Drones Still Going Open Source

Julien BERAUD

October 12th 2016 - Embedded Linux Conference Europe Berlin

Introduction

ArduPilot on Parrot Bebop 2

Parrot Disco and C.H.U.C.K

Software Architecture for Video

Video implementation with ArduPilot

Building code for Parrot Disco

What’s left to do ?

Parrot SLAM.dunk

Conclusion

Introduction

Introduction

Parrot Bebop 2

- 500g
- Parrot P7 SoC (dual Cortex A9)
- IMU, Barometer, Compass, Vertical Camera, Sonar, GPS
- Linux kernel 3.4 (no mainline support)
- Front camera with fish-eye lens

Parrot Bebop 2

- 500g
- Parrot P7 SoC (dual Cortex A9)
- IMU, Barometer, Compass, Vertical Camera, Sonar, GPS
- Linux kernel 3.4 (no mainline support)
- Front camera with fish-eye lens

Parrot Bebop 2

- 500g
- Parrot P7 SoC (dual Cortex A9)
- IMU, Barometer, Compass, Vertical Camera, Sonar, GPS
- Linux kernel 3.4 (no mainline support)
- Front camera with fish-eye lens

Parrot Bebop 2

- 500g
- Parrot P7 SoC (dual Cortex A9)
- IMU, Barometer, Compass, Vertical Camera, Sonar, GPS
- Linux kernel 3.4 (no mainline support)
- Front camera with fish-eye lens

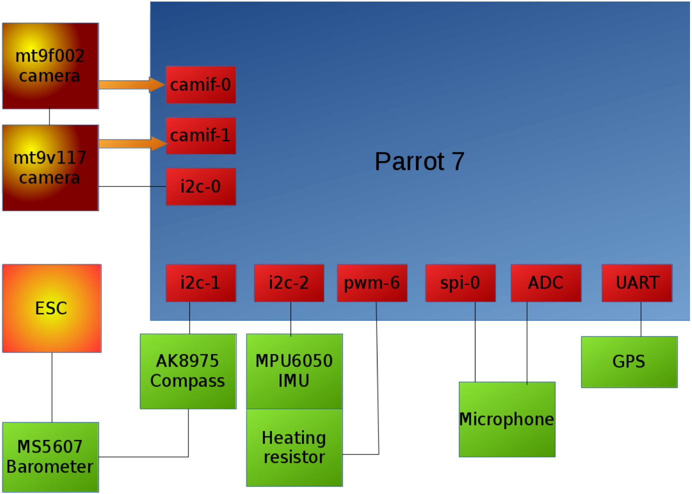
Parrot Bebop 2

- 500g
- Parrot P7 SoC (dual Cortex A9)
- IMU, Barometer, Compass, Vertical Camera, Sonar, GPS
- Linux kernel 3.4 (no mainline support)
- Front camera with fish-eye lens

ArduPilot on Parrot Bebop 2

ArduPilot on Parrot Bebop 2

Hardware architecture



Ardupilot

Ardupilot

Ardupilot (APM)

- Open Source - GPLv3
- Originally developed to run on an Arduino
- C++
- Some linux boards already supported before Bebop

Ardupilot (APM)

- Open Source - GPLv3
- Originally developed to run on an Arduino
- C++
- Some linux boards already supported before Bebop

Ardupilot (APM)

APM
MULTIPLATFORM AUTOPILOT

- Open Source - GPLv3
- Originally developed to run on an Arduino
- C++
- Some linux boards already supported before Bebop

Ardupilot (APM)

APM
MULTIPLATFORM AUTOPILOT

- Open Source - GPLv3
- Originally developed to run on an Arduino
- C++
- Some linux boards already supported before Bebop

Software architecture

- Vehicle specific flight code (ArduCopter, ArduPlane, ArduRover)
- Shared libraries that include sensor drivers
- Hardware Abstraction Layer providing access to platform-specific methods
- AP_HAL_Linux giving access to spidev, i2c-dev, uart drivers, etc...

Software architecture

- Vehicle specific flight code (ArduCopter, ArduPlane, ArduRover)
- Shared libraries that include sensor drivers
- Hardware Abstraction Layer providing access to platform-specific methods
- AP_HAL_Linux giving access to spidev, i2c-dev, uart drivers, etc...

Software architecture

- Vehicle specific flight code (ArduCopter, ArduPlane, ArduRover)
- Shared libraries that include sensor drivers
- Hardware Abstraction Layer providing access to platform-specific methods
- AP_HAL_Linux giving access to spidev, i2c-dev, uart drivers, etc...

Software architecture

- Vehicle specific flight code (ArduCopter, ArduPlane, ArduRover)
- Shared libraries that include sensor drivers
- Hardware Abstraction Layer providing access to platform-specific methods
- AP_HAL_Linux giving access to spidev, i2c-dev, uart drivers, etc...

