Extra Projects You Can Do With Your Raspberry Pi

These are the projects that were emailed to me that people asked to have included in a packet.

Some of them are super cool! I included one that I found using a pi-top Pulse. It isn't cheap ($49.99) but it is super EASY. You can just plug it into your pi-top and instantly get Alexa, the amazon virtual assistant.

Of course, if you need *any help at all* be sure to email me at samuel.bechara@marquette.edu and I would love to help however I can. This sort of thing is my favorite thing to do.

No matter what you do, remember that learning is **hard** and that **anything worth doing is going to be difficult**. You should be proud of all you have learned this week. I am proud of all of you.

Keep up the hard work and I hope to hear from you about all your future projects!

Dr.B

pi-topPULSE

Visit the pi-topPULSE product page on the pi-top website for more information.

Table of Contents

- Quick Start
- Hardware Overview
- Software
  - pi-topPULSE on pi-topOS
  - pi-topPULSE on Raspbian
  - How it works - 'under the hood'
- Using pi-topPULSE
  - Amazon's Alexa Voice Service
  - Using a custom Python script
- Documentation & Support
  - Links
  - Troubleshooting

Quick Start

pi-topPULSE on pi-topOS

- Boot into pi-topOS (released on or after 12-07-2017)
- Plug in pi-topPULSE
- Follow on-screen instructions, if necessary
- Enjoy - check out the examples to see what you can do! And try asking Alexa questions via pi-topDASHBOARD!

---

pi-topPULSE on Raspbian

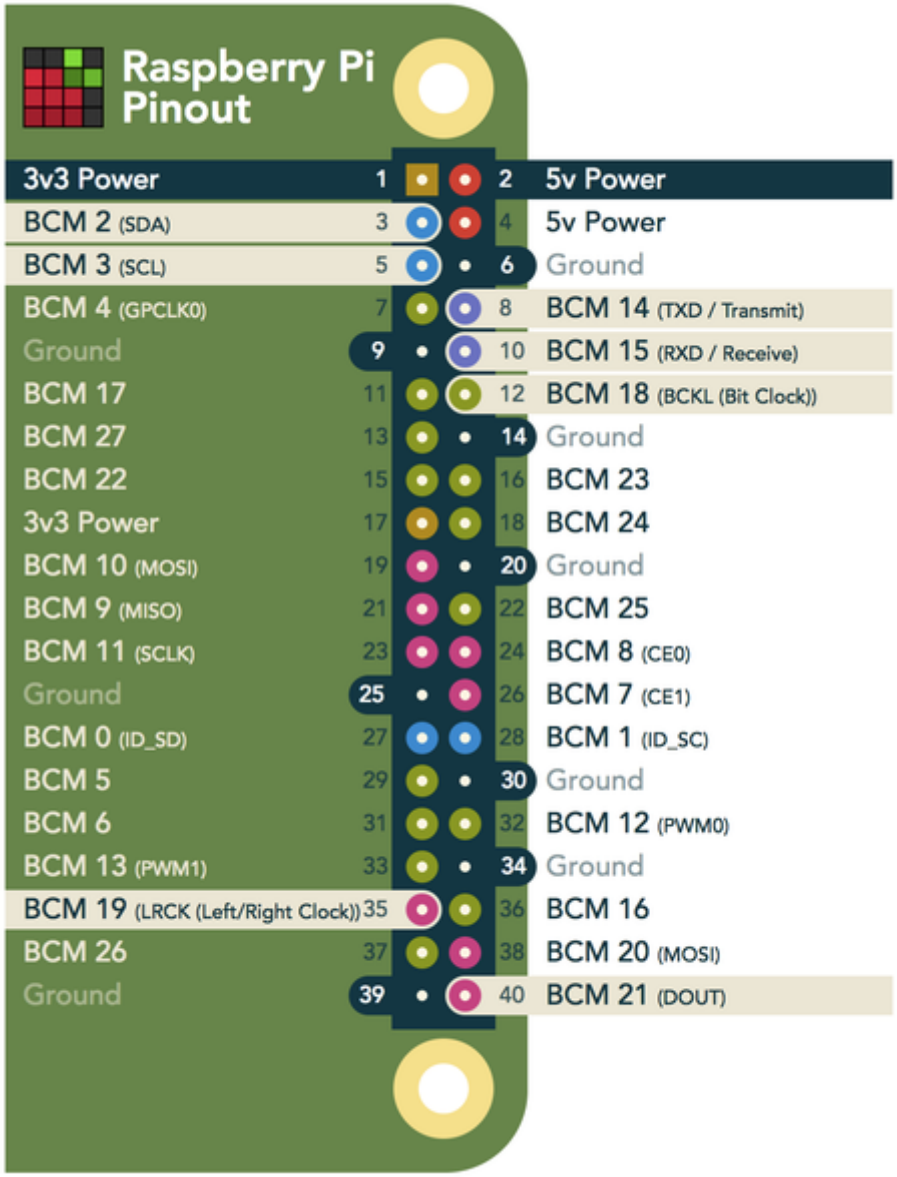
- Run the following commands in the terminal (with an internet connection):

```
sudo apt update
sudo apt install pt-pulse
```

- Plug in pi-topPULSE
- Follow on-screen instructions, if necessary
- Enjoy - check out the examples to see what you can do! See here for more on using pi-top's Alexa Voice Service integration.

Hardware Overview

pi-topPULSE is a 7x7 LED array, a speaker and a microphone. Additionally the device features ambient lights which reflect the state of the LED array, 4 around the speaker, and 3 on the underside. pi-topPULSE uses a variety of interfaces to communicate with the Raspberry Pi: the speaker uses I2S, and the LEDs and microphone use serial (UART) - Tx and Rx respectively. pi-topPULSE can be used either as a HAT or as pi-top addon.



Software

pi-topPULSE on pi-topOS

All pi-topPULSE software and libraries are included and configured 'out-of-the-box' as standard on pi-topOS (released on or after 12-07-2017). Simply connect a pi-topPULSE to your pi-top, reboot if instructed to do so, and it will be automatically initialised and ready to produce light, capture and play audio. Volume control is handled by the operating system.

Download the latest version of pi-topOS at https://pi-top.com/products/os#download.

As mentioned in the Hardware Overview, the speaker on the pi-topPULSE uses I2S. This requires some configuration, which will require a reboot from a typical Raspbian configuration using the default sound drivers. This is also true in reverse - if you have configured a pi-topPULSE and you wish to use the standard HDMI or 3.5mm sound outputs, you will require a reboot.

Additional information

Automatic initialisation is performed by a software package called `pt-peripheral-cfg`. It contains a program called `pt-peripherals-daemon`, which runs in the background and scans for newly connected devices. If a device is detected, and the appropriate library is installed, it will be initialised.

The `pt-pulse` package on pi-topOS installs and starts this background process, as well as the Python library. In the case of pi-topPULSE, it enables I2S and configures UART for next boot, if not currently enabled/configured, and notifies the user if a reboot is required. If a reboot is not required, it will initialise the device.

---

pi-topPULSE on Raspbian

The pi-topPULSE software exists on the Raspbian software repositories. Simply run the following commands at the terminal (and then reboot):

```
sudo apt update
sudo apt install pt-pulse
```

If you prefer to manually install the packages or want to install a specific set of packages see the Manual Configuration and Installation page on the wiki.

How it works - 'under the hood'

For more information on how to use the library files, take a look at the initialisation section of the 'Manual Configuration and Installation' page on

the wiki. Also check out the examples folder for guidance of what the library is capable of.

Using pi-topPULSE

Amazon's Alexa Voice Service

See here for more on using pi-top's Alexa Voice Service integration.

Using a custom Python script

Using the `ptpulse` Python module requires root access to function. If you are running a script, make sure that you are running it with root access. You can do this with the "sudo" command:

```
sudo python3 my_cool_pulse_script.py
```

Alternatively, if you are running Python in `IDLE`, please make sure you start LXTerminal and run idle or idle3 with the "sudo" command, like so:

```
sudo idle3
```

Documentation & Support

Links

- Support

Troubleshooting

Why is my pi-topPULSE not working?

- Currently, **pi-topPULSE is only supported on Raspberry Pi 3**. This is due to problems setting the UART clock speed on earlier Raspberry Pi models. It might be possible to get this to work on earlier versions, but this is not currently supported.

I have installed pi-topPULSE software manually...

