Embedded Systems Engineering
Project - Part A: Quadcopter Technical Review

R00159222 - Zachary Dair
zachary.dair@mycit.ie

Introduction

In this report, we aim to outline the overall hardware specifications and the firmware design of the quadcopter, manufactured by *"Parrot"* known as the *"Rolling Spider Drone"*.

Above is the "Rolling Spider" drone, with and without wheels

The *"Rolling Spider"*, is one of the drones that appears in the line of *"MiniDrones"* by *"Parrot"*, this specific subsection of quadcopters are designed to be portable, ultra-compact with amazing stability and speed, for usage both indoors and outdoors.

Piloted through an app called *"Freeflight"* or *"Freeflight Mini"* on most modern smartphones, provides the ability to fly the quadcopter without a standalone controller, increasing portability massively.

The main unique selling point for this drone is the wheel attachment, allowing the drone to roll on the floor or ceiling, climb walls and also acts as a safety mechanism.

Hardware Overview

General Information

All the *"Parrot" "MiniDrones"* are based around the *"Parrot"* SIP6 SoC (*Parrot's* code refers to it as the P6I), which uses an 800MHz ARMv5TEJ ARM926EJ-S CPU (Arm A9 processor).

The motherboard used is a PF070070, attached to this we can find our sensors, wireless controllers and even a vertically attached camera running at 60 fps, 300,000 px resolution.

The drone is powered by a 550 mAh removable Lithium-Polymer battery, which allows for roughly 8 minutes of flight time.

The drone frame measures approx 96 mm in length, 95mm in width and 34 mm in height, and weights approx 60 grams, and according to manufacture specifications can fly at a speed of approx 18km/h.

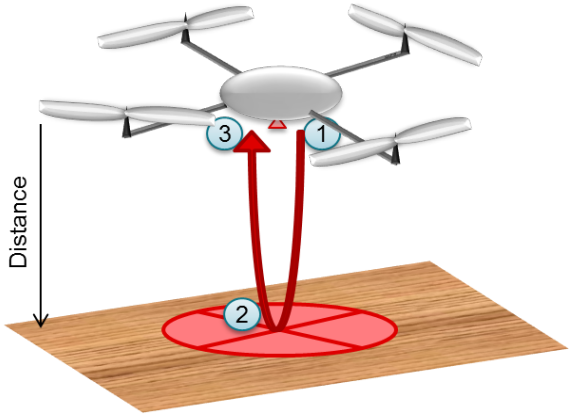
Sensors

There are several sensors and a 3-axis gyroscope, a 3-axis accelerometer that aid the drone in achieving stable flight and landing.

Using the ultrasonic sensor we can measure the drone's height by sending and receiving ultrasonic waves.

A pressure sensor (barometer) helps to maintain the drone's height, for stable hovering.

There is also a sensor for the battery, to initiate a landing when the battery is insufficient to continue flight.

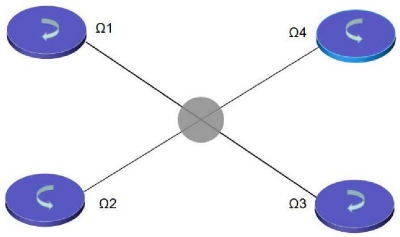


The above diagram illustrates the ultrasonic height measuring.

Propulsion System

The *"Rolling Spider"*, as most quadcopters, relies on four BLDC (Brushless DC) motors, each motor has a removable impact resistant plastic propellor attached (Propellers' Diameter: 55 mm Motor Spacing: 85 mm).

As with all components of the drone, the motors and propellers can be replaced.



These motors, as outlined in the above diagram, have been measured to run at almost 25,000 rpm.

A sophisticated feature of these motors is their ability to shut off preventing any further damage to the motors, or the propellers after a collision.

Wireless Communication

The *"Rolling Spider"* is controlled wirelessly using Bluetooth 4.0 LE (lower energy), and this requires a smartphone with this capability, and the app as previously mentioned.

Once the battery is inserted, the drone opens it's bluetooth band to accept an inbound connection, this connection arrives from the user's phone, it can be sent via the pairing menu on the phone, or from within the app itself.

