Controlling a drone using Rust

Who am I?

Lachezar Lechev

- ⛓️ 🔗 ⛓️ 🔗 at 🌞 @AdExNetwork
- 🚀 🛰️ 🛰️ 📡 at 🌑 @AeroRust
- 🛬 🌍 🛫 Travel
- 🙌 🤩 🙌 Trance 🎶

😅

Make it fly!

Parrot Anafi 4k

Parrot Anafi 4k

1. WiFi
2. 4k Camera
3. C / Android / iOS SDKs
4. Other SDKs (PyParrot)
5. Sphinx simulation tool
6. FreeFlight 6
7. SkyController
8. ...

How does it work?

Communication

Connecting over WiFi to the drone

1. Handshake
2. Protocol
3. Frames

`arsdk-rs` crate

Handshake

Handshake

- TCP to the drone's IP address (port: 44444)
- Send JSON serialized Request (`serde`)
  - `d2c_port` 🛩️ ➡️ 🎮
  - ...
- A Response deserialized from JSON (`serde`)
  - `c2d_port` 🎮 ➡️ 🛩️
  - `status: 0` OK 👍
  - ...

Handshake

We have a connection! 🙌

```
[.. INFO  arsdk_rs] Init address 10.202.0.1:44444
[.. INFO  arsdk_rs::handshake] Connecting controller arsdk-rs
[.. INFO  arsdk_rs::handshake] Request:
{"controller_name":"arsdk-rs","controller_type":"computer","d2c_port":432
10,"arstream2_client_stream_port":44445,"arstream2_client_control_port":4
4446}
```

Handshake

We have a connection! 🙌

```
[.. INFO  arsdk_rs::handshake] Read 106 bytes!
[.. INFO  arsdk_rs::handshake] Response: { "c2d_update_port": 51,
"c2d_user_port": 21, "status": 0, "c2d_port": 2233, "qos_mode": 0,
"proto_v": 1 }
```

Protocol

Protocol

There is a protocol however! You should use Frames!

- Ping-Pong
- Listener
- Cmd Sender

Ping-Pong

Keeping the connection 🛩️🔗🎮

- Drone sends PING
- Expects PONG

> 7 secs ➡️ disconnect ❌

Listener

Receiving Frames from the Drone 🛩️ ➡️ 🎮

- UDP (Response::c2d_port)
- Contains >= 1 Frames
- Deserializes (Un)Known Frames
- PONG & Acknowledge Responses

Listener Logs

```
[.. DEBUG arsdk_rs::listener] Received: 8 bytes from 10.202.0.1:2233
[.. DEBUG arsdk_rs::listener] Bytes: 1 139 0 8 0 0 0 0
[.. INFO  arsdk_rs::parse] Frame: Frame { frame_type: Ack,
buffer_id: ACKFromSendWithAck, sequence_id: 0,
feature: Some(Common(None)) }
```

CMD Sender

Sending Frames to the Drone 🎮 ➡️ 🛩️

- UDP (Request::d2c_port)
- Sends raw bytes to the drone

CMD Sender Logs

```
[.. INFO  arsdk_rs] Frame to sent: [4, 11, 0, 11, 0, 0, 0, 1, 0, 1, 0]
[.. INFO  arsdk_rs] Sent Frame (length: 11) => Ok(Frame {
frame_type: DataWithAck, buffer_id: CDAck, sequence_id: 0,
feature: Some(ArDrone3(Some(Piloting(TakeOff)))) })
```

The logs from earlier

```
[.. DEBUG arsdk_rs::listener] Received: 8 bytes from 10.202.0.1:2233
[.. DEBUG arsdk_rs::listener] Bytes: 1 139 0 8 0 0 0 0
[.. INFO  arsdk_rs::parse] Frame: Frame { frame_type: Ack,
buffer_id: ACKFromSendWithAck, sequence_id: 0,
feature: Some(Common(None)) }
```

Frames

Message like this:

```
let buf: [u8; 35] = [
    // first:
    2, 0, 1, 23, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 233, 72, 37, 42, 0, 0, 0, 0,
    // second:
    4, 126, 1, 12, 0, 0, 0, 0, 14, 1, 0, 0,
];
```

Translates to this:

First Frame:

[2] Type - Data
[0] BufferID - Ping
[1] Sequence id 1
[23,0,0,0] Length 23
[0,0,0,0,0,0,0,233,72,37,42,0,0
,0,0] Ping's Gibberish Data

Second Frame:

[4] Type - DataWithAck
[126] BufferID - DCEvent
[1] Sequence id 1
[12,0,0,0] Length 12
[0] Feature - Common
[14] Class - CalibrationState
[1,0,0] x y z axis

With types:

```
FrameType::Known(Frame {
    frame_type: Type::Data,
    buffer_id: BufferID::PING,
    sequence_id: 1,
    feature: Some(Feature::Unknown {
        feature: 3,
        data: vec![0, 0, 0, 0, 0, 0,
0, 233, 72, 37, 42, 0, 0, 0, 0],
    }),
}),
```

```
FrameType::Known(Frame {
    frame_type: Type::DataWithAck,
    buffer_id: BufferID::DCEvent,
    sequence_id: 1,
    feature:
Some(Feature::Common(Some(crate::comm
on::Class::Unknown {
        // CalibrationState
        class: 14,
        // x y z axis
        data: vec![1, 0, 0],
    })))),
})
```

