gobot.io

Hello, Tello - Hacking Drones With Go

Ron Evans

9-11 minutes

The DJI Tello is a new quadcopter which combines powerful technology from DJI and Intel into a very tiny package. We recently got our hands on a couple of these new drones, and we now have support for Gobot, so you can control the drone using the Go programming language.

For only around $99 US, it is amazing how many features are included. Like many more expensive models, the Tello connects to the ground controller via WiFi, and includes an onboard 5 megapixel HD camera along with various other sensors.

However, what really sets the Tello apart is the 14-core processor from Intel that includes an onboard Movidius Myriad 2 VPU (Video Processing Unit) for advanced imaging and vision processing.

Despite all this cool hardware, the official Tello SDK only offers a very limited set of the drone's overall capabilities. Luckily,

there is a much better API hiding under the surface for those who know how to look. This undocumented API is the one used by the official Tello Android and iOS applications.

An active community has quickly formed around hacking the ground station and video protocols used by the Tello. Thanks to tools like Wireshark, and the generous nature of open source, we've collectively already deciphered much of the protocol used by the Tello.

We have used this knowledge to create a new driver for Gobot that lets you explore the drone's functionality, and combine it with the other capabilities of the Gobot framework, such as controlling the Tello drone with a Playstation Dualshock3 joystick.

Obtaining a drone

The initial shipments of this interesting new drone have already sold out very quickly in many places. You can purchase online via Amazon.com, or you can also order directly from the DJI store.

Ready for takeoff

In the Gobot implementatation, we have tried to expose the Tello API in a way that is mostly compatible with other drones that are already supported in Gobot.

Here is an example that just takes off, and 5 seconds later,

lands:

```
package main

import (
    "time"

    "gobot.io/x/gobot"
    "gobot.io/x/gobot/platforms/dji/tello"
)

func main() {
    drone := tello.NewDriver("8888")

    work := func() {
        drone.TakeOff()

        gobot.After(5*time.Second, func() {
            drone.Land()
        })
    }

    robot := gobot.NewRobot("tello",
        []gobot.Connection{},
        []gobot.Device{drone},
        work,
    )
```

```
    robot.Start()
}
```

Controlling your flight

The Tello is pretty manueverable, and there are a number of different axes on which you can control the drone. Like most fly by wire devices, the control system closely parallels those used by radio control helicopters and airplanes.

However, one big advantage of most multicopters is their stability while in-flight. This allows you to simply send flight motion commands to the drone, like "go right", and then have the drone respond relative to its original position. This is a lot easier to control then the traditional control stick of an R/C helicopter, especially if your intention is to avoid crashes while coding your drone.

For example, instead of worrying about "yaw" or "pitch", you can just tell the drone to fly "forward" or "right" like this:

```
package main

import (
    "time"

    "gobot.io/x/gobot"
    "gobot.io/x/gobot/platforms/dji/tello"
)
```

```
func main() {
    drone := tello.NewDriver("8888")

    work := func() {
        drone.TakeOff()

        gobot.After(3*time.Second, func() {
            drone.Forward(10)
        })

        gobot.After(6*time.Second, func() {
            drone.Backward(10)
        })

        gobot.After(9*time.Second, func() {
            drone.Land()
        })
    }

    robot := gobot.NewRobot("tello",
        []gobot.Connection{},
        []gobot.Device{drone},
        work,
    )

    robot.Start()
}
```

Want to do tricks? No problem, the Tello can perform flips.

Here is a short example:

```
package main

import (
    "time"

    "gobot.io/x/gobot"
    "gobot.io/x/gobot/platforms/dji/tello"
)

func main() {
    drone := tello.NewDriver("8888")

    work := func() {
        drone.TakeOff()

        gobot.After(5*time.Second, func() {
            drone.FrontFlip()
        })

        gobot.After(10*time.Second, func() {
            drone.BackFlip()
        })

        gobot.After(15*time.Second, func() {
            drone.Land()
```

```
        })
    }

    robot := gobot.NewRobot("tello",
        []gobot.Connection{},
        []gobot.Device{drone},
        work,
    )

    robot.Start()
}
```

Getting the messages

The Tello sends a number of different telemetry messages back down to the ground station. For example, the current battery level and the drone's height. Take a look at the `tello.FlightDataEvent` for more information.

