Automatic liquid dispenser with facial recognition

Annoy-o-tron - "Bartholomew"

Contents

1 Introduction

This project focuses on the more fun, lighthearted aspect of things. There is no self-driving AI (though there should've been one), only a tiny little robot on wheels that will release liquid when it sees a person, much like a dog would when it sees its owner come back.

The project uses an Arduino Nano and a Raspberry Pi - rather, clones of them. The "Nano" is used for most of the electrical components due to an issue with the Pi clone, and as such will practically be 90% of the project.

2 What is its purpose?

To apply what I've learned, and to have a bit of fun experimenting, I decided to go for the less serious route in robotics.

I decided to make this robot mainly a fun nuisance in the living room. If you already have a pet dog, you might be familiar with what happens if it gets too excited.

The upsides of Bartholomew are that it can't bite, destroy your furniture, or play fetch. The downsides are that it's incapable of hearing you, only responding to your presence.

If you get too close to it, you'll have a watery mess to clean up.

3 Components

As mentioned, the project uses a clone of both a Nano, and of a RPI - one called Banana Pi M2 Zero. I chose the BPi because it allowed me to use facial recognition software and a finer control of itself and the things connected to it. Unfortunately, I ran into several problems, which I will detail later on, but for now, it's best to say that it's nothing more than a facial recognition camera, despite its GPIO pins.

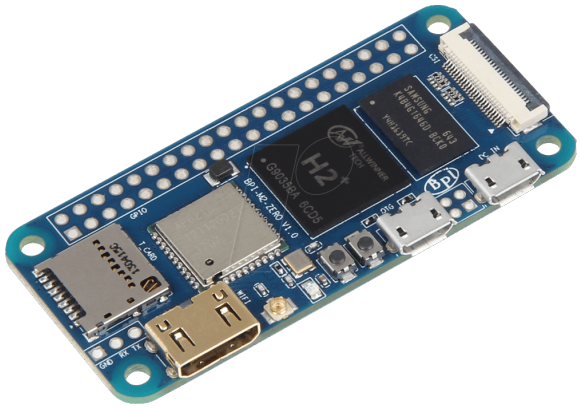

Figure 1: BananaPi M2 Zero

The Nano is an instrument for connecting the sensors and actuators to it, as well as receiving signals from the Pi.

Other components used are:

- Generic:
  - 330Ω resistors
  - red LED light
  - NPN transistor (2N2222)
  - jumper cables, M-F and M-M
- Sensors
  - HC-SR04 ultrasonic distance sensor
  - Water level sensor
  - OV5640 camera for the Pi
- Actuators

– Mini water pump

3.1 How and what for the components are used

Going over them as they were listed, the resistors were mostly used for preventing overvoltage (in the case that it occurred) for the smaller components and those without internal protection - mainly the LED and the water level sensor.

The distance sensor is used to measure the distance between Bartholomew and the object its facing. I have estimated that the ideal range - based on the pump's capabilities - is between 10-50cm. Once the object is in range, the robot has to also check if a face is also in view.

The water level sensor within the container is a great way to make sure that there is enough water so that the pump doesn't do a dry run and possibly break. The LED is a warning sign that there isn't enough water in the container, which is "calibrated" to around the height of the pump itself.

The camera is only used for facial recognition within the Pi. While a powerful camera, its resolution is cut down to 480p due to the hardware limitations of the Pi.

Finally, the water pump is activated with the use of the NPN transistor, when all 3 conditions are met: there is a face, a person is within 10-50cm, and there is enough water within the container.

