 5V – 2,1V$$
$$U_{R1} = 2,9V$$

Powyżej korzystamy wprost z II prawa Kirchoffa - napięcie zasilania jest równe spadkom napięć na odbiornikach. Teraz skorzystamy z prawa Ohma, aby wyliczyć minimalną wartość rezystora.

$$R1 = U_{R1} / I$$
$$R1 = 2,9V / 20mA$$
$$R1 = 2,9V / 0,02A$$
$$R1 = 145Ω$$

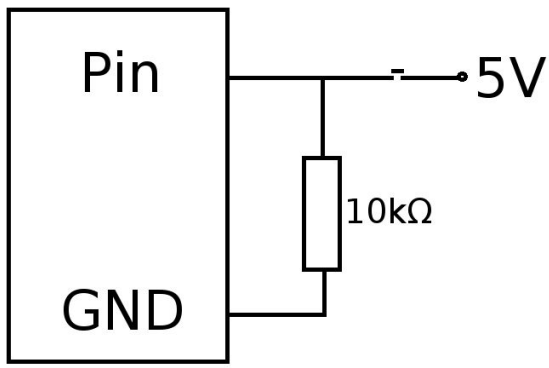
Obliczyliśmy, że rezystor powinien mieć minimalną wartość 145Ω. Możemy użyć jednego rezystora o wartości większej niż obliczona, bądź kilku połączonych szeregowo.

3.4. Podłączenie przycisku tact switch

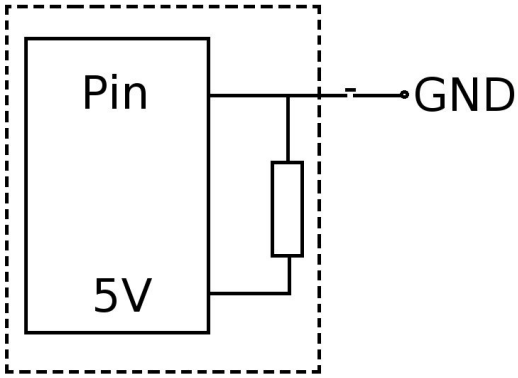


źródło: majsterkowo.pl

Na zdjęciu powyżej mamy przedstawiony typowy przycisk tact switch, taki też został użyty do projektu. Najważniejsze co musimy o nim wiedzieć, to fakt, że nóżki 3 i 4 oraz 1 i 2 są ze sobą zwarte na stałe. Obie strony będą ze sobą zwierane podczas naciskania guzika.
Podczas podłączania przycisku do złącza GPIO logicznym by się mogło wydawać, aby podłączyć przycisk z jednej strony do jakiegoś pinu, a z drugiej do napięcia. Jest to błędne podejście, gdyż nie wiemy jaki stan mamy w danym pinie, niepodłączonym ani do zasilania, ani do masy (nie naciśnięty przycisk nie zwiera do napięcia). Najprostszym rozwiązaniem jest podciągnięcie pinu tzw. rezystorem podciągającym do masy. Przedstawia nam to poniższy schemat:

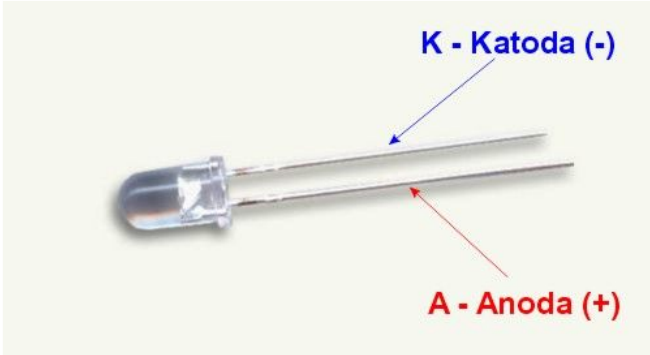


Taki układ nazywamy podciągnięciem w dół - PULL-DOWN. Zamieniając miejscami na schemacie powyżej GND z 5V otrzymalibyśmy układ z podciągnięciem w górę - PULL-UP.
Warto mieć świadomość, że architektura procesora Raspberry Pi (BCM2835, BCM2836 , BCM2837 - numery rdzeni chipów kolejnych minikomputerów Raspberry Pi) posiada oba wbudowane rezystory podciągające. To, z którego podciągnięcia będziemy korzystać jest konfigurowane z poziomu programu. Na poniższym schemacie, to co jest zawarte w ramce (przerywana linia) jest wbudowane w minikomputer. Niweluje nam to użycie rezystora.



Zatem jedną stronę przycisku podpinamy do pinu na Raspberry Pi, a drugą do masy. W tej konfiguracji w programie musimy ustawić rezystor podciągający w górę.