- If you are running Linux kernel version 4.9.x previous to 4.9.35, pi-topPULSE may not be fully functional. In particular, this issue prevents the pi-topPULSE LEDs from working. If you are experiencing this issue, please check your kernel version by typing `uname -r` at the terminal. You can update your kernel version to the latest by running `sudo apt install raspberrypi-kernel`.
- If you are attempting to use Python 3, and have installed manually, you need to ensure that you have the latest version of the PySerial module. Take a look at the script in the `manual-install` directory for how to do this.

The red LED on the underside of my pi-topPULSE is on - what does this mean?

- This LED indicates that the sampling rate of the microphone is set to 16kHz. By default, when plugged in, you will see that this LED is switched on, and can be used as a guide to show that it has not yet been initialised. Once initialised, the default sample rate is 22050Hz (~22KHz), and this is why the red LED is switched off. *Note: pi-top's Amazon Alexa Voice Service integration uses 16kHz, which is denoted with the red LED being on. In this context, the red LED can be considered as an indicator that the pi-topPULSE is capturing audio.*

Why can't I get my Bluetooth working after connecting a pi-topPULSE?

- This is a known issue, and we are evaluating the best user experience for resolving this issue. In the meantime, this issue is captured here - follow the instructions to re-enable Bluetooth.

Unknown date | **Unknown author**

Facial Recognition Security System

Summary

Someone at the university asked for advice on a low cost, "smart" security system. He wanted to monitor people entering his dorm room but he didn't want a lot of false alarms since his roommate and friends were often in and out of his room when he wasn't there. The solution - use facial recognition to distinguish between people who are authorized to be there and those who are not. If the person is recognized by the security system, he or she is welcomed into the room; if the person isn't recognized, a photo of the intruder is stored in the system for the room owner to review when he returns.

For this project, I wrote an Universal Windows Platform (UWP) application in C# which runs on Windows 10 IoT Core on a Raspberry Pi 3. I connected a Passive Infrared (PIR) motion sensor to detect when someone enters the room. I included a speech synthesizer and added a speaker to announce that a person has been detected. The attached webcam captures an image of the person and sends it to Microsoft Cognitive Services running on Azure for analysis.

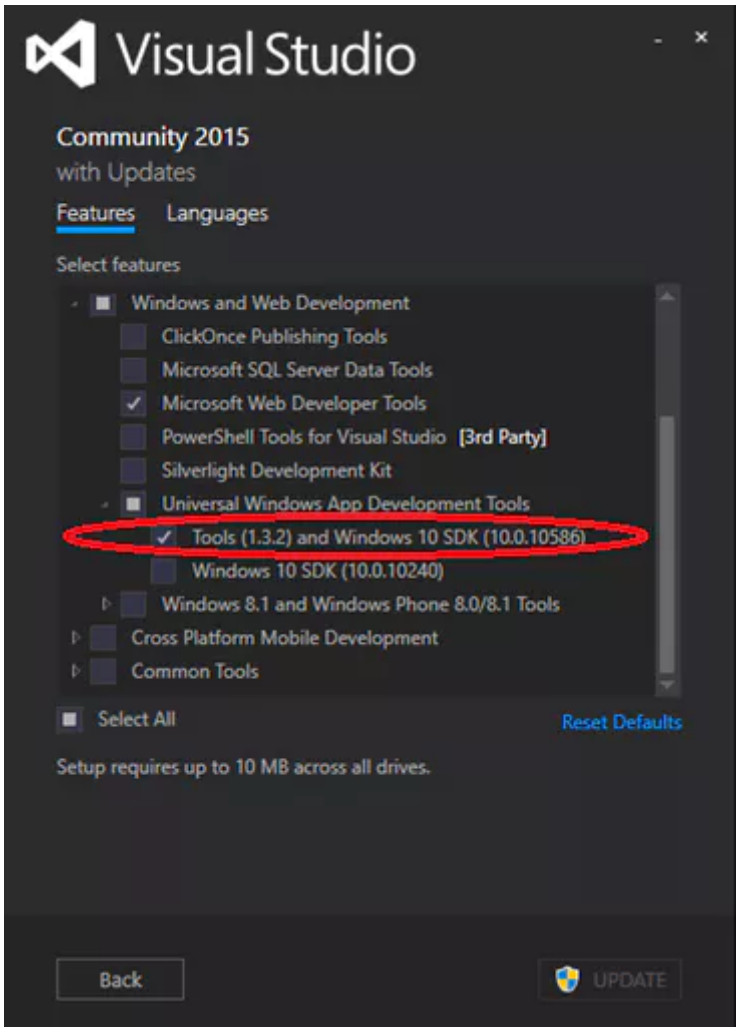
This project was a good introduction to facial recognition and the Microsoft Cognitive Service's Face API made it easier than expected - especially when using their open source SDK. I also resused a lot of the Facial Recognition Door Lock project created by the Windows IoT Team.

Prerequisites

- You'll need to have Windows 10 (version 10.0.10240 or better) running on your development machine.
- You'll need the Windows 10 IoT Core Dashboard on your development machine.
- You'll need to image an SD card with Windows 10 IoT Core - minimum supported OS version for this project is build 10.0.14295.
- You'll need Visual Studio 2015 Update 2.

Installing all the prerequisites could take up to an hour or so but most of that time is unattended while the bits download and get installed so plan accordingly. Microsoft has documented the setup process well in a step-by-step guide.

With respect to Visual Studio 2015 Update 2, you may need to customize your installation (if you are installing it for the first time) or modify your existing installation to ensure that you have the most recent version of the Window 10 SDK installed - the Windows 10 SDK is not installed by default in the the Community Edition and it is required for this project (I believe it is installed by default in the Professional and Enterprise editions).



Update your installation of Visual Studio 2015 Update 2 to ensure that the Windows 10 SDK is installed.

To verify that you completed the setup process correctly use the Windows 10 IoT Core Dashboard; the Dashboard will discover the Pi once Windows 10 IoT Core has booted and connected to the network (you can plug a network cable between the Pi and your network's router or directly into your development machine's Ethernet port if you turn on Internet connection sharing).

Use the device portal to configure WiFi. To open the device portal, go to the 'My Devices' tab in the Windows 10 IoT Core Dashboard and click on the globe under the 'Open in Device Portal column'. You'll be prompted to enter the credentials for the device - "administrator" and "p@ssw0rd" by default. Go the Networking page to setup WiFi.

Components

The following components are required for this project. In the United States, total component costs are less than $75.

- Raspberry Pi 3 (this project works equally well on the Raspberry Pi 2 but you would then need a separate WiFi dongle if you want to go wireless) (you could also use an Intel MinnowBoard Max or a Qualcomm DragonBoard 410c).
- Passive Infrared Motion Sensor
- Supported webcam (I used the Microsoft LifeCam HD-3000 but others are support as well; check the list here)
- Speaker (plug the speaker's 3.5mm audio jack directly into theRaspberry Pi; I'd recommend powering the speak separately)
- Jumper wires