Flying

`https://www.youtube.com/watch?v=HJN_gT1eNDk`
`https://www.youtube.com/watch?v=ZnEFcJx1qko`
Thanks to Randy Mackay (Copter Maintainer)

Parrot Disco and C.H.U.C.K

Parrot Disco and C.H.U.C.K

Parrot Disco

- Fixed Wing
- Architecture close to the Bebop's
- Built so its main board (C.H.U.C.K) can be used on another vehicle

Parrot Disco

- Fixed Wing
- Architecture close to the Bebop's
- Built so its main board (C.H.U.C.K) can be used on another vehicle

Parrot Disco

- Fixed Wing
- Architecture close to the Bebop's
- Built so its main board (C.H.U.C.K) can be used on another vehicle

C.H.U.C.K

- Targetting hobbyists, Research and Education
- ArduPlane seemed like a perfect candidate
- Porting made easier by previous Bebop porting
- Expert in that area in the person of Tridge

C.H.U.C.K

- Targetting hobbyists, Research and Education
- ArduPlane seemed like a perfect candidate
- Porting made easier by previous Bebop porting
- Expert in that area in the person of Tridge

C.H.U.C.K

- Targetting hobbyists, Research and Education
- ArduPlane seemed like a perfect candidate
- Porting made easier by previous Bebop porting
- Expert in that area in the person of Tridge

C.H.U.C.K

- Targetting hobbyists, Research and Education
- ArduPlane seemed like a perfect candidate
- Porting made easier by previous Bebop porting
- Expert in that area in the person of Tridge

Differences with Bebop 2

- MS415DO Airspeed sensor
- RC input over UART
- ESC differences
- Compass calibration issues
- Wiki for users `http://ardupilot.org/plane/docs/airframe-disco.html`

Differences with Bebop 2

- MS415DO Airspeed sensor
- RC input over UART
- ESC differences
- Compass calibration issues
- Wiki for users `http://ardupilot.org/plane/docs/airframe-disco.html`

Differences with Bebop 2

- MS415DO Airspeed sensor
- RC input over UART
- ESC differences
- Compass calibration issues
- Wiki for users `http://ardupilot.org/plane/docs/airframe-disco.html`

Differences with Bebop 2

- MS415DO Airspeed sensor
- RC input over UART
- ESC differences
- Compass calibration issues
- Wiki for users
  `http://ardupilot.org/plane/docs/airframe-disco.html`

Differences with Bebop 2

- MS415DO Airspeed sensor
- RC input over UART
- ESC differences
- Compass calibration issues
- Wiki for users
  `http://ardupilot.org/plane/docs/airframe-disco.html`

Missing features

What's missing ?

Video

Missing features

What’s missing ?

Video

Software Architecture for Video

Software Architecture for Video

DragonProg

- We came from a world of RTOS
- 1 big process
- Lots of threads and priorities
- Little reusability and high maintenance overhead
- Switch to a new architecture based on several processes

DragonProg

- We came from a world of RTOS
- 1 big process
- Lots of threads and priorities
- Little reusability and high maintenance overhead
- Switch to a new architecture based on several processes

DragonProg

- We came from a world of RTOS
- 1 big process
- Lots of threads and priorities
- Little reusability and high maintenance overhead
- Switch to a new architecture based on several processes

DragonProg

- We came from a world of RTOS
- 1 big process
- Lots of threads and priorities
- Little reusability and high maintenance overhead
- Switch to a new architecture based on several processes

DragonProg

- We came from a world of RTOS
- 1 big process
- Lots of threads and priorities
- Little reusability and high maintenance overhead
- Switch to a new architecture based on several processes

Dragonfly

- Processes are now split
- 3 main processes
- IPC to exchange data at high rate
- One process in charge of video : Parrot IMaging Process
- Use of libshdata/libtelemetry to export/import data

Dragonfly

- Processes are now split
- 3 main processes
- IPC to exchange data at high rate
- One process in charge of video : Parrot IMaging Process
- Use of libshdata/libtelemetry to export/import data

Dragonfly

- Processes are now split
- 3 main processes
- IPC to exchange data at high rate
- One process in charge of video : Parrot IMaging Process
- Use of libshdata/libtelemetry to export/import data

Dragonfly

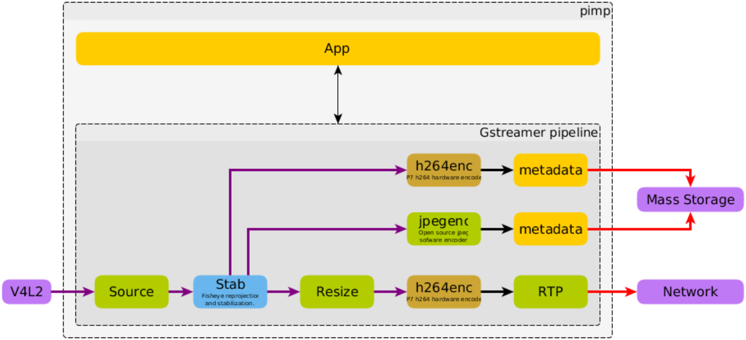
- Processes are now split
- 3 main processes
- IPC to exchange data at high rate
- One process in charge of video : Parrot IMaging Process
- Use of libshdata/libtelemetry to export/import data