3.5. Podłączenie diod LED

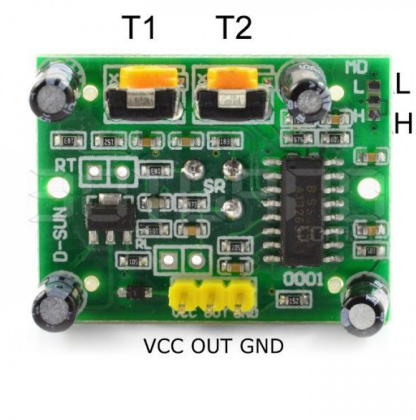


źródło: e-alarmy.pl

Na powyższym obrazku przedstawioną mamy diodę świecącą. Dłuższa z nóżek to anoda, krótsza - katoda. W przypadku, gdy obie nóżki mają tę samą długość (np zostały złamane), aby rozróżnić je należy spojrzeć w samą diodę. Nóżka, która odchodzi od dłuższej blaszki wewnątrz diody to katoda (-).
Podłączając diodę do płytki stykowej należy pamiętać o rezystorze. To jakiego rezystora użyć zostało opisane w punkcie 3.3 tej dokumentacji. Zatem prawidłowo zrealizowane połączenie wygląda następująco:

pin -> anoda katoda -> rezystor -> masa

3.6. Podłączenie czujnika ruchu



źródło: botland.com.pl

Czujnik ruchu PIR (Passive Infra Red) pozwala na wykrywanie ruchu. Działanie opiera się na bardzo precyzyjnym pomiarze temperatury; każda zmiana na wyższą lub niższą (nawet podmuch powietrza) jest traktowana jako alarm. Głównymi elementami czujnika są: pyroelement (detektor podczerwieni) i soczewka Fresnela. Czujnik zasilany jest napięciem od 4,5V do 20V. Do Raspberry Pi pin VCC z powyższego obrazka zostanie podpięty do zasilania 5V, pin GND do masy. Wykrycie obiektu sygnalizowane jest stanem wysokim na pinie oznaczonym OUT.
Czujnik posiada dwa potencjometry, którymi można regulować:
T1 - czas trwania stanu wysokiego po wykryciu obiektu
T2 - czułość czujnika
Mamy na obrazku jeszcze dwa oznaczenia: L i H. Znajdują się one obok trzech dodatkowych pinów, każde dwa zewnętrzne można ze sobą połączyć za pomocą zworki - w tryb pracy L lub w tryb pracy H.
Tryb pracy H (retriggering) - stan wysoki na wyjściu jest utrzymywany przez cały czas wykrywania ruchu; czujnik domyślnie znajduje się w tym trybie (bez zworki).
Tryb pracy L (non-retriggering) - wyjście osiąga stan wysoki tylko raz po wykryciu obiektu, potem niezależnie od ruchu przechodzi w stan niski.

4. Biblioteka wiringPi

4.1. Instalacja

Na początek potrzeba zainstalować podstawowe narzędzia do języka Python. Jeśli istnieją już one na naszym minikomputerze, wykonanie poniższego polecenia nic nie wniesie, ale też nie zaszkodzi:

```
~$ sudo apt-get install python-dev python-pip
```

Następnie pobieramy i instalujemy bibliotekę wiringPi:

~$ sudo pip install wiringpi

To wszystko. Teraz możemy sprawdzić działanie w środowisku Python. Poniższymi poleceniami uruchomimy środowisko Python, zaimportujemy bibliotekę oraz sprawdzimy wersję naszego minikomputera Raspberry Pi:

```
~$ sudo python
>>> import wiringpi
>>> wiringpi.piBoardRev()
```

Poniższymi poleceniami możemy sprawdzić wersję biblioteki oraz schemat pinów na Raspberry Pi oraz każdą z możliwych numeracji:

```
~$ gpio -v
~$ gpio readall
```

4.2. Podstawowe informacje

Pod poniższym linkiem znajdziemy dokumentację biblioteki wiringPi. Dotyczy ona języka C, aczkolwiek używanie jej w języku Python jest analogiczne.

https://projects.drogon.net/raspberry-pi/wiringpi/functions/

Poniżej przedstawione zostaną podstawowe informacje, dotyczące tego jak korzystać z biblioteki.

Na początku potrzebujemy zaimportować do naszego skryptu bibliotekę oraz ustawić początkowo piny GPIO:

```
import wiringpi as GPIO

GPIO.wiringPiSetupGpio()
```

Funkcje do inicjalizacji pinów są trzy, każda z nich w różny sposób to robi. My będziemy korzystali z **wiringPiSetupGpio(),** która pozwala korzystać z numeracji BCM.