These can be seen in the following figure:

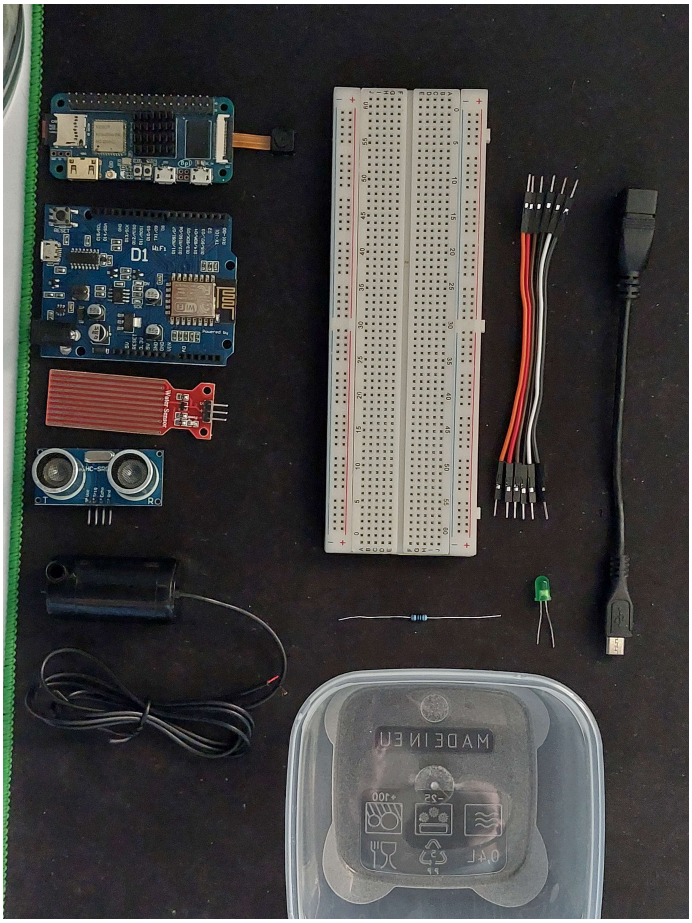

Figure 2: Components used

Some other items are the USB converter (for the Pi), a water container, and not among these is a glue gun, and a 15cm soft tube.

There were supposed to be both a wheel assembly and a solenoid pump, but seeing how they didn't arrive on time, I had to finish it with what I had so far.

4 Working process

I started off with the brains of the project - the microcomputer and microcontroller. In total, this project took me around two weeks from start to finish, if we're to disregard the time spent waiting for parts to arrive.

4.1 The Banana Pi

For the Pi I had to flash a specific version of Armbian that supports the OV5640 camera, as without it, this project will lack the original vision of employing AI. The version comes with many pre-installed packages and built repositories for ease of use and access.

The main package that I needed to use for the facial recognition was OpenCV lite. There were examples given as part of the ISO, and with these - along with some help from the internet - I was able to create my own simple, yet effective facial recognition software. The same software is also tasked with sending a simple 1bit signal through USB to the Nano to make sure that it knew if there was a face.

4.2 The Nano

After this was done, I moved onto the Arduino. At first there was a Wemos R1 D1 board, but after a mishap with the water, it was left in a practically unusable state. Luckily I had a Nano clone lying around, and it did the job just fine.

The pins I used are all located in the source code, as well as their explanation on how they are connected.

The portability of the Nano will be more useful in the long term, as its small form-factor takes up little space and doesn't require anything other than a 5V power supply to run it and all the components connected to it.

4.3 The assembly

To finish up the project, I made a hole at the bottom of the container, and cut up a part of the lid to allow the cables to go through. Inside the container, I glued the water level sensor and the mini pump to the sides, and pushed through a 15cm soft tube through the hole, sealing it with hot glue.

The last thing needed was to somehow attach all of this to a vehicle platform, and what better option than an remote controlled toy car, especially since the wheel chassis never arrived.