Streaming video

You can also view streaming video being sent from the drone to the ground station. Here is a sample that obtains the video stream from the Tello, and then displays it using the excellent open source MPlayer (https://mplayerhq.hu)

```
package main

import (
```

```
        "fmt"
        "os/exec"
        "time"

        "gobot.io/x/gobot"
        "gobot.io/x/gobot/platforms/dji/tello"
    )

    func main() {
        drone := tello.NewDriver("8890")

        work := func() {
            mplayer := exec.Command("mplayer",
"-fps", "25", "-")
            mplayerIn, _ := mplayer.StdinPipe()
            if err := mplayer.Start(); err !=
nil {
                fmt.Println(err)
                return
            }

            drone.On(tello.ConnectedEvent,
func(data interface{}) {
                fmt.Println("Connected")
                drone.StartVideo()
                drone.SetVideoEncoderRate(4)
```

```
gobot.Every(100*time.Millisecond, func() {
                drone.StartVideo()
            })
        })

        drone.On(tello.VideoFrameEvent,
func(data interface{}) {
            pkt := data.([]byte)
            if _, err :=
mplayerIn.Write(pkt); err != nil {
                fmt.Println(err)
            }
        })
    }

    robot := gobot.NewRobot("tello",
        []gobot.Connection{},
        []gobot.Device{drone},
        work,
    )

    robot.Start()
}
```

You can also use computer vision software such as GoCV (https://gocv.io/), our Go language wrapper around OpenCV to perform face tracking and other interesting applications.

How it works

The ground station connects to the Tello drone using WiFi. The protocol to send commands to the drone uses UDP packets in a specific binary format. The responses from the drone back to the ground station likewise use UDP packets.

Control messages

The packet format for the control messages, and the responses from the drone, both use the same basic format.

Position	Usage
0	`0xcc` indicates the start of a packet
1-2	Packet size, 13 bit encoded `([2] << 8) | ([1] >> 3)`
3	CRC8 of bytes 0-2
4	Packet type ID
5-6	Command ID, little endian
7-8	Sequence number of the packet, little endian
9-(n-2)	Data (if any)
(n-1)-n	CRC16 of bytes 0 to n-2

So for example, the takeoff command packet looks like this:

```
0xcc 0x58 0x00 0x7c 0x68 0x54 0x00 0xe4 0x01 0xc2 0x16
```

Value	Usage

Value		Usage
`0xcc`		start of packet
`0x58`	`0x00`	Packet size of 11
`0x7c`		CRC8 of bytes 0-2
`0x68`		Packet type ID
`0x54`	`0x00`	"Takeoff" command ID, little endian
`0xe4`	`0x01`	Sequence number of the packet, little endian
`0xc2`	`0x16`	CRC16 of bytes 0 to 8

Video packets

The streaming video is also sent to the ground station on a different UDP port. Unfortunately the Tello does not just use one of the existing standards for streaming video such as RTSP. Instead, the raw video packets are sent via UDP and need to be reassembled and decoded before they can be viewed.

Since the size of a single video frame is larger than the size of a single UDP packet, the Tello breaks apart each frame, and sends the packets with header of 2 bytes to indicate how they need to be reassembled.

Position	Usage
0	Sequence number
1	Sub-sequence number
2-n	Video data

The video data itself is just H264 encoded YUV420p. Using this information, it is possible to decode the video using standard tools such as ffmpeg, once you remove the header bytes.

As mentioned in the beginning of this post, the Gobot driver for the DJI Tello could not exist without the many contributions and investigation of the awesome community on the unofficial Tello forum.

Special thanks to @Kragrathea who figured out a LOT of the packets and code as implemented in C#: https://github.com/Kragrathea/TelloPC

Also thanks to @microlinux with the Python library which served as the first example for the Tello community: https://github.com/microlinux/tello

And bluejune's incredible work on the https://bitbucket.org/PingguSoft/pytello repo, especially the Wireshark Lua dissector for Tello which has proven indispensable.

Last but not least, thank you @maruel for helping figure out various Go-related video decoding questions.

Keep In Touch

Want to know about the latest Gobot news? Follow us on Twitter at @gobotio. Thank you!