Funkcja **pinMode(PIN, I/O)** pozwala na ustawienie konkretnego pinu jako wejściowy lub wyjściowy, przykładowe użycie:

```
GPIO.pinMode(23, GPIO.INPUT)
GPIO.pinMode(27, GPIO.OUTPUT)
```

W przypadku potrzeby ustawienia rezystora podciągającego (patrz punkt 3.4.) skorzystamy z funkcji **pullUpDnControl(PIN, UP/DOWN)**, przykładowe użycie:

```
GPIO.pullUpDnControl(23, GPIO.PUD_UP)
```

Do odczytania wartości z danego pinu służy funkcja **digitalRead(PIN).** Wartością zwracaną jest wartość LOW lub HIGH, przykład użycia wraz ze sprawdzeniem stanu:

```
if(GPIO.digitalRead(25) == GPIO.LOW)
```

Funkcją służącą do zapisywania wartość LOW lub HIGH na dany pin jest **digitalWrite(PIN, LOW/HIGH).** Jej przykładowym użyciem jest:

```
GPIO.digitalWrite(22, GPIO.HIGH)
```

Dobrą praktyką jest, aby przed zakończeniem działania skryptu uporządkować używane przez nas piny. W tym celu, przy pomocy powyższych funkcji, ustawiamy piny wyjściowe na wejściowe oraz do każdego pinu przypisujemy wartość LOW.

4.3. Uruchamianie skryptu

Aby wykonać skrypt, korzystający z biblioteki wiringPi konieczne jest uruchomienie go z uprawnieniami administratora. W tym celu korzystamy z polecenia:

```
~$ sudo python script.py
```

5. Moduł rozpoznawania twarzy

5.1 Instalacja modułu

Do korzystania z modułu wykrywania twarzy potrzebna nam będzie biblioteka openCV czyli Open Source Computer Vision. Jej instalacja jest dosyć skomplikowana i pracochłonna, może zająć nawet kilka godzin, musimy bowiem wykonać wszystkie poniżej opisane kroki:

Krok 1 - Instalacja zależności:

Na tym etapie pracy zakładamy, że nasz system Raspbian jest w aktualnej wersji. Poniższymi poleceniami wykonamy instalację zależności

~$ sudo apt-get install -y build-essential
~$ sudo apt-get install -y cmake
~$ sudo apt-get install -y libgtk2.0-dev
~$ sudo apt-get install -y pkg-config
~$ sudo apt-get install -y libjpeg-dev
~$ sudo apt-get install -y libtiff5-dev
~$ sudo apt-get install -y libjasper-dev
~$ sudo apt-get install -y libpng12-dev
~$ sudo apt-get install -y libavcodec-dev
~$ sudo apt-get install -y libavformat-dev
~$ sudo apt-get install -y libswscale-dev
~$ sudo apt-get install -y libv4l-dev
~$ sudo apt-get install -y libxvidcore-dev libx264-dev
~$ sudo apt-get install -y libatlas-base-dev gfortran

Krok 2 - Pobranie biblioteki:

Poniższe polecenia służą do pobrania biblioteki openCV w wersji 3.1.0:

~$ cd ~
~$ wget -O opencv.zip https://github.com/ltseez/opencv/archive/3.1.0.zip
~$ unzip opencv.zip
~$ wget -O opencv_contrib.zip
https://github.com/ltseez/opencv_contrib/archive/3.1.0.zip
~$ unzip opencv_contrib.zip

Krok 3 - Konfiguracja Pythona i środowisk

Najpierw należy pobrać program **pip**, którym można instalować pythonowe paczki:

~$ wget https://bootstrap.pypa.io/get-pip.py
~$ sudo python get-pip.py

Następnie instalujemy biblioteki do środowisk wirtualnych w Pythonie:

~$ sudo pip install virtualenv virtualenvwrapper
~$ sudo rm -rf ~/.cache/pip

Aby mieć możliwość uruchomienia wirtualnego środowiska trzeba dodać dwie linijki na końcu pliku **.profile**, uczynimy to przy pomocy edytora tekstowego **vim**:

~$ vim ~/.profile

Na końcu tego pliku wklejamy poniższe linijki:

```
# virtualenv and virtualenvwrapper
export WORKON_HOME=$HOME/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

Aby zmiany zostały wprowadzone musimy wykonać jedną z czynności: wylogować się zalogować ponownie do systemu lub po prostu wykonać poniższe polecenie:

```
~$ source ~/.profile
```

Środowisko tworzymy poleceniem:

```
~$ mkvirtualenv cv
```

Aby przejść do środowiska musimy wykonać:

```
~$ source ~/.profile
~$ workon cv
```

Powyższe dwie linijki będą konieczne na początku każdej pracy z modułem do wykrywania twarzy. Uwarunkowane jest to tym, że bibliotekę openCV zbudujemy w wirtualnym środowisku, które wcześniej utworzyliśmy. Aby mieć pewność, że prawidłowo przenieśliśmy się do wirtualnego środowiska sprawdźmy wiersz poleceń w konsoli. Musi on być poprzedzony poprzez: “(cv)”. Do poprawnego zbudowania biblioteki openCV ważna jest zależność o nazwie NumPy. Instalujemy ją poniższym poleceniem (proces zajmie około 10-15 minut):

```
~$ pip install numpy
```

Krok 4 - instalacja OpenCV

Z katalogu głównego wykonujemy kolejno:

```
~$ cd ~/opencv-3.1.0/
~$ mkdir build
~$ cd build
~$ cmake -D CMAKE_BUILD_TYPE=RELEASE \
        -D CMAKE_INSTALL_PREFIX=/usr/local \
        -D INSTALL_C_EXAMPLES=OFF \
        -D INSTALL_PYTHON_EXAMPLES=ON \
```

-D BUILD_EXAMPLES=ON ..

~$ make -j4

Ostatnie polecenie (**make -j4**) zajmie trochę czasu, nawet do dwóch godzin. Po skompilowaniu biblioteki openCV musimy zainstalować ją na naszym sytemie:

~$ sudo make install
~$ sudo ldconfig

Na sam koniec pozostaje nam powiązać openCV z naszym środowiskiem “cv”:

~$ cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
~$ ln -s /usr/local/lib/python2.7/site-packages/cv2.so cv2.so

Jeżeli wszystko zostało wykonane poprawnie proces instalacji jest w tym momencie zakończony. Możemy to zweryfikować w poniższy sposób:

~$ workon cv
~$ python
>>> import cv2
>>> cv2.__version__

Powinniśmy otrzymać aktualną wersję openCV, czyli w tym przypadku 3.1.0.
Po zainstalowaniu biblioteki openCV możemy przejść do korzystania ze skryptów, które wykorzystują detekcję twarzy.

5.2. Konflikt użytkowników

Tworzenie environmentu jako inny użytkownik niż root może spowodować późniejsze problemy z odpaleniem skryptów jako administrator (sudo python skrypt.py), co jest konieczne dla skryptów, które korzystają z funkcji biblioteki wiringpi.

Rozwiązaniem tego problemu jest zmiana dostępu do plików naszego środowiska wirtualnego:

$ cd ~/.virtualenvs
$ sudo chown -R root:root cv
$ sudo chmod -R a+rX cv

W przypadku, gdy powyższe działanie nie przyniesie efektu (objawia się to zazwyczaj błędem “No module named cv2” przy wywołaniu skryptu) pozostaje obejście problemu poprzez skopiowanie pliku z poniższej ścieżki:

/usr/local/lib/python2.7/site-packages/cv2.so

do katalogu, w którym odpalamy skrypt.

6. Moduł Bluetooth

6.1. Instalacja modułu

Do Raspberry Pi 2 wymagany jest zewnętrzny odbiornik Bluetooth, w naszym przypadku korzystamy z odbiornika podłączonego do USB. Projekt jest kompatybilny z Raspberry Pi 3, a w tym modelu minikomputera moduł Bluetooth jest już wbudowany w płytkę, więc nie jest potrzebny zewnętrzny odbiornik.
Najnowszy obraz systemu Raspbian (wrzesień 2016) posiada już potrzebne rzeczy do poprawnej obsługi Bluetooth, aczkolwiek warto sprawdzić, czy istnieją nowsze sterowniki. Dla starszych obrazów systemu poniższe kroki są obowiązkowe do poprawnego działania modułów bezprzewodowej komunikacji.

Po podłączeniu zewnętrznego modułu Bluetooth możemy wykonać poniższe polecenie. Pokaże nam ono wszystkie urządzenia podpięte do naszego Raspberry.

~$ lsusb

W terminalu wykonajmy poniższe polecenie:

~$ sudo apt-get install bluetooth blueman bluez python-gobject python-gobject-2 libbluetooth-dev

Zainstaluje (lub, jeśli już istniały, zaktualizuje) wsparcie dla modułu Bluetooth oraz graficzne narzędzie do obsługi bezprzewodowej komunikacji. Od tego momentu wraz z uruchamianiem minikomputera moduł Bluetooth automatycznie będzie startowany. Kolejnym poleceniem możemy sprawdzić informacje, dotyczące modułu Bluetooth:

~$ sudo service bluetooth status

Wykonując poniższe polecenie możemy sprawdzić nasz zewnętrzny odbiornik:

~$ hcitool dev

Aby móc korzystać z modułu bluetooth z poziomu pythona konieczna jest także instalacja pybluez, którą wykonamy poniższą komendą:

~$ pip install pybluez

6.2. Komunikacja

Do komunikacji Raspberry - urządzenie zewnętrzne potrzebne będzie skojarzenie tych urządzeń. Możemy to zrobić prosto za pomocą wcześniej wspomnianego graficznego interfejsu. Jednak równie dobrze możemy to zrobić za pomocą konsoli, będziemy potrzebować do tego kilku komend oraz adresu MAC naszego zewnętrznego urządzenia np. telefonu.

Komunikacja pomiędzy telefonem a minikomputerem będzie oparta o port szeregowy, zatem musimy załadować **“Serial Port Profile”.** Zanim to uczynimy musimy wprowadzić zmianę do pliku **etc/systemd/system/dbus-org.bluez.service.** Po zmianie zmienna **ExecStart** powinna przyjąć wartość:

ExecStart=/usr/lib/bluetooth/bluetoothd -C

Następnie uruchamiamy ponownie system:

~$ reboot

“Serial Port Profile” ładujemy poleceniem:

~$ sudo sdptool add SP

Sprawmy, aby nasz minikomputer był widoczny dla innych urządzeń:

~$ sudo hciconfig hci0 piscan

Aby umożliwić parowanie urządzeń za pomocą Bluetooth musimy wpierw uruchomić serwer dźwięku na Raspberry Pi. Posłuży nam do tego polecenie:

~$ pulseaudio --start

Następnie wykonujemy kilka poleceń, dzięki którym uruchomimy obsługę modułu Bluetooth w terminalu:

~$ sudo bluetoothctl

~$ agent on
~$ default-agent
~$ scan on

Na tym etapie powinniśmy mieć wyświetloną listę dostępnych urządzeń do skojarzenia. Połączmy teraz ze sobą urządzenia (xx:xx:xx:xx:xx oznacza adres MAC urządzenia zewnętrznego):

~$ pair xx:xx:xx:xx:xx
~$ trust xx:xx:xx:xx:xx
~$ connect xx:xx:xx:xx:xx

Istnieje możliwość, że zostaniemy poproszeni o podanie klucza, który zostanie ukazany np. na ekranie telefonu [można też go ustawić ręcznie podczas łączenia urządzeń]. Jednakże nie zawsze ta opcja wystąpi np. przy parowaniu klawiatury nie zostaniemy poproszeni o klucz.
Aby nasze urządzenie zewnętrzne było łączone za każdym razem automatycznie z minikomputerem będziemy potrzebowali wpisać poniższą linijkę do pliku **/etc/rc.local.** W tym celu wykonujemy polecenie:

~$ sudo nano /etc/rc.local

Do otwartego pliku wklejamy linijkę:

sudo hidd -i hci0 --connect xx:xx:xx:xx:xx

Na tym etapie jesteśmy gotowi do komunikacji pomiędzy telefonem a Raspberry Pi.

6.3. Skrypt Python

Na minikomputerze uruchamiany skrypt będzie pełnił rolę serwera. Do napisania skryptu komunikującego się za pomocą Bluetooth z zewnętrznym urządzeniem posłużył nam przykładowy skrypt, który możemy znaleźć pod ścieżką:

/usr/share/doc/python-bluez/examples/simple/rfcomm-server.py

Powyższy plik, zmodyfikowany, posłużył do napisania skryptu obsługującego kamerę z wszelkimi funkcjonalnościami oraz zapewnia komunikację z telefonem.

Uruchomienie skryptu może powodować błąd:

bluetooth.btcommon.BluetoothError: (115, 'Operation now in progress')

Aby go wyeliminować do pliku **/etc/bluetooth/main.conf** dodajemy linijkę:

DisablePlugins = pnat

Najważniejszą informacją jest, aby wartość zmiennej UUID w skrypcie była identyczna jak wartość zmiennej UUID w kodzie aplikacji mobilnej, co zostanie również wspomniane w część dokumentacji, poświęconej aplikacji mobilnej.

7. Aplikacja mobilna

Aplikacja mobilna została napisana na system operacyjny Android przy wykorzystaniu API19. Odpowiada to wersji systemu Android 4.4 KitKat, oraz każdej wyższej. Za pomocą udostępnionych funkcjonalności możemy z telefonu sterować jednym z programów uruchomionym na naszym minikomputerze. Tym programem jest plik **camera.py** z katalogu **Camera**. Więcej na temat funkcjonalności programu, uruchamianego na Raspberry Pi znajdziemy w dokumentacji użytkownika. Tam też znajdziemy instrukcję obsługi aplikacji mobilnej.

7.1. Podstawowe informacje

Aplikacja mobilna łączy się z minikomputerem za pomocą Bluetooth, przy czym robi to automatycznie. Jedyne co musimy zrobić, aby zapewnić kompatybilność i połączenie, to ustawić nazwę Bluetooth dla Raspberry Pi na “raspberrypi”. Możemy także użyć innej nazwy, aczkolwiek musimy pamiętać, aby zmienić też daną nazwę w linijce **165** w pliku **ControlPanel.java** aplikacji mobilnej.

Kolejną istotną rzeczą jest, aby UUID w obu aplikacjach się zgadzały. W skrypcie naszego programu, będącego naszym serwerem UUID musi być identyczne z tym, które znajduje się w kodzie aplikacji mobilnej, będącej klientem całego połączenia.

Po ewentualnych zmianach możemy przejść do kompilacji oraz instalacji aplikacji mobilnej na naszym telefonie. Projekt aplikacji mobilnej zaleca się otworzyć w programie Android Studio.

7.2. Instalacja