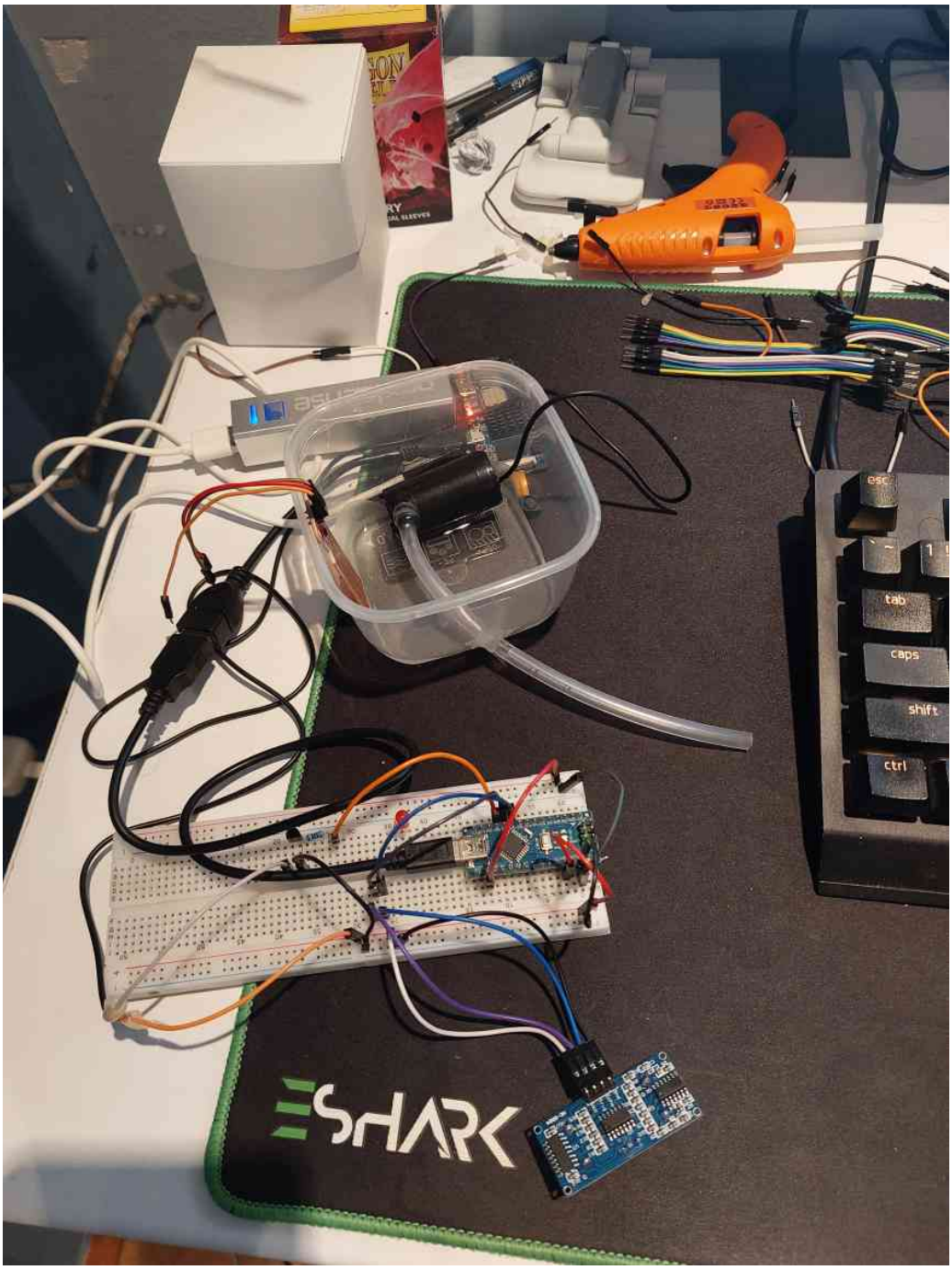

Figure 3: The work in progress mess

4.4 Problems encountered

4.4.1 The BananaPi M2 Zero

This microcomputer has caused me the most inconvenience of all. From having a Mini HDMI port, to requiring a specific connector layout for the camera, to improperly powering its pins (they all read at 1/10th of the voltage they are supposed to supply, even the 5V and 3.3V).

This was not the end of the problems encountered. The aforementioned OpenCV inclusion was not built to my requirements. It lacked the visual recognition component, so I had to rebuild it from scratch.