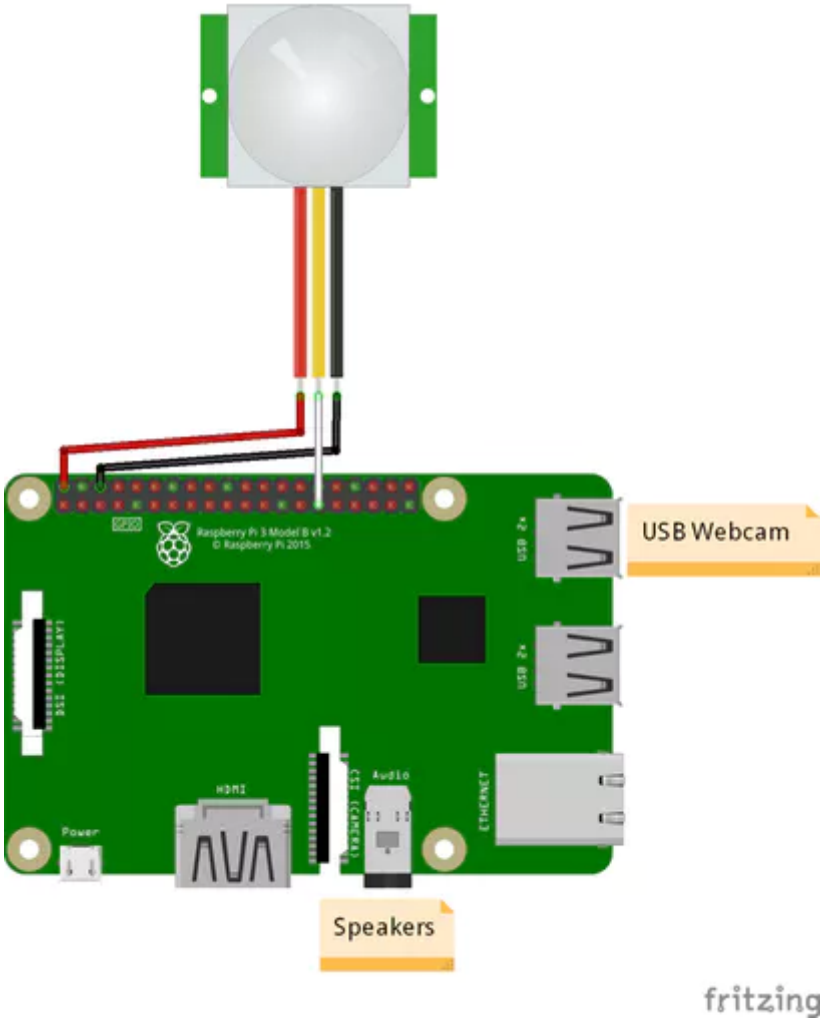
Instructions - Follow these steps

Step 1 - Wire Up the Security System

Time - 5 minutes

The security system is comprised of a PIR sensor, speaker, and webcam connected to a Raspberry Pi 3. The PIR sensor will detect when someone enters your room. The speaker plays the text to speech messages. And, the webcam will capture an image which will be processed by the facial recognition service.

First, connect the PIR motion sensor's pins to the Raspberry Pi using jumper wires - 5V on the sensor to 5V PWR on the Pi (I used pin 2), GND on the sensor to GND on the Pi (I used pin 6), and signal on the sensor to a GPIO pin on the Pi (I used pin 29 / GPIO 5). Double check the data sheet for your sensor to make sure you correctly identify the sensor's 5V, GND, and signal pin as their order varies by manufacturer. (I bent the pins on a female header so that I could mount the sensor upright on the mini breadboard.)



1 / 2 · Plug in your webcam and speakers and wire up the PIR sensor.

My PIR sensor operates at 5 volts which is why it's power pin needs to be connected to a 5V power pin on the Pi. But, it's signal pin operates at only 3.3V which is why it can be connected directly to a GPIO pin. Again, double check the data sheet for your sensor to ensure that the signal's voltage is 3.3V. If it's 5V, you'll have to divide the voltage before connecting it to the Raspberry Pi's GPIO pin.

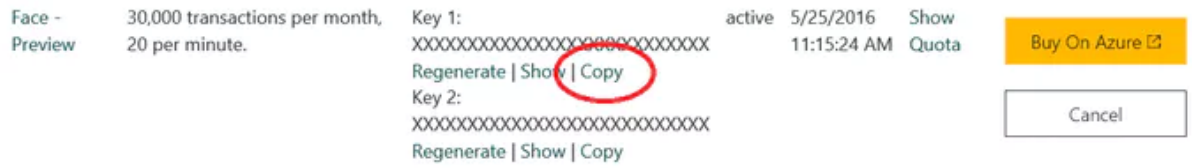
There is jumper on the back of the PIR sensor that lets you select "Single Trigger" or "Repeat Trigger". Ensure that the jumper is set to Single Trigger so that once a person enters your room, he/she doesn't have to put his/her face in front of the camera every few seconds to be recognized.

Next, connect the webcam to one of the Raspberry Pi's USB ports. And, connect your speaker to the Raspberry Pi's audio jack.

Step 2 - Sign Up for Microsoft Cognitive Services

Time - 5 minutes

In order to use Microsoft Cognitive Services, you must have a subscription. There is a free tier for the Face API which is more than sufficient for this project (the free tier allows 30k transaction per month). Sign up here. Make sure you check Face API when setting up your subscription. Once you have setup your subscription, copy the key for the Face API - you will need it in the next step.



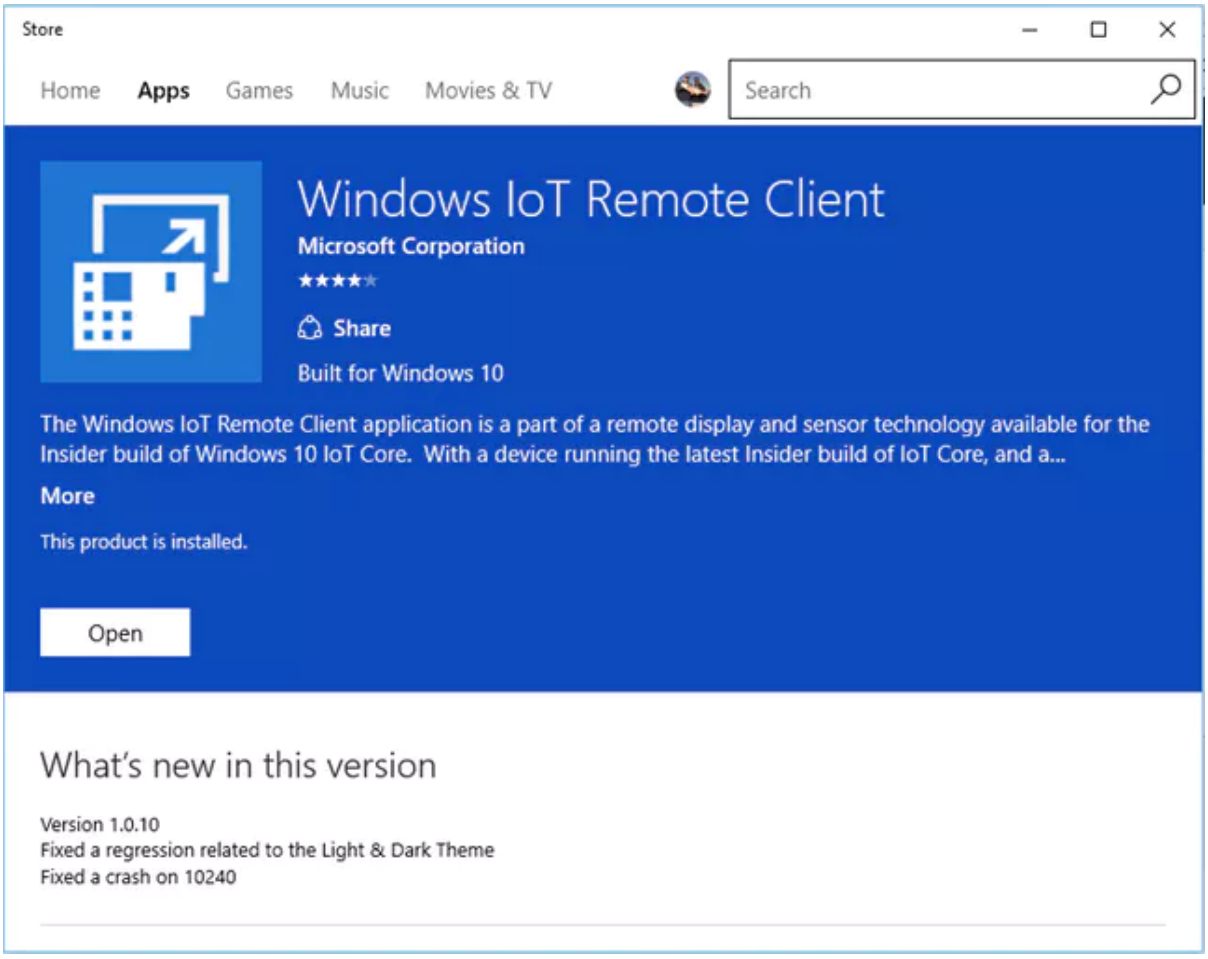
Copy your subscription key for the Face API as you'll need to add it to the app.

Step 3 - Add a Display, Keyboard, and Mouse

Time - 5 minutes

This project requires a display as you'll need to interact with the UI to setup facial recognition and to review photos of intruders. If you have an external HDMI monitor, keyboard, and mouse, connect those now and move on to step 4. If not (I only have laptops and don't have any separate displays), you can use the remote display feature.