W tym punkcie musimy sprawdzić, czy są spełnione założenia z punktu **7.1**. Po tej czynności możemy przejść do instalacji aplikacji na telefonie z systemem Android w wersji co najmniej 4.4 KitKat. Pierwszą czynnością jaką musimy wykonać jest uruchomienie trybu programisty. Należy przejść do opcji **“Informacje o urządzeniu”**

w ustawieniach telefonu, a następnie odnaleźć pozycję **“Numer wersji”.** Dotykamy tę pozycję 7 razy w bardzo krótkich odstępach czasu.

Pojawi nam się na dole ekranu informacja o tym, że przeszliśmy w tryb programisty. Wchodzimy ponownie w ogólne ustawienia telefonu, gdzie na dole odnajdziemy opcję **“Opcje programisty”.** Po pojawieniu się kolejnego ekranu szukamy opcji **“Debugowanie USB”,** zaznaczamy ją. Teraz nasz telefon jest przygotowany pod instalację aplikacji.

Podpinamy telefon do komputera za pomocą kabla USB. W programie Android Studio opcją **“Run”** uruchamiamy aplikację. Pojawi nam się okienko, w którym będziemy mieli możliwość wyboru urządzenia docelowego. Po wyborze czekamy, aż na telefonie otworzy nam się aplikacja.

Teraz jesteśmy gotowi do komunikacji aplikacji na telefonie z programem na Raspberry Pi.

8. Bibliografia

http://raspi.tv/how-to-install-wiringpi2-for-python-on-the-raspberry-pi
https://www.raspberrypi.org/learning/parent-detector/worksheet/
http://www.pyimagesearch.com/2015/10/26/how-to-install-opencv-3-on-raspbian-jessie/

9. Link do repozytorium

Całość kodu projektu jest dostępna pod linkiem:
https://github.com/MarcinLason/EmbeddedSystems

II. Dokumentacja użytkownika

1. Struktura katalogów i proste programy

Projekt składa się z kilku katalogów, które dostarczają zarówno skrypty oferujące pełną funkcjonalność projektów oraz kilka skryptów demonstrujących działalność konkretnych modułów aplikacji. Są one zaopatrzone w dodatkowe komentarze oraz dodatkowe działanie prezentujące możliwości modułu.

1.1. Bluetooth_module

W katalogu Bluetooth_module znajduje się skrypt, pozwalający łączyć się z telefonem użytkownika przy pomocy modułu Bluetooth. Prosta komunikacja pomiędzy minikomputerem, a aplikacją mobilną, opisaną w poniższych punktach polega na wysyłaniu i odbieraniu prostych łańcuchów znaków.

1.2. Camera

W tym katalogu znajduje się skrypt, który wykorzystuje możliwości kamery oraz wykrywacza ruchu. Po inicjalizacji odpowiednich pinów GPIO przechodzi do pętli, w której użytkownik może przechodzić pomiędzy różnymi trybami za pomocą przycisku tact switch. Dostępne tryby to: wykonywanie zdjęć, nagrywanie filmu oraz wykrywanie ruchu.

Uruchomienie skryptu sprowadza się do wykonania z poziomu katalogu:

~$ sudo python Camera.py

Po starcie skryptu użytkownik znajduje się w trybie wykonywania zdjęć. Aby przejść do kolejnych trybów wystarczy przycisnąć odpowiedni guzik. Drugi z nich z kolei służy do wykonywania akcji zdefiniowanej dla danego trybu. I tak oto w trybie wykonywania zdjęć możemy w dowolnym momencie wykonać zdjęcie, które zostanie zapisane do katalogu: `/home/pi/Desktop/Camera/Photos/`. W trybie nagrywania wideo użytkownik nagrywa film, którego długość zależy od tego jak długo dany przycisk będzie trzymany. Ostatni tryb powoduje wprowadzenie urządzenia w stan wykrywania ruchu, w momencie zarejestrowania zmiany ciepła w najbliższej przestrzeni skrypt zaczyna nagrywanie filmu, który trwa aż do ustabilizowania się otoczenia. Film ten zapisywany jest do katalogu: `home/pi/Desktop/Camera/Videos/`.

O wszystkich akcjach, takich jak przełączenie trybów, zrobienie zdjęcia, czy nagranie filmu użytkownik jest informowany poprzez odpowiednie komunikaty na konsoli.

1.3. OpenCV_Module

W powyższym katalogu znajdziemy skrypt prezentujący możliwości biblioteki openCv2 oraz plik haarcascade_frontalface_default.xml, który stanowi zbiór wartości, które muszą zostać spełnione, aby dany element zdjęcia został zakwalifikowany jako twarz. Ogólnie rzecz biorąc jest to plik, który powoduje, że skrypt jest w stanie rozpoznawać twarze.

Aby uruchomić skrypt musimy wcześniej przejść do odpowiedniego środowiska pythonowego:

```
~$ source ~/.profile
~$ workon cv
```

A następnie z poziomu katalogu:

```
~$ python OpenCVTest.py
```

Jeśli chodzi o działanie skryptu to polega ono na wykonaniu zdjęcia i zapisaniu go w odpowiednim katalogu. Następnie jest ono z powrotem wczytywane do programu i konwertowane, na potrzeby OpenCv, do obrazu w odcieniach szarości. Skrypt dokonuje potem detekcji twarzy i wypisuje komunikat o rezultacie tego działania. Na oryginalnym zdjęciu (tym wykonanym w kolorze) zaznaczane są prostokątne ramki w miejscach, gdzie wykryto twarz. O ile na zdjęciu udało się jakąś znaleźć, zdjęcie jest zapisywane w osobnym katalogu, a na pulpicie wyświetla się obraz dopasowany do rozdzielczości monitora użytkownika, przedstawiający wynik działania skryptu.

1.4. CameraControl

Powyższy katalog zawiera część głównej aplikacji projektu. Jest to część przygotowana do instalacji na urządzeniu mobilnym, która będzie służyć do sterowania przebiegiem działania skryptu z katalogu Full_Program. Proces instalacji został opisany w punkcie 7.2 dokumentacji technicznej.

1.5. Full_Program

Ten katalog zawiera skrypt, będący połączeniem wszystkich opisanych wcześniej funkcjonalności. Jest to najważniejsza część projektu i została opisana w następnym punkcie dokumentacji.

2. Obsługa głównego programu

2.1. Założenia programu

Główny program posiada możliwość pracy w trzech, wcześniej wspomnianych trybach: wykonywania zdjęć, nagrywania filmików, a także wykrywania ruchu, wzbogacony o moduł wykrywania twarzy.
Uruchomienie programu zostało zautomatyzowane poprzez skrypt napisany w bashu. Do jego odpalenia wystarczy jedynie, że wpiszemy:

~$ chmod +x run.sh (jednorazowo, przy pierwszym uruchomieniu)
~$ sudo /.run.sh

Użytkownik rozpoczyna działanie w trybie wykonywania zdjęć. Przełączenie pomiędzy trybami może realizować zarówno poprzez przyciski Tact Switch na urządzeniu oraz z poziomu aplikacji mobilnej.

Po uruchomieniu skryptu na dysku użytkownika zostanie utworzona struktura katalogów, którą wykorzystuje aplikacja:
`/home/pi/Desktop/Camera/Photos/` - tutaj będą zapisywane wszystkie zdjęcia
`/home/pi/Desktop/Camera/Videos/` - tutaj będą zapisywane nakręcone filmy
`/home/pi/Desktop/Camera/FacesDetected/` - tutaj będą zapisywane zdjęcia, na których wykryto twarz

2.2. Obsługa poszczególnych funkcjonalności

2.2.1. Przyciski

Do dyspozycji użytkownika są 3 przyciski. Ten najbardziej po lewej to przycisk służący do zamknięcia funkcjonalności samych przycisków (aby zabezpieczyć działanie skryptu przed niepowołanymi osobami np. podczas wykrywania ruchu i rozpoznawania twarzy można wyłączyć przyciski i sterować jedynie za pomocą aplikacji mobilnej). Środkowy służy do przełączania trybów pracy programu, a ostatni to przycisk akcji. W zależności od obecnie wybranego trybu może służyć do zrobienia zdjęcia (tryb zdjęć), nagrania filmu (tryb wideo) lub być nieaktywny w trybie wykrywania ruchu. Istotnym punktem jest, że za pomocą przycisku tact switch (środkowego) nie możemy uruchomić trybu z detekcją ruchu i rozpoznawanie twarzy, można to zrobić jedynie za pomocą aplikacji mobilnej.

2.2.2. Czujnik ruchu

W trybie wykrywania ruchu wykorzystywany jest bardzo czuły czujnik, który wykrywa ruch na podstawie zmian temperatury w otoczeniu. Jego działanie jest połączone z modułem wykrywania twarzy. Całość działania opisana jest dokładniej w

punkcie 2.5.

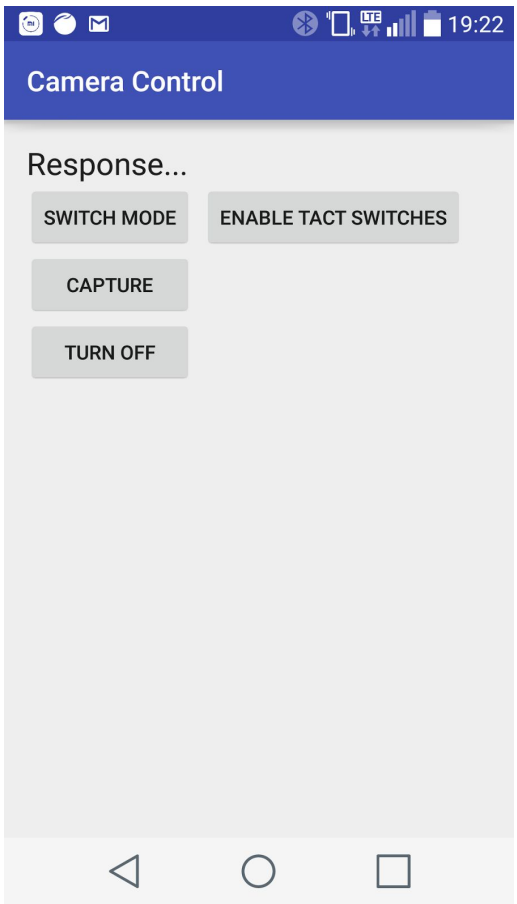
2.2.3. Obiektyw kamery