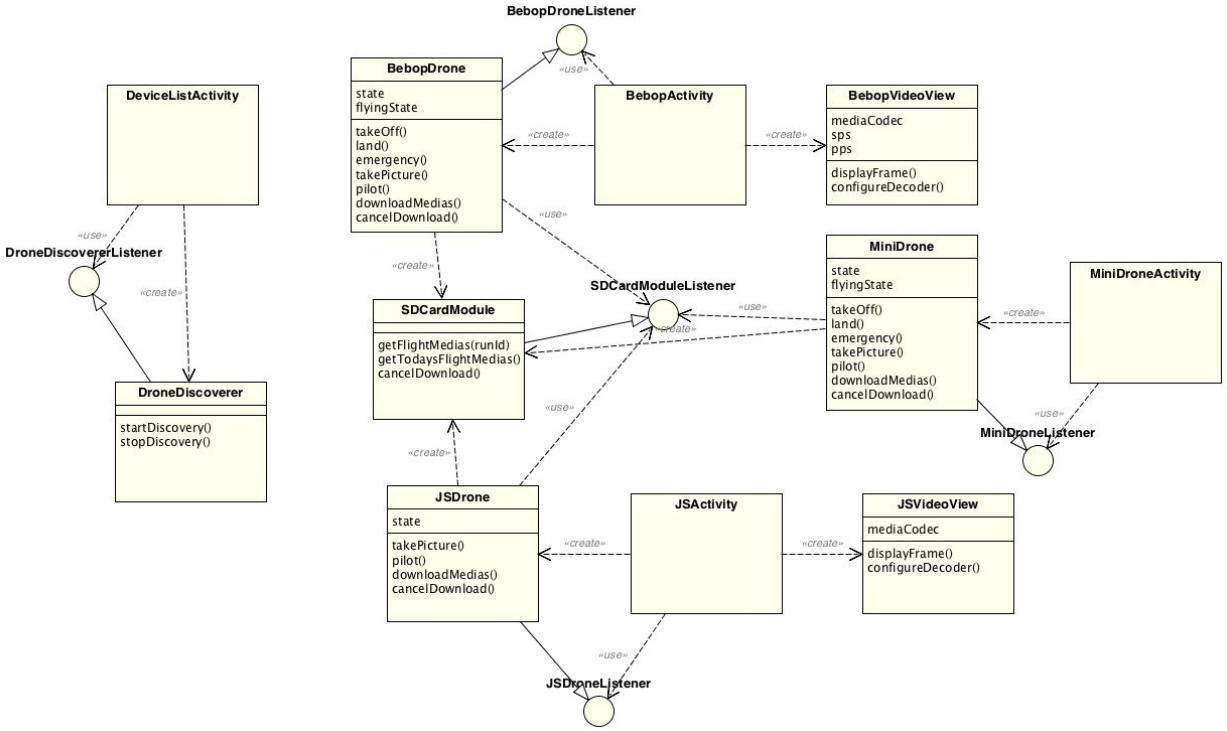
The drone also has two LED lights on the front in order to display various status to the user, both will be green once the controlling device has been connected.

Firmware

MiniDrone Framework

As *"Parrot"* creates a wide range of drones, they have multiple frameworks with varying degrees of similarity , all of the *"MiniDrones"* follow a very similar framework, these drones include the *"Bebop", "Mamba", "Rolling Spider*" and more.

Below is a UML diagram of the architecture for the Bebop drone, which is a more advanced minidrone, but has a very similar architecture to the *"Rolling Spider*".



From the above diagram, we can see various classes and their functions, we can see a drone discovery system to the left, this demonstrates the functionality, when the battery has been inserted into the drone, and it is not searching for inbound connections.

Then we can see the *BebopDrone* class itself, in the *"Rolling Spider*" firmware we can expect this to be a different name, but the same functions.

The main difference between this architecture and that of the *"Rolling Spider*", is the *BebopVideoView, SDCardModule*, and *JSVideoView* classes, as these outline features are not present in the *"Rolling Spider*".

Source Code Sample

Available on Github are various samples of source code that reflects the previously described framework, and others for the different drone variants, also made available by *"Parrot"* are SDK's for the various drones.

The code sample for the drones can be found here Github Link

In MiniDrone.h we can say a brief outline of the functions and their usages, such as checking connection, battery states, piloting states, video decoding and frame management for imaging, media download controls, and then finally various function initializations for the drone such as *(void)emergency, (void)takeOff, (void)land*, ect...