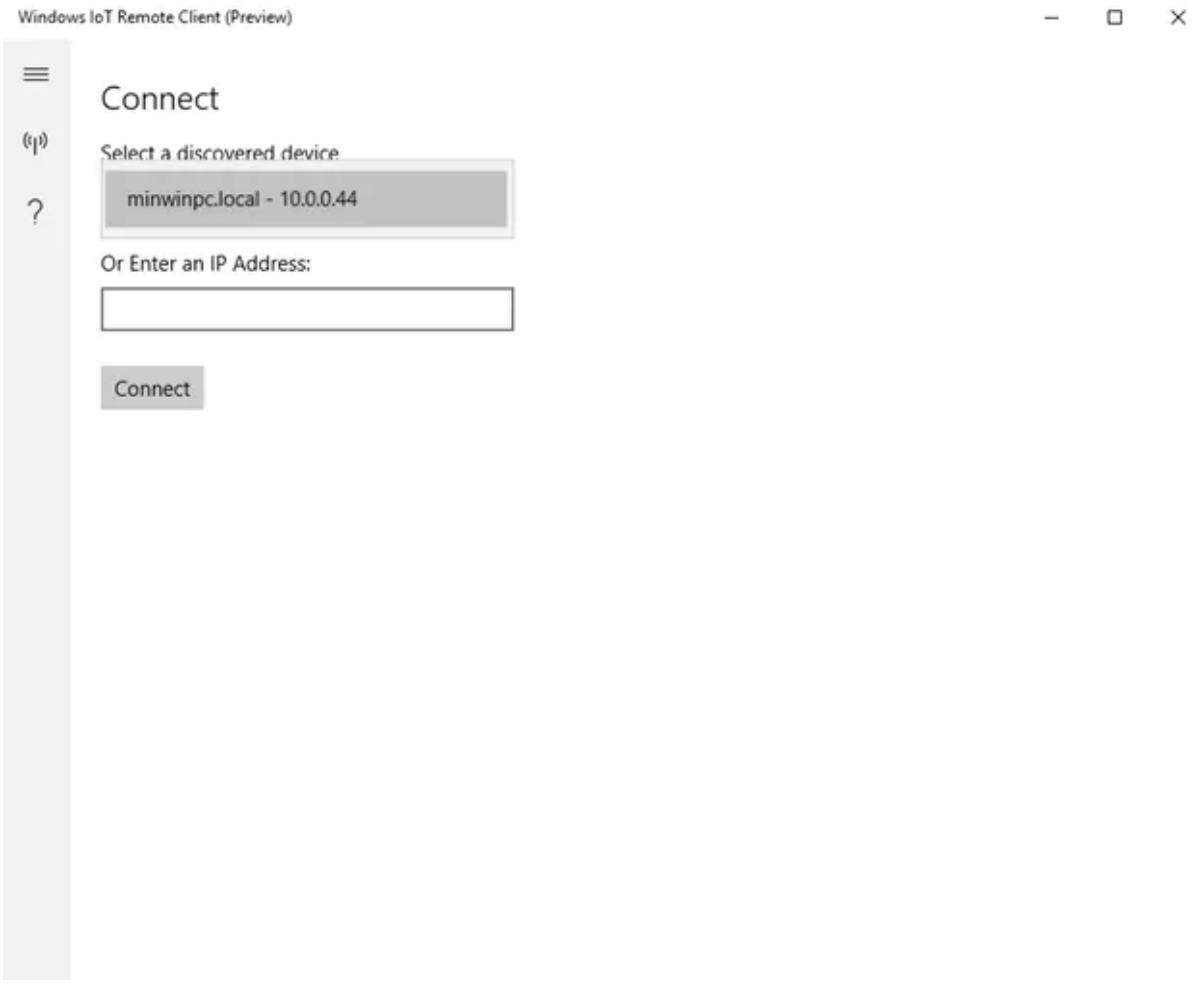
The remote display allows you to remotely control your UWP app running on Windows 10 IoT Core. You can use the display from any of your other Windows 10 devices (phone, tablet, laptop, desktop) to view the app's UI and the keyboard, mouse, and touch of your Windows 10 device to interact with and to provide input to the app. You'll just need to get the remote display app from the Store and install it on your Windows 10 device - here is the link to the Store. The only limitation is that the Windows 10 device need to be on the same network as your Raspberry Pi running Windows 10 IoT Core.



1 / 2 • Download the Windows IoT Remote Client app from the Store onto your dev machine.

Once you have install the remote display client on your Windows 10 device - i.e. your development machine - you'll need to log in to the Raspberry Pi and enable the remote server. Open the Device Portal in your browser: navigate directly to the Raspberry Pi (http://{Pi's IP address}:8080) or open Windows 10 IoT Core Dashboard, go to the 'My Devices' tab, and click on the globe under the 'Open in Device Portal' column. Enter the credentials for your Raspberry Pi and then click on the 'Remote' menu option. On the Remote Server page, click the check box to enable the Windows IoT Remote Server.

Now, launch the remote display client app, select your Raspberry Pi from the drop down, and click the 'Connect' button. The display from your Raspberry Pi will be shown. You'll notice that you can now use the mouse from the machine running the remote display client to interact with the Raspberry Pi.



1 / 2 • Launch the remote display client and select your Raspberry Pi from the drop down.

Step 4 - Deploy the App to the Raspberry Pi

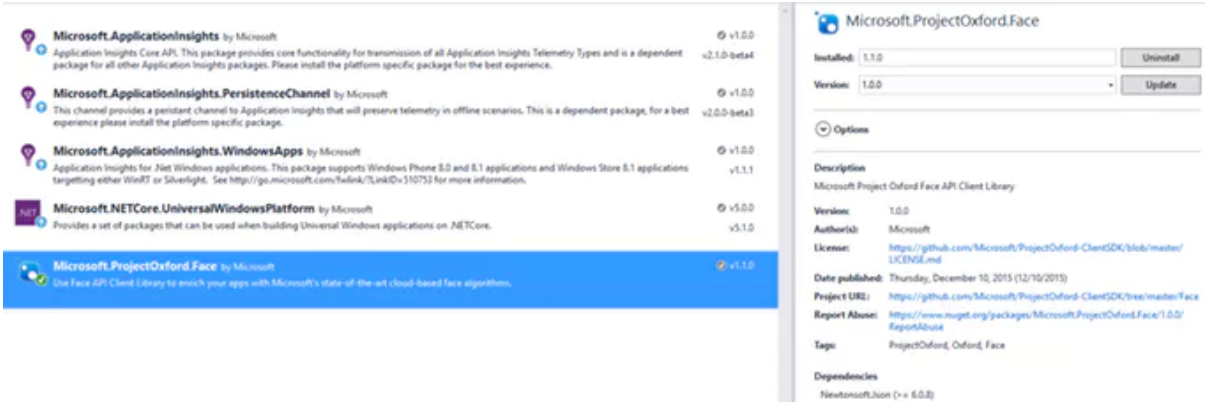
Time - 10 minutes

Download the code from the GitHub repository and open the solution in Visual Studio 2015. Before building the project, you will need to update the GeneralConstants class with your Face API subscription key. (Microsoft Cognitive Services was formerly known as Project Oxford so you will see several references to Oxford - especially in the client libraries.)

```
public const string OxfordAPIKey = "";
```

The solution should automatically pull in the all references required for this project including the Microsoft.ProjectOxford.Face NuGet package. The

project was built using version 1.1.0.