Istotną rzeczą dla prawidłowego działania programu jest odpowiednie ustawienie obiektywu kamery. Najbardziej wyczulony na to jest moduł wykrywania twarzy. Niestabilność obiektywu może doprowadzić do robienia zamazanych zdjęć, co mocno zwiększa ilość błędów przy detekcji. Ważny jest także odpowiednia odległość od miejsca, gdzie spodziewamy się wykryć ruch. Twarz nie może być kilku pikselowym elementem obrazka lub zajmować niemal cały obrazek.

2.2.4. Bluetooth

Program może być obsługiwany z poziomu aplikacji mobilnej. Posiada ona zbliżone możliwości do fizycznych guzików przełączania. Ważnym jest, aby pamiętać, że aplikacja nawiązuje połączenie z programem na platformie Raspberry Pi poprzez bluetooth, więc nie powinny one działać na odległość większą niż możliwości tej technologii (zazwyczaj przyjmuje się, że odległość poniżej 10 metrów jest odległością bezpieczną).

2.2.5. Aplikacja mobilna



Jak zostało wspomniane w poprzednim punkcie, aplikacja mobilna ma zbliżone działanie do przycisków tact switch. Znajdują się w niej trzy przyciski ułożone jeden pod drugim. Ich funkcjonalność to kolejno: zmiana trybu, akcja, wyłączenie całkowite aplikacji. Na górze wyświetlacza pojawia nam się informacja, dotycząca aktualnie pracującego trybu.

Aplikacja została także wzbogacona w przycisk “Enable Tact Switches”. Dzięki niemu guziki, które fizycznie znajdują się na platformie raspberry, a które zostały wcześniej zablokowane (patrz punkt 2.2.1) odzyskują swoją funkcjonalność.

2.3. Tryb zdjęć

Tryb, w którym rozpoczyna się działanie aplikacji. Udostępnia możliwość zrobienia zdjęcia po każdorazowym wciśnięciu przycisku akcji. Program daje użytkownikowi czas ok. 3 sekund przed zrobieniem zdjęcia na prawidłowe ustawienie kamery. Po tym czasie zrobione zdjęcie jest zapisywane w utworzonym wcześniej katalogu: `/home/pi/Desktop/Camera/Photos/.`

2.4. Tryb wideo

Tryb, który następuje po trybie zdjęć. Pozwala na kręcenie różnej długości filmików (długość zależy od czasu wciskania przycisku akcji). Każdy film zapisywany jest w odpowiednim katalogu, utworzonym na starcie aplikacji: `/home/pi/Desktop/Camera/Videos/.`

2.5. Tryb monitoringu (tryb detekcji ruchu)

Ostatni i najbardziej zaawansowany tryb. To w nim realizowane jest główne zadanie projektu, czyli dostarczenie oprogramowania, które może służyć jako system monitoringu. Na samym starcie zaczyna swoje działanie, opisywany już wcześniej czujnik ruchu. Gdy wykryje ruch, wykonuje zdjęcie i analizuje je pod kątem wystąpienia na nim ludzkich twarzy. Jeżeli jakąś znajdzie, zapisze to zdjęcie (z zaznaczoną twarzą w prostokątnej ramce) w odpowiednim katalogu (`/home/pi/Desktop/Camera/FacesDetected/)` a także poinformuje użytkownika poprzez komunikat na konsoli o wyniku działania. Dzięki zapisywaniu tego rodzaju zdjęć w osobnym katalogu oraz nadawaniu im nazw w postaci dnia i godziny wykonania urządzenie idealnie nadaje się na uproszczony system monitoringu. Wystarczy wprowadzić urządzenie w ostatni tryb i zostawić je z kamerą ustawioną na miejsce, które chcemy obserwować, na przykład na klatkę schodową przed naszymi drzwiami do mieszkania. Dzięki temu dowiemy się, kto kręci się pod naszymi drzwiami, gdy jesteśmy poza domem!

Od momentu ustawienia tego trybu do rozpoczęcia wykrywania ruchu mamy 5 sekund czasu, aby opuścić pomieszczenie. Jest to o tyle łatwiejsze, że tryb włączamy tylko za pomocą aplikacji mobilnej.

Aby wyjść z trybu monitoringu wystarczy ponownie nacisnąć w naszej aplikacji na przycisk zmiany trybu pracy.