TelloXplus

Robert W Senser, PhD

June 10, 2022

Abstract

The TelloXplus project showed that the Tello drone can be used in an entirely autonomous mode, with no ground control present. In the process we showed that the Tello SDK API is "capable" and that it is possible to add a microprocessor and GPS to the drone, enabling autonomous operations. Unfortunately, this drone has limited flight duration and is sensitive to wind. Several open issues remain in terms of reliable flight and use under poor lighting conditions.

1 Introduction

Our goal was to use the Tello drone in an autonomous mode by attaching a microprocessor and GPS. The GPS provided navigation information to the microprocessor and the microprocessor communicated with the drone. No ground or operator communications were involved.

2 TelloXplus Physical Layout

Figure 1 shows the physical layout of this drone. The drone's upper plastic cover was removed and replaced by the microprocessor board; the GPS was attached to the top of this board. The small battery, to power the microprocessor and GPS, was attached below the Tello. The bottom view, Figure 4, shows the battery placement. The white string attached was used to keep the Tello under positive control.[1],[2] The drone flies autonomously with the microprocessor's WiFi connected to the Tello.[3]

[1] This string also guarantees that this autonomous flier does not run afoul of the US FAA regulations.

[2] Unfortunately this string did create issues when it sometimes slipped under the Tello as it then seemed to confused the Tello's down-facing optical flow sensor.

[3] There is no real-time, human operator control.

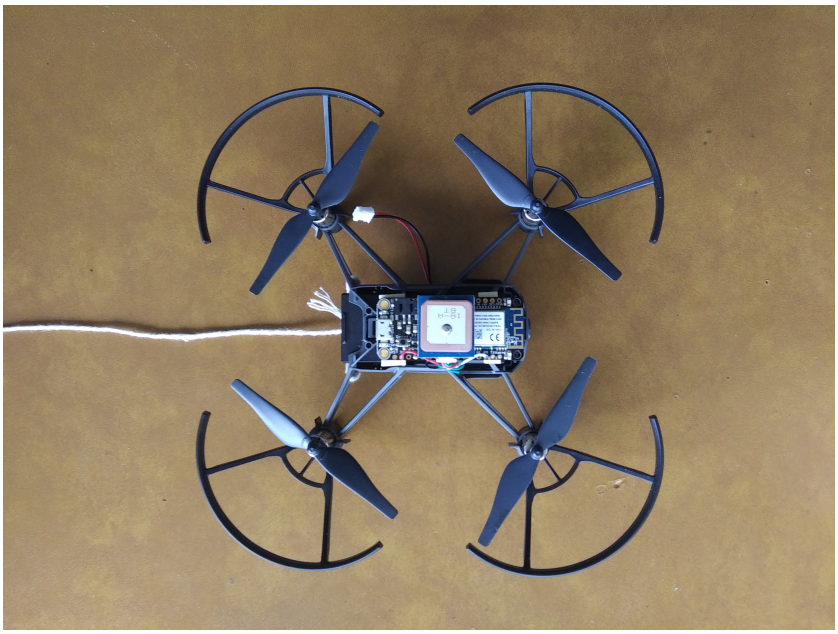

Figure 1: TelloXplus Layout

3 Example Flight

This short YouTube video shows the TelloX flying. Shown is an earlier version of TelloX; the GPS had not yet added. The drone made a number of 5-meter steps, all in the same direction.[4]

`https://www.youtube.com/shorts/vkR1ZG9w-lw`

4 The "Stock" Tello Drone

The Tello consumer-grade drone includes a programming interface that can be used from a number of programming languages, including C/C++. For specific details about the basic Tello and it's capabilities, please see these web sites:

1. `https://en.wikipedia.org/wiki/Tello_UAV`
2. `https://www.ryzerobotics.com/news/announcing-the-programmable-tello-edu-drone-page`
3. `https://dl-cdn.ryzerobotics.com/downloads/tello/20180910/Tello%20SDK%20Documentation%20EN_1.3.pdf`

The TelloXplus software adheres to the Tello SDK 1.3 specification.

[4] Door's background music just seemed appropriate!

5 Microprocessor (MPU): Adafruit Feather

An Adafruit Feather board, with WiFi, was used to control the Tello, input the GPS information and manage the Tello's flight. The TelloXplus C/C++ software accessed the Tello drone via WiFi, utilizing UDP. The Adafruit MPU's part number is 3010. This board's battery connection was very helpful; it permitted easy connection to the additional battery. URL for this board:

`https://www.adafruit.com/product/3010`

It is likely that other small, lightweight microprocessors with WiFi support could have be used.

6 GPS: BN-220 GPS Module

The GPS used was a common "hobby grade" GPS, available from Amazon.