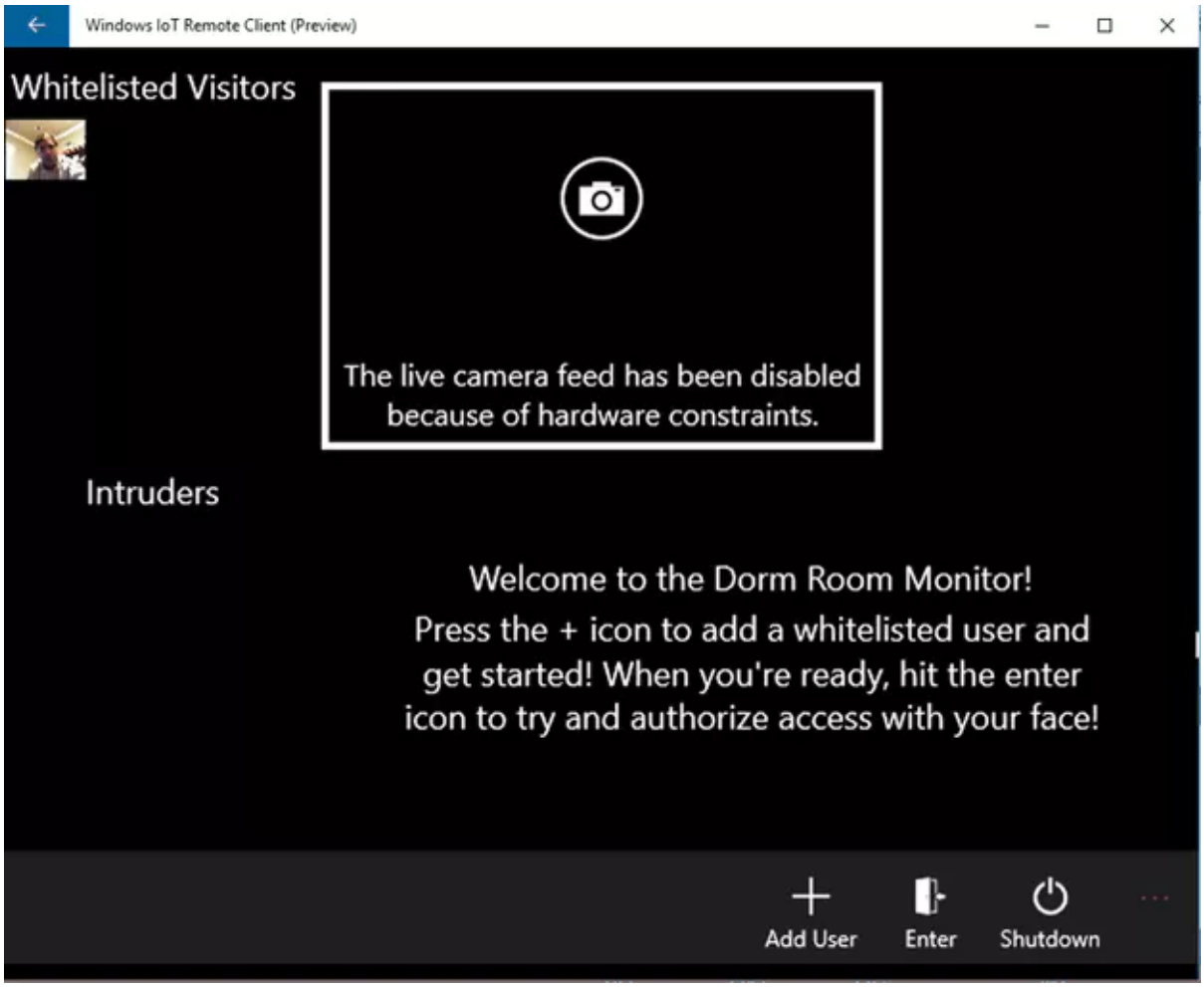
Now, build the project and deploy the app to the Raspberry Pi.

Step 5 - Create Your Whitelist

Time - 5 minutes

When someone enters your room, the PIR sensor detects their presence and takes a picture with the webcam. The app then uses Microsoft Cognitive services to detect the face in the picture and to attempt to recognize it. The face in the picture is compared against the list of people approved to be in your room - i.e. the whitelist. You must first define the whitelist before anyone can be recognized.

From the app's main screen, click the '+' icon at the bottom to add someone to the whitelist. The next screen will allow you to take a picture of the person and enter his/her name. Then, click 'Confirm'. Now, that person has been added to the whitelist.



1 / 4 • Click the 'Add User' button to add a new person to the whitelist.

However, to improve recognition, it's recommended that you have additional images for each person. So, from the main screen, click on the person's face in the 'Whitelisted Visitors' section. Now, click the 'Add Photo' button to capture additional images of the person's face. When finished, clicked the 'Go Home' button to return to the main screen. You can also use the 'Delete User' button whenever you want to remove someone from the whitelist entirely.

Once you've added everyone who should have access to your room to the whitelist, you are ready to start monitoring. The main screen has an 'Intruders' section, This is where you will see photos of anyone who entered your room and who wasn't recognized. If you want to access the images separate from the app or what to copy them to some other location, you can find them on the SD card at this location:

```
C:\Data\Users\DefaultAccount\Pictures\Dorm Room Monitor
Intruders
```

Platforms

This app can also be run on your development machine (or Windows phone) directly - it really is an Universal Windows Platform app. (I found this helpful as I started to develop the app on my laptop while I was waiting for the Raspberry Pi 3 to arrive.) If you run it on a non-Windows IoT device, you will not have GPIO pins with which to connect to the PIR motion sensor. Therefore, there is a button on the main screen's UI to "trigger" the motion sensor - just click on the 'Enter' button.

In addition to the Raspberry Pi 2 and 3, Windows 10 IoT Core is supported on the Intel MinnowBoard Max and the Qualcomm DragonBoard 410c. Both the MinnowBoard Max and the DragonBoard 410c have GPU support in Windows 10 IoT Core. Therefore, the app UI can display the live stream from the webcam. To enable live video on either of these devices, change the DisableLiveCameraFeed variable to 'false' in Constants.cs:

```
public const bool DisableLiveCameraFeed = true;
```

---

2 years ago | **Unknown author**

Personal Mirror

The Personal Mirror

Motivation

The idea to create a mirror with additional Information is not new and there are a lot of projects with instructions in the web to create such a mirror. (e.g. here: http://michaelteeuw.nl/post/80391333672/magic-mirror-part-i-the-idea-the-mirror)

My idea is to create a mirror, which is able to recognize your face and brings up additional information which is interesting for you in case you are in front of the mirror. Maybe your calendar entries of the day or the last tweets or something else.

Almost everybody knows the app "How old are you" from Microsoft. It was written as an example of the face API (Project Oxford) and is provided as Azure Service from Microsoft.

Everybody can use it without additional costs (there are some restrictions with the number of usage (accesses per second and total accesses per month are limited), but this should be not relevant for such a project.

Current Status

The project is a work in progress and unfortunately not finished until the deadline of the Microsoft IoT Challenge. 4 weeks were definitely not enough

time for me to finalize this project from the idea to the working sample besides my daily work.

Nevertheless I will keep the work running and update this Project mainly on a weekly or 2-weekly base.

Mechanical Construction

The mirror cabinet (frame) was made from pine wood. The wooden strips are glued and screwed. To have a very good looking mirror you will need a minimum of tools for wood working and some experiences. I asked a friend, a cabinet maker for the job and was very happy with the result.

Cabinet for the Mirror

The mirror glass is semitransparent and a little bit darker than a regular mirror. Behind the mirror glass, the monitor is mounted and you can read the bright parts of the screen through the mirror glass. The mirror glass itself is not cheap, I paid nearly $110 at a local glazier's workshop. Buying from Internet is a little bit cheaper, but the delivery time is 3 weeks up (here in Germany, maybe this will vary in other Countries)

I use a 24" Monitor from Benq, I think 24" is a good size with reasonably value for Money. But you can decide if you want to have a bigger or smaller

Screen. It's important to use a Monitor with HDMI Input to connect it with Raspberry Pi.

For the Ultrasonic Range finder there must be an opening in the Frame.

Opening for the Ultrasonic Sensor in wooden frame

Electrical Construction

The Ultrasonic Sensor SRF02 is connected with the Raspberry Pi module with a flat ribbon cable, for the wiring connection please see the diagram below.

The monitor is connected to the Raspberry with a HDMI cable, all other parts are connected via USB connectors (1 for the Webcam and 1 is for the WiFi dongle). You will need a power supply with Micro USB jack and 5V, 2Amps.

The Ultrasonic Sensor is for detection if a person is in front of the mirror. If nobody is there, the monitor can power down to safe energy.

Software

Bringing Windows 10 IoT core to the Raspberry is very easy. There is a good step by step tutorial here: http://ms-iot.github.io/content/en-US/win10/SetupPCRPI.htm

There's also no Problem to install and prepare the Visual Studio 2015. On the Microsoft site http://ms-iot.github.io/content/en-US/win10/StartCoding.htm you can download a lot of samples for the Raspberry Pi which easy to use and work out of the box.

A good starting point for the mirror project is the Webcam example. http://ms-iot.github.io/content/en-US/win10/samples/WebCamSample.htm . For me it was much fun to see the same code working on a Raspberry, on my Notebook and on my tablet.

The next step was to integrate the Azure Face recognition Service. Here began the hard part of the work and it's still ongoing.