Unfortunately, the z-ram configured for this image was too low for me to use to recompile and install OpenCV, so I had to redo that. That, again, was met with another problem. As it was using an SD card, it would allow me to essentially use half of it as swap memory. It was a great idea until I realized that it was only using 16 of the 32 GB available, as the OS only had 16GB allocated by default. This was quickly resolved by opening up Gparted and adjusting the partitions accordingly.

After this, I was finally able to install and use OpenCV, but the C++ files didn't recognize the headers. By default, these headers should go to /bin/, but because of the way the OS was built, they instead had to be installed under /usr/bin/.

Finally, after jumping from issue to issue, I made an attempt at connecting the GPIO pins to my components. When I found out that there was too little power being passed through them, I switched over to the Arduino to connect these and finish the project.

That however, wasn't the end of my problems with it. For some reason, the Pi program hangs after opening the USB port at times. The cause is unknown, but I believe that it enters a blocking mode. The camera also refuses to open, despite following the instructions and making sure everything is setup properly according to the documentation of the OS.

4.4.2 The Arduino clones

At first, I used a more powerful Lolin WeMos R1 D1 with an ESP8266 attached to it. It had some problems with the driver and baud rate at the start, but doing some reinstalls of the driver allowed me to use it normally each time it would suddenly stop.

During my testing with it, I managed to damage the board with water, as I had forgotten to remove a line of code that allows the pump to work without any other checks, causing me to use a Nano instead. The Nano turned out to be a far lesser hassle than the D1, not requiring me to install any drivers, and was capable of the same functions - and has more pins readily available.

4.4.3 Sensors and actuators

A minor issue that happened was that I had incorrectly wired the higher voltage components to the 3.3V output, which caused me to spend more time than necessary trying to test for shortages. I resolved this swiftly and everything was put back in working order.

Additionally, I had some issues with the transistor, but after seeing its schematic, I realized that I had it turned the opposite direction, which didn't allow for any flow to the pump.

4.4.4 Missing components

As mentioned, I had also ordered a small chassis with wheels to allow automatic wandering and target seeking, but unfortunately as it never came (and that is primarily the reason why it took this long to send this in).

The lack of the solenoid pump was not as big of an issue, as I already had a replacement lying at home. Though less powerful, it still proved to be quite useful.

The idea was to have the robot go into 3 modes:

- Seeking - where it would actively wander around the area, looking for potential targets
- Release - the robot would release liquid when the conditions were met
- Retreat - the robot would then drive in reverse and attempt to hide and wait for some minutes, then restart from the beginning