Here we take a quick look at the sample MiniDrone.m

We can see a multitude of functions and code allow us to get an in-depth comprehensive understanding of how flight is achieved.

In line 44 we can see a *connect* function, this function checks to see if the drone is already connected, if not we create a new device controller object, we also check to ensure that the drone is supported presumably supported by the application as different drone types require different applications, such as *"Freeflight Mini"* for *"MiniDrones"*, and "*Freeflight Pro*" for others.

Further down, we find commands such as *emergency, takeOff, land*, these functions may sound familiar as they were previously initialized in MiniDrone.h.

In line 149 is the *emergency* command, which checks if we have a device controller, and an ongoing connection, seems to then send a state or a call to a function that sends a piloting emergency to the device controller and the drone as reference, the expected outcome of this, is that the drone motors will then no longer receive power, and the drone will stop flying, in order to prevent further damage.

Similarly, we can see in the *takeOff* command a similar call to a piloting state or function, which will control the power to the motors.

Due to *"Parrot"* having a large range of drones, their code is structured in a quite scalable-friendly manner, thus requiring high level of C programming knowledge and patience to follow through all of the classes and references in the sample code, and possibly even the disassembling of the original firmware to fully understand the framework.

However from what we can discern from these samples, is that we have a drone class containing the core drone specific functions such as take off, land etc, a device controller class for connection functions, even subclasses within these for encapsualtion.

Custom Firmware

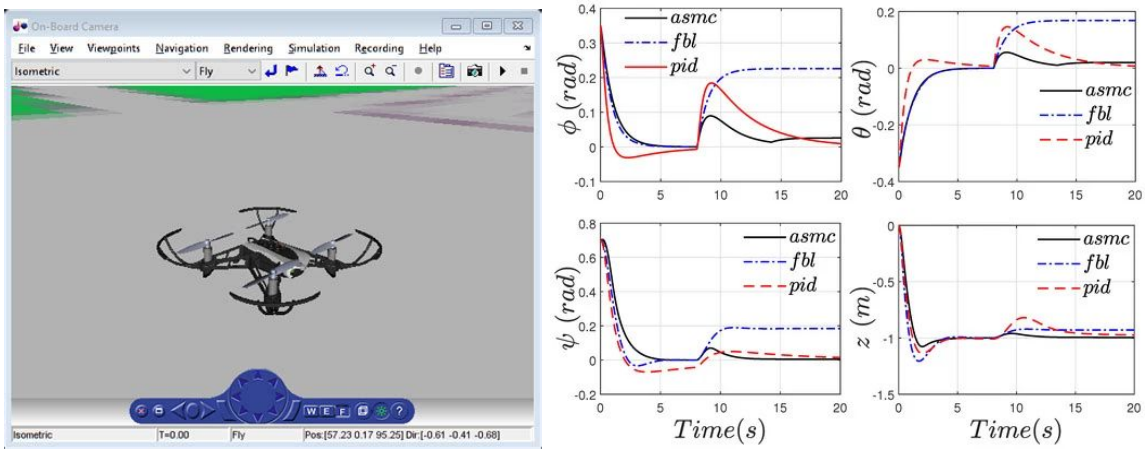
As seen in the *"Rolling Spider"* software package for Education provided by *"Parrot"* on Github, we are able to create and upload custom firmware to the drone, with relative ease, and without requiring any disassembly of the source code.

They also include a quick-start guide to creating and installing custom firmware, this can be done using a VM and the provided files in the Parrot_customFirmware directory of the Github repository.

Simulations

Alongside the custom firmware package, there is also a MatLab toolbox, which allows us to create various simulations, and deeper analysis of the drone and it's performance.

Further exploration of *MatLab* and *Parrots Simulink* simulations can be found here.



Above you can see simulink, and some kinematic result graphs

Conclusion

"Parrot" provides a wide range of support to developers, allowing them to explore the embedded frameworks of their drones, for experimentation, evaluation or even improvements.

The *"Rolling Spider"* drone whilst being a premium quality drone, with a multitude of complex systems that are also found in top-tier drones, is affordable and easily customized, providing the best experience for an eager developer to explore the quadcopter area.