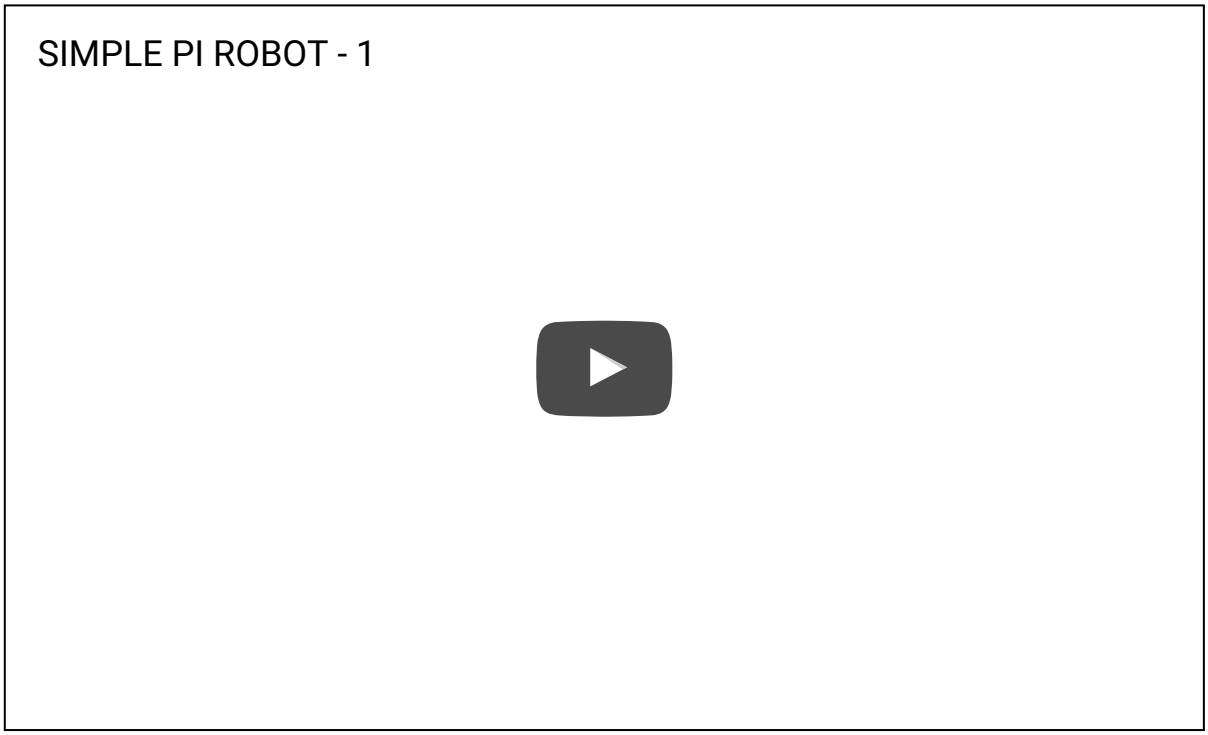
I followed the getting started tutorial on the Azure Website: https://www.projectoxford.ai/doc/face/Get-Started/csharp . You will need an special account for this project, it's a little bit tricky, but good described in the paper.

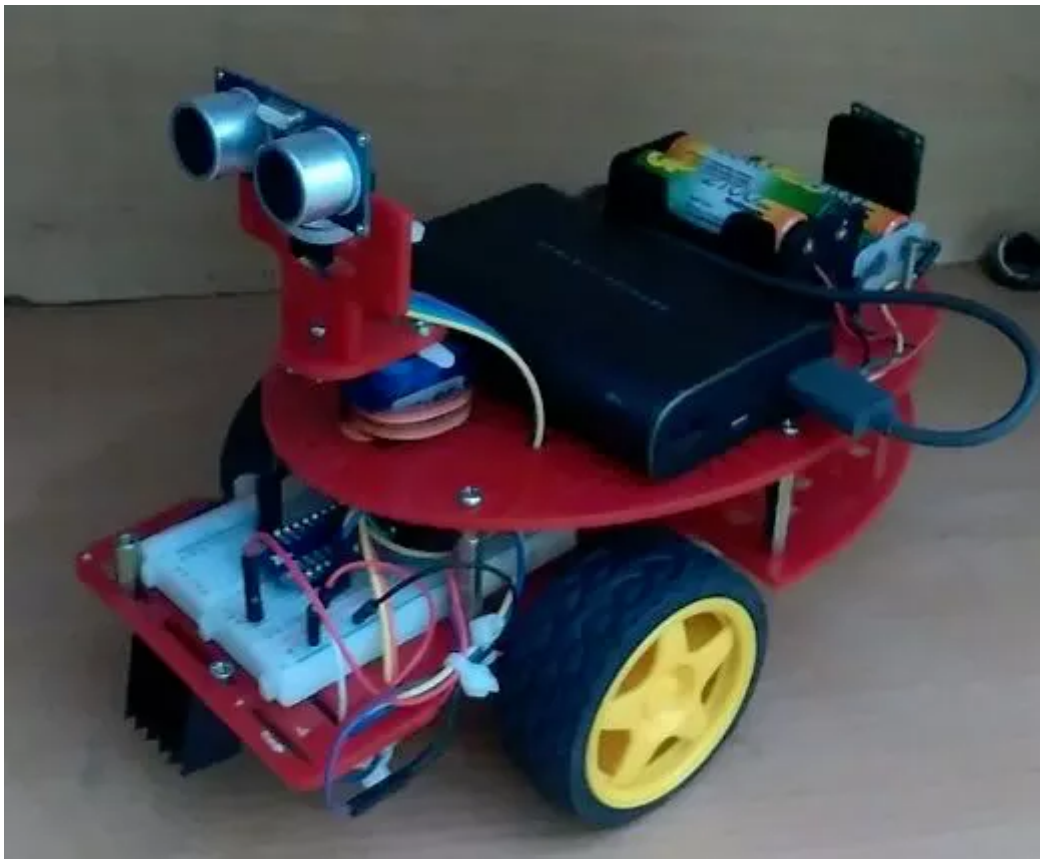
The next drawback for me, the paper uses the WPF libs, for the Universal Apps it has to be replaced by the Windows Runtime approach and this is a big challenge - at least for me.

When the first version of Face recognition is ready I will deploy it here in source code area.

Unknown date | Unknown author

Simple Pi Robot





Simple Pi Robot aims to put robot control in simple form.

The part list

(1) Raspberry pi (Any model) but with the recent launch of pizero or pi 2 shall be a good option, my current model employs B+.

(2) 40 pin GPIO cable (if using pi B+ or pi 2).

(3) Breadboard (for quickly assembly various sensors).

(4) A 2wd chassis.

(5) Distance sensor (Ultrasonic HC SR04).

(6) Power bank (for powering up the pi).

(7) Rechargeable batteries AA (preferably 2100 mAH).

(8)Jumper wires both Male & female type, Resistors.

(9) WiFi adapter (EDUP/EDIMAX for communicating wirelessly with pi).

(10) Memory card (4GB and higher for running the OS on pi).

(11) Motor drivers (L298).

(12)servo motor.

(13) Miscellaneous - cable ties (for tying the jumper wires) and foam tape (for holding servo or any other sensor where screw assembly cannot be used).

Step 1: Choosing the Motor drive shield

Currently there are very few motor drive shield available for raspberry pi, to name some:-