5 Showcase

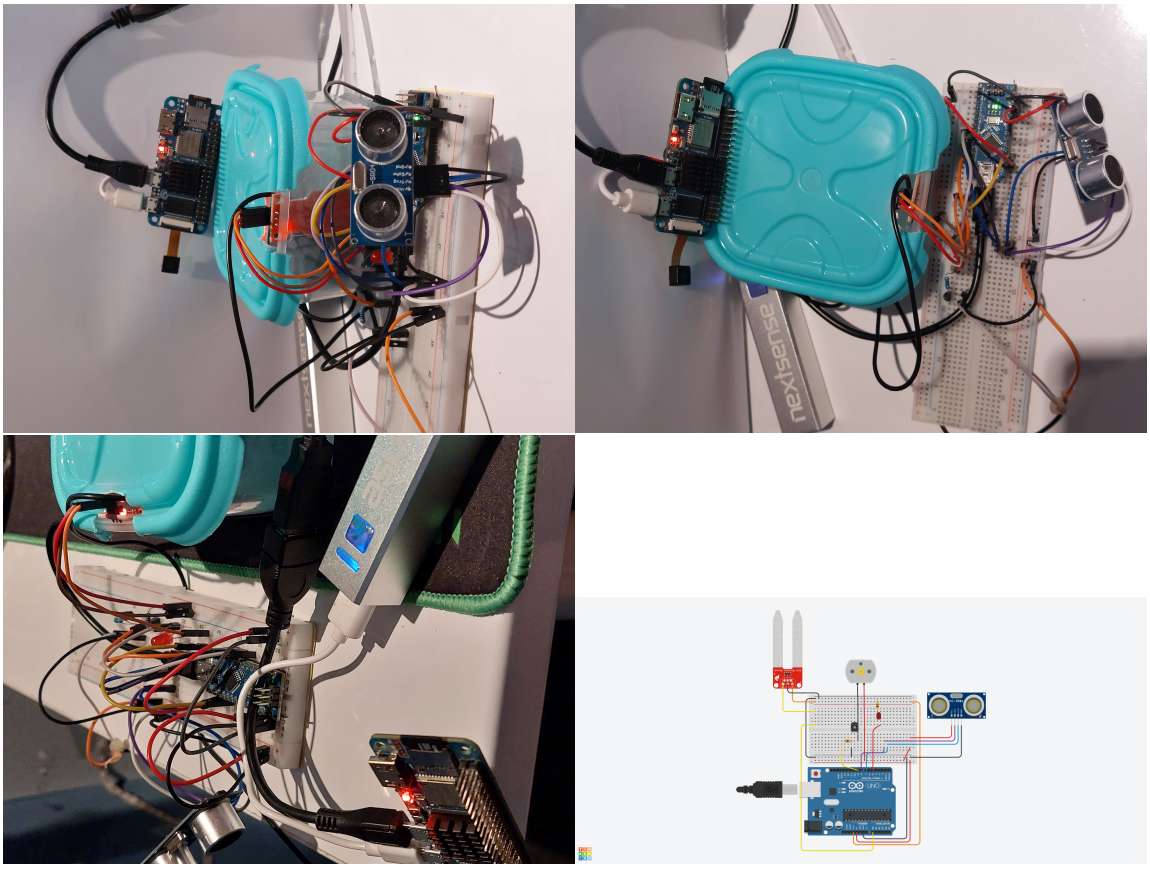

Figure 4: Full showcase of the robot

6 Strengths, weaknesses and extra

6.1 Strengths

The robot is capable of accurately discerning human faces and is able to correctly measure the distance between itself and a person. Its several checks for the multiple conditions it needs to properly function make it robust enough for occasional usage.

6.2 Weaknesses

Damage from the elements could happen easily, as it's not quite housed within an enclosed space.

The way the water tube is positioned essentially creates a miniature version of the Pythagorean cup, causing all the water to spill out after it meets a certain threshold.

6.3 Possible changes and additions

Until I wrote this, I hadn't thought of the possibility of adding an accelerometer for finer control over the position and orientation of the robot, but as I'm using an RC car as the body as a substitute, it doesn't need it.

As for anything else, it has everything that I need it to do, so I have no plans to add anything on top of it.

6.4 Extra

One of my biggest inspirations as to why I made this project is Michael Reeves. He creates unorthodox and unique robots and other creations. I have linked his YouTube channel at the end.

The video of the robot in action is also at the Links section.

7 Links

- Transistors: https://www.aliexpress.com/item/32856312651.html
- Various components: https://www.aliexpress.com/item/1005002739136953.html
- Water sensor: https://www.aliexpress.com/item/1005006022454236.html
- Jumper cables: https://www.aliexpress.com/item/1005003641187997.html
- Breadboard: https://www.aliexpress.com/item/1005004841243658.html
- Nano clone: https://www.aliexpress.com/item/4000145286231.html
- Banana Pi: https://www.aliexpress.com/item/1005004403536632.html
- OV5640 camera: https://nl.aliexpress.com/item/32660117929.html
- Wheel chassis: https://www.aliexpress.com/item/1005001576493929.html
- BPi OS: https://github.com/Qengineering/BananaPi-M2-Zero-OV5640
- Source codes: https://github.com/zodiuxus/bartholomew
- Michael Reeves: https://www.youtube.com/@MichaelReeves
- Bartholomew in action: https://youtu.be/ZfNanCZel_Y