`https://www.amazon.com/gp/product/B086ZK9BT4`

Other similar GPS units were tried and this unit showed good performance and had a small "foot print". Note that the GPS unit was close to the propellers and that only the transmit GPS-to-microprocessor-board serial wire was connected. The GPS ran in its default mode. Figure 3 shows how close the propellers were to the GPS.[5]

7 Additional Battery

The additional 100mAh battery, which powers the MPU and GPS, was placed under the Tello. We tried batteries between 100mAh and 250mAh. We experienced no issues with the smaller, lighter 100mAh battery.

Important note about batteries and the Adafruit MPU connector. Due to the high cost of Adafruit-supplied batteries, we used this common battery from Amazon **AFTER SWITCHING** the connector leads.

`https://www.amazon.com/gp/product/B07BTSTHFD`

8 TelloXplus Software

This software had to manage a number of concurrent resources (GPS, serial input, blinking-LED, WiFi connection, etc.). This required that the TelloXplus software support software threading. This "soft" threading was controlled by the C/C++ program as no operating system was present. [6]

[5] There was a gap under the GPS and the propellers could spin through this gap, if needed.

[6] The TelloXplus software was totally rewritten, somewhat based on the previous TelloX software. The previous software was hard to maintain and enhance.

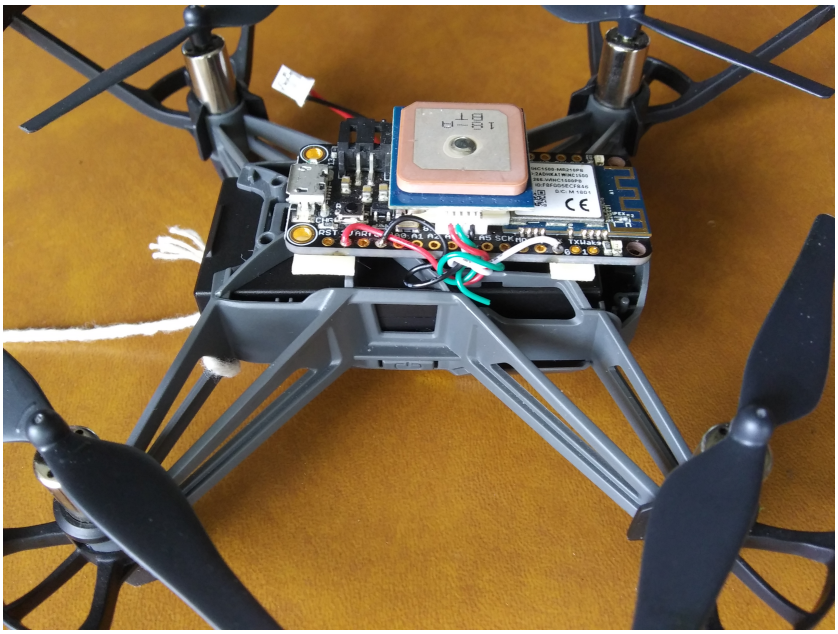

Figure 2: Wiring and GPS Placement

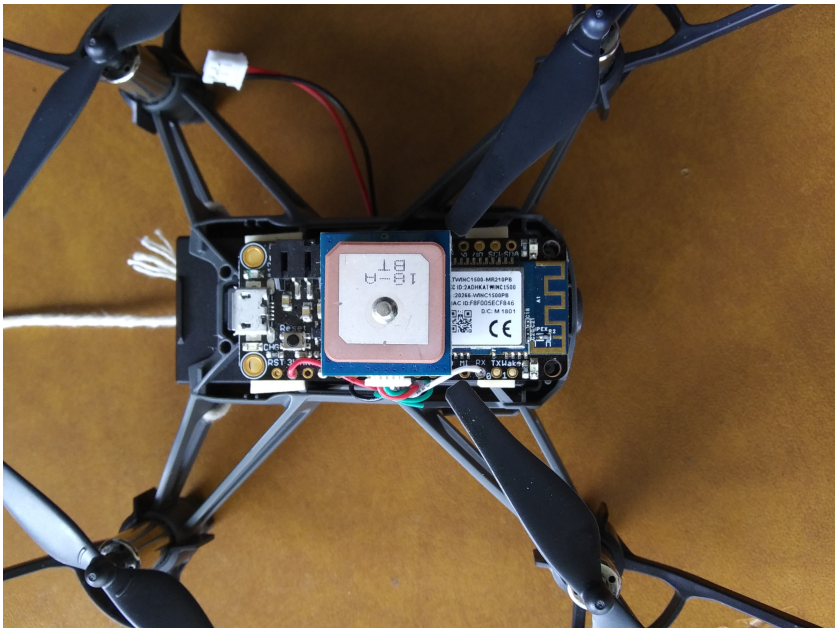

Figure 3: GPS Clearance

Figure 4: Placement of Additional Battery

8.1 Software Overview

8.2 uNav

The *universal navator* (uNav) is a set of Arduino-style modules that support soft threading and support a number of sensor devices.[7] The TelloXplus application used uNav's threading support and also used the timer, blink-LED and GPS support. The detailed specifics of uNav are outside the context of this document. Suffice it to say that uNav took over the **setup()** and **loop()** methods of the Arduino environment and supplied the replacement **setupX()** and **loopX()** methods for application use. For more details, see the uNav document.[8]