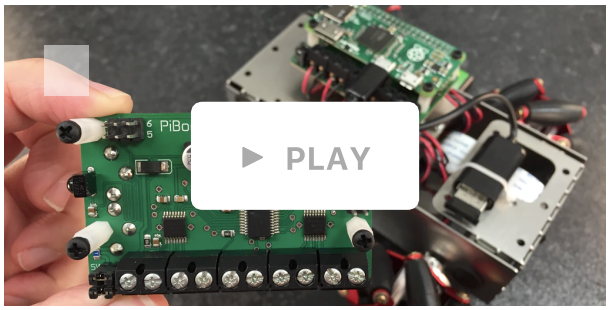
(1) RTK Motor controller shield

(2) Pololu DRV8835 Dual Motor Driver Kit for Raspberry Pi

(3) Adafruit DC and Stepper Motor HAT for Raspberry Pi

(4) Raspirobot board made by Adafruit

and recently underdevelopment like ZEROBORG



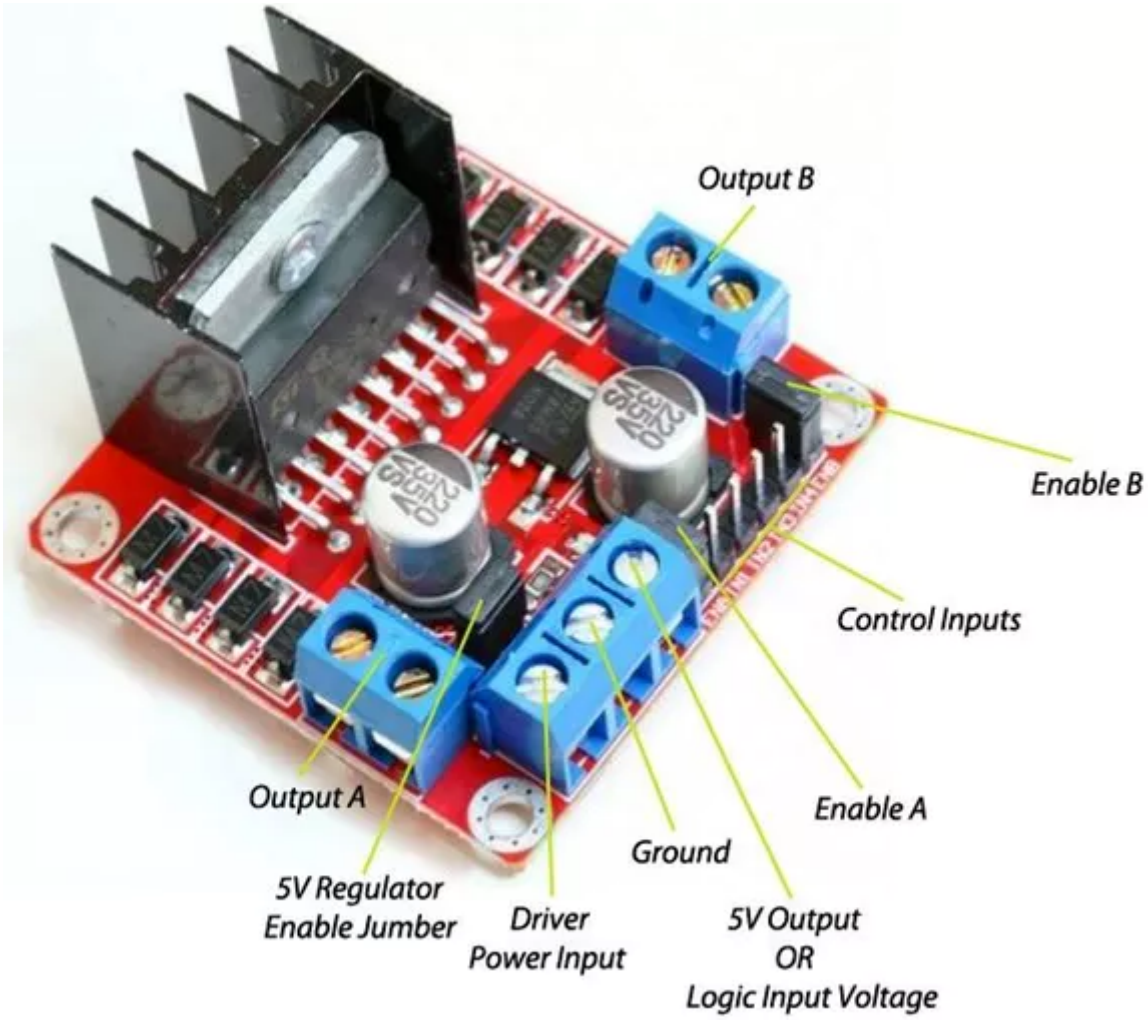
One of the most common problem building any robot, is to minimize the wiring requirement and the same can be achieved by using the shields/hat. I tried building my robot, first with one of the shield equivalent to Raspirobot from the manufacture ALSROBOT. The kits ships from china but the problem was i was unable to increase the motor input voltage. The maximum was less than 5 Volts, with slight unbalance between the voltages, still one can check the following link - ALSROBOT - PI Motor driver shield

Anyhow In my current tutorial i have used the cheap and versatile L298 motor driver - L298.

The advantage with the above board apart from being cheap, is that there is regulated 5Volts output is available.

I have used the L298 board to drive two DC motors and 1 servo motors.

Step 2: The L298 Motor Driver



I have Mounted L298 Motor driver at the bottom of my chassis, now to connect the DC motor and servo motor connect

(i) DC Motor -A to output -A (+ & -).

(ii) DC Motor -B to output -B (+ & -).

(iii) Servo motor +ve to + 5V regulated supply of L298 Board & servo motor -ve to GND of L298 Board.

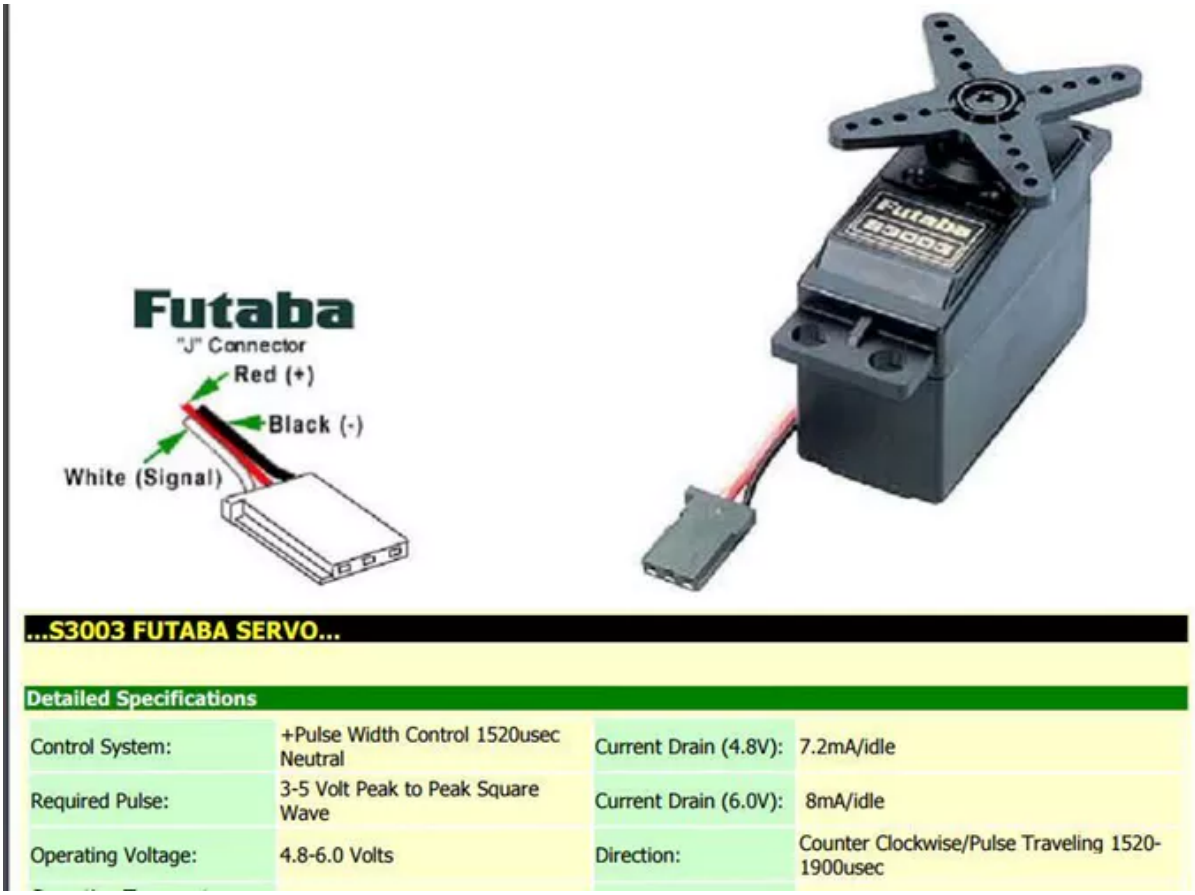
Keep the enable pin jumper as it is, if you don't want speed control, else remove the jumper. The jumper in place ensures +5V supply to enable pin

which in turn drives motor at rated speed.

Now connect 4 nos jumper wires to the control inputs, connect other end of jumper wires to GPIO pin as designated in next step.

For servo motor connect one no. jumper wire for control to GPIO pin as in next step

Step 3: The servo Motor



1 / 2

The servo has a 3 wire connection: power, ground, and control. The power source must be constantly applied.

The control signal is pulse width modulated (PWM), but here the duration of the positive-going pulse determines the position of the servo shaft. For instance, a 1.520 millisecond pulse is the center position for a Futaba S148 servo. A longer pulse makes the servo turn to a clockwise-from-center position, and a shorter pulse makes the servo turn to a counter-clockwise-from-center position.

I have used Futuba s3003 servo - the connection is very simple the "+" & "-" goes to the L298 Board as outlined earlier. It's important to look at the operating voltage of servo (in my case its 4.8 - 6 V see image above), the signal wire is to be connected to GPIO output , normally its white or orange.