Dragonfly

- Processes are now split
- 3 main processes
- IPC to exchange data at high rate
- One process in charge of video : Parrot IMaging Process
- Use of libshdata/libtelemetry to export/import data

Pimp

Imaging Process



libtelemetry

- Library to import/export data between processes
- Built on top of libshdata
- Non-blocking
- Uses shm
- Timestamped data
- `https://github.com/Parrot-Developers/libshdata`
- `https://github.com/Parrot-Developers/telemetry`

libtelemetry

- Library to import/export data between processes
- Built on top of libshdata
- Non-blocking
- Uses shm
- Timestamped data
- `https://github.com/Parrot-Developers/libshdata`
- `https://github.com/Parrot-Developers/telemetry`

libtelemetry

- Library to import/export data between processes
- Built on top of libshdata
- Non-blocking
- Uses shm
- Timestamped data
- `https://github.com/Parrot-Developers/libshdata`
- `https://github.com/Parrot-Developers/telemetry`

libtelemetry

- Library to import/export data between processes
- Built on top of libshdata
- Non-blocking
- Uses shm
- Timestamped data
- `https://github.com/Parrot-Developers/libshdata`
- `https://github.com/Parrot-Developers/telemetry`

libtelemetry

- Library to import/export data between processes
- Built on top of libshdata
- Non-blocking
- Uses shm
- Timestamped data
- `https://github.com/Parrot-Developers/libshdata`
- `https://github.com/Parrot-Developers/telemetry`

libtelemetry

- Library to import/export data between processes
- Built on top of libshdata
- Non-blocking
- Uses shm
- Timestamped data
- `https://github.com/Parrot-Developers/libshdata`
- `https://github.com/Parrot-Developers/telemetry`

Video implementation with ArduPilot

Video implementation with ArduPilot

AP_Module

- Dynamically linked plugin
- Implements hooks called by different subsystems in ArduPilot
- Needs a plugin implementation

AP_Module

- Dynamically linked plugin
- Implements hooks called by different subsystems in ArduPilot
- Needs a plugin implementation

AP_Module

- Dynamically linked plugin
- Implements hooks called by different subsystems in ArduPilot
- Needs a plugin implementation

libapm-telemetry

ap_hook_AHRS_update

```c
void ap_hook_AHRS_update(const struct AHRS_state *state)
{
        struct timespec ts;
        int ret;
        uint64_t time_us;

        /* check structure version */
        if (state->structure_version != AHRS_state_version) {
                ULOGE("Wrong structure version");
                return;
        }

        /* copy AHRS data to local structures */
        memcpy(export.body_quaternion, state->quat,
                        sizeof(export.body_quaternion));

        /* export the data in telemetry */
        time_us = state->time_us + export.time_offset;
        ret = time_us_to_timespec(&time_us, &ts);
        if (ret < 0) {
                ULOGE("error converting timespec to us %s",
                                strerror(-ret));
                return;
        }
        tlm_producer_put_sample(export.ahrs_producer, &ts);
}
```

Video on Disco

pimp-ctl

```
pimp-ctl stream-start 192.168.42.2 9999
```

gst-launch on remote side

```
gst-launch-1.0 udpsrc port=9999 ! ”application/x-rtp,
payload=96” ! rtph264depay ! avdec_h264 ! autovideosink
```

Building code for Parrot Disco

Building code for Parrot Disco

Build ardupilot for Disco

Using Alchemy

https://github.com/ncarrier/apm-disco-manifest

Using waf build system

```
git clone https://github.com/ArduPilot/ardupilot cd ardupilot git
submodule init git submodule update ./modules/waf/waf-light
configure –board=disco ./modules/waf/waf-light build -j6
```

What's left to do ?

What's left to do ?

Remaining tasks

- Image quality improvements
- MAVLINK support to start streaming
- Piloting from Skycontroller 2
- Allow users to develop video plugins
- Write a fully Open Source version of the video pipeline

Remaining tasks

- Image quality improvements
- MAVLINK support to start streaming
- Piloting from Skycontroller 2
- Allow users to develop video plugins
- Write a fully Open Source version of the video pipeline

Remaining tasks

- Image quality improvements
- MAVLINK support to start streaming
- Piloting from Skycontroller 2
- Allow users to develop video plugins
- Write a fully Open Source version of the video pipeline

Remaining tasks

- Image quality improvements
- MAVLINK support to start streaming
- Piloting from Skycontroller 2
- Allow users to develop video plugins
- Write a fully Open Source version of the video pipeline

Remaining tasks