8.3 In GIT

Source modules used in TelloXplus software:

- **rwsGPS4.h**: Specialized GPS NMEA string parser with threading support.
- **rwsTiny.h**: Wrapper for Tiny GPS course and distance methods.
- **TinyGPS.cpp**: Standard TinyGPS code.

[7] These sensors included the BME280 barometer and the MPU6050 gyroscrope boards.
[8] Which is not yet available.

- **TinyGPS.h**: Standard TinyGPS header.
- **uNavBase.h**: Base uNav code, provided "soft" threading and sensor interfaces.
- **uNavDecl.h**: Function definitions for uNav.
- **uNavMacros.h**: Configuration macros for uNav made it possible to omitted unused sensors.
- **uNavTelloX5-GIT.ino**: Application code, with WiFi support.

TelloXplus public GIT repository URL:

`https://github.com/rwsenser/TelloXplus`

9 TelloXplus Specifics

Actual measurements showed that a standard Tello drone weights 84.2 grams w/battery. The published weight for a Tello is 80 grams.[9] The TelloXplus weights 97.1 grams with added MPU board, GPS and both batteries.

The published maximum flight duration for a this drone is up to 13 minutes.[10] We normally limited our flights to 200 seconds, using the included TelloXplus watch-dog timer.[11] Our tests, performed with "aged" Tello batteries, showed maximum flight times up to 4:30. The TelloXplus added weight caused the motors to "spin aggressively." In flight, the TelloXplus seemed to generate noticeably higher-pitched sounds, relative to a standard Tello.

10 TelloXplus Discussion

10.1 Overall Success

Our TelloXplus functioned as envisioned, although with some unexpected limits. These are noted below. The stock Tello drone is stable carrying the added weight. No stability issues were encountered. The UDP communications between the MPU and Tello drone were reliable. The drone climbed, moved around and landed as requested. The added GPS made it possible for the drone to have real directional data available. The threading used in the software enabled the the status LED output, navigational calculations, GPS input and communication with the drone to all be handled concurrently, all via the small MPU.

[9] See: https://www.ryzerobotics.com/tello/specs

[10] See: https://www.dronethusiast.com/dji-tello-quadcopter-drone/

[11] This feature minimized potential "fly aways" and the timer value can be changed in the application source code.

10.2 Discovered Limits and Associated Issues

Discovered Limits

1. In general, TelloXplus flight times were limited to flights of 60 to 120 seconds. This was enough time for the flight paths to completed. With longer flights, we did encounter occasional issues with unexpected landings. Poor lighting conditions seemed to contribute to the test Tello to "self landing".[12] We experienced these erroneous self landings under all sorts of lighting conditions.

2. As with a stock Tello, TelloXplus is wind sensitive.

Associated Issues

1. We had the sense that updates to the Tello SDK API (version 1.1 to 1.3) actually made the Tello drone less predictable and more prone to unexpected landings.[13]

2. We also had the sense that the sensor performance of our test Tello decreased of over the 2+ years of development effort. In particular, we suspect the "optical flow" performance was reduced.

11 Next Steps

11.1 Understand the Unexpected Landings

Before putting substantially more effort into our TelloXplus project, we would like to understand what caused the unexpected Tello landings. This issue could be due to our use of a 2-year old Tello. On the other hand, these unexpected landing could be due to something not yet identified.

11.2 WTOK Operation

The Tello SDK replies with a UDP-transmitted "OK" literal after each successful movement command. TelloXplus uses an set, fixed wait interval after each operation and does not check for this "OK" literal. The previous version of the TelloX software checked for this "OK" literal and upon detection, it then stopped waiting. The current software just waits for the full period. The old WTOK code is commented out and needs to be reintegrated and debugged so that the drone moves more quickly from command to command.

[12] The exact meaning of "poor lighting considerations" remained elusive.

[13] We did not have the ability to do behavior comparisons better SDK version 1.1 and 1.3.

11.3 Variable Flight Path Support

The flight plan embedded in the TelloPlusX software is largely hard coded. The previous software version used a "general" data structure that could be modified at program start. This capability needs to be introduced into TelloXplus.

12 Closing Thoughts

It works, it works!!