Type & BufferID

```
pub enum Type {
    Uninitialized = 0,
    Ack = 1,
    Data = 2,
    LowLatency = 3,
    DataWithAck = 4,
    Max = 5,
}
```

```
pub enum BufferID {
    PING = 0,
    PONG = 1,
    CDNonAck = 10,
    CDAck = 11,
    CDEmergency = 12,
    CDVideoAck = 13,
    DCVideo = 125,
    DCEvent = 126,
    DCNavdata = 127,
    ACKFromSendWithAck = 139,
}
```

The Frame

```
pub struct Frame {
    pub frame_type: Type,
    pub buffer_id: BufferID,
    pub sequence_id: u8,
    pub feature: Option<Feature>,
}
```

```
pub enum Feature {
    Common(Option<common::Class>),
    ArDrone3(Option<ArDrone3>),
    Minidrone,
    JumpingSumo(jumping_sumo::Class),
    SkyController,
    PowerUp,
    Generic,
    FollowMe,
    Wifi,
    RC,
...
```

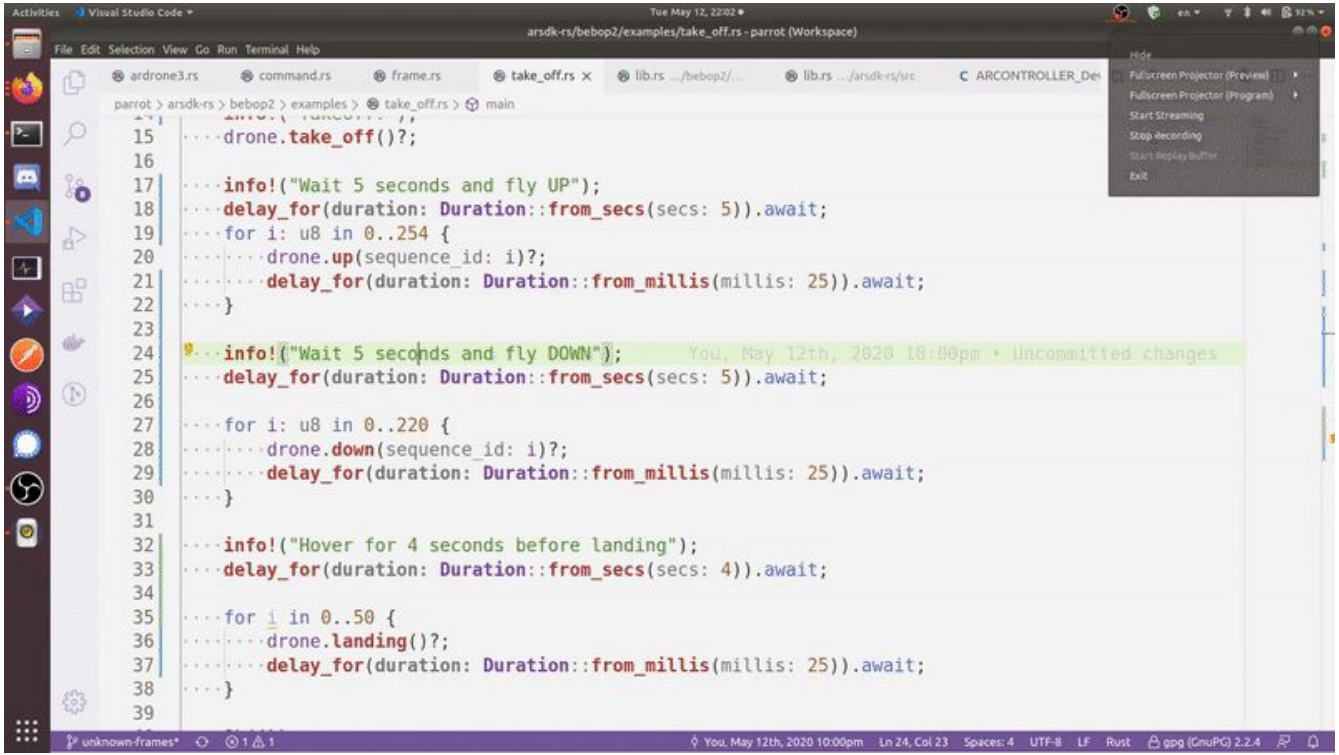
(De)Serialization

Crate `scroll`

- impl TryIntoCtx
- impl TryFromCtx

Flying... 🎮 🛬 🛩️ 🛬 ???

Flying... 🎮 🛫 🛩️ 🛬 ???



Speed: x2

🚧 Work in progress 🚧

Many unexplored depths of cool things to make!

- Rust Computer Vision
- Flying?!
- Improvements
- !! Implementations !!

Inspired to build?

Thank you ❤️ !

ÄroRust:

https://github.com/AeroRust/

- arsdk-rs: https://github.com/AeroRust/arsdk-rs/
- Parrot Developer: https://developer.parrot.com/
- PyParrot: https://github.com/amymcgovern/pyparrot
- Rust Computer Vision: https://github.com/rust-cv/