Controlling servo motors in raspberry pi might be tricky, but then there is very powerful library hosted @ RPIO.PWM ,to install it on pi use the following code.

```
sudo apt-get install python-setuptools
sudo easy_install -U RPIO
```

To know more about RPIO.PWM and the DMA used , please refer to the link https://pythonhosted.org/RPIO/pwm_py.html

Step 4: The Robot chasis

SIMPLE PI ROBOT - II

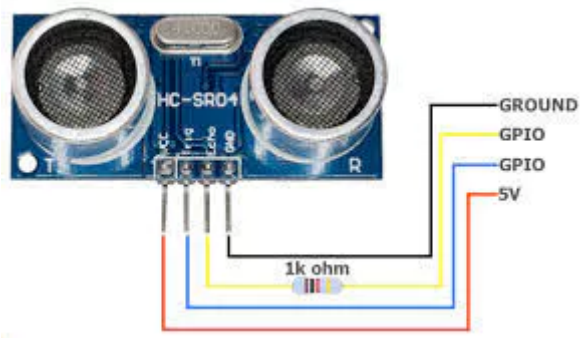
I have used Ellipzo Robot chassis with 2wd, 2wd are simple and easy to control.

The kit comes included with DC Motors, pan kit for servo motors and all necessary hardware's to assemble the kit.

The detailed link is available at - Ellipzo Robot chaisis kit.

Pl. refer to video for assembly of Raspberry Pi, along with L298 motor driver, the Breakout board, camera module and the power bank.

Step 5: The Distance sensor



Integrating the distance sensor is easy and we need just 1k resistor, along with jumper wires. Connect VCC & GND to pi +5Volts & GND respectively.

The other two pins TRIG & ECHO are to be connected to GPIO pins as in preceding steps. Remember to connect the resistor as shown in image.

The Python code to measure distance is included in the last step.

Step 6: Raspberry Pi -camera - Video streaming using VLC player

Here I have used Pi camera module, the set up is quite easy and you can refer the link :Raspberry pi Camera setup

For video streaming, with VLC begin with installing VLC on raspberry pi

```
sudo apt-get install vlc
```

To start streaming the Camera Video Using RTSP enter following

```
raspivid -o - -t 0 -n | cvlc -vvv stream:///dev/stdin --sout '
```

or with proper width & height use following code

```
raspivid -o - -t 0 -n -w 600 -h 400 -fps 12 | cvlc -vvv strear
```

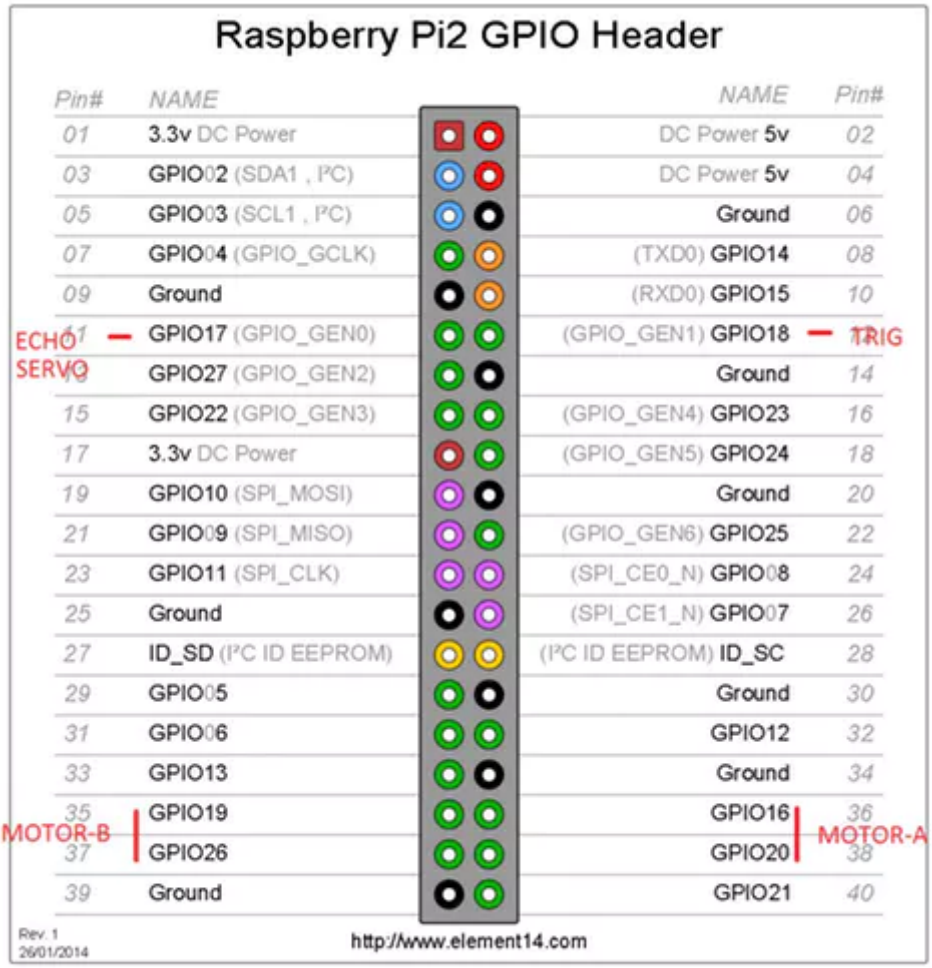
Now to view the stream over VLC player open VLC on your remote system than open a network stream using

rtsp://###.###.###.###:8554/

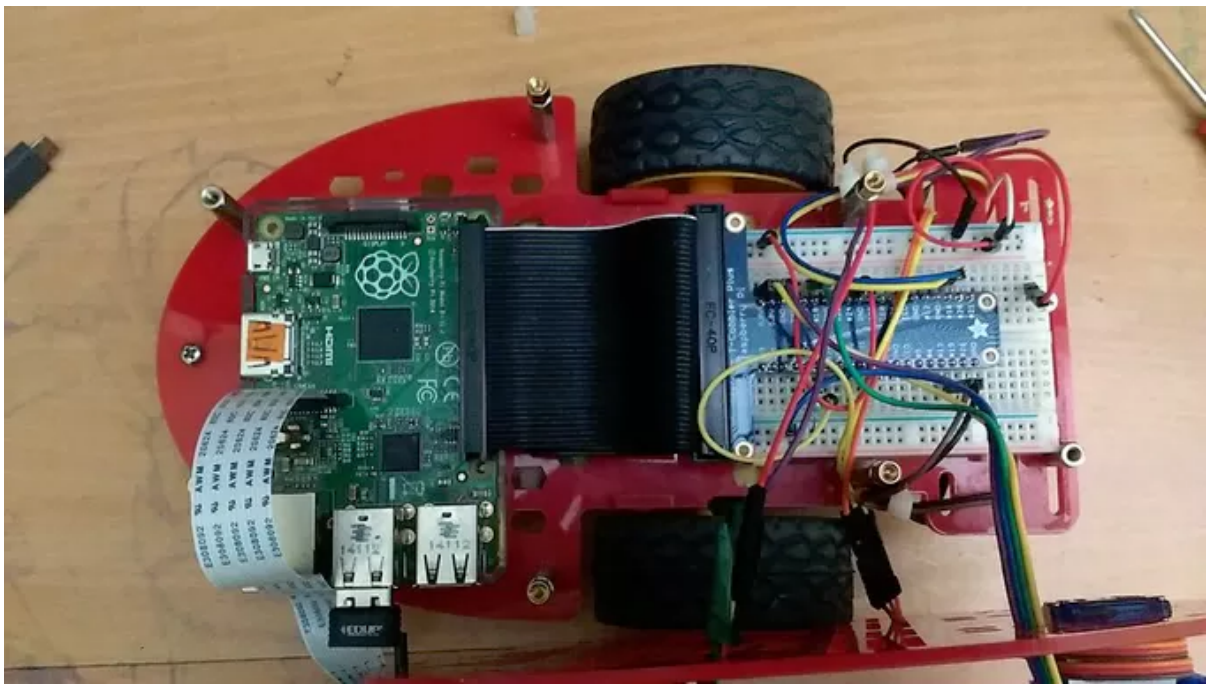
where ###.###.###.### is the address of your pi given by the network router.

Now as the pi moves inside your home see the video stream on your remote system.

Step 7: Raspberry pi pin-out & python code



Step 8: Some Assembly Images



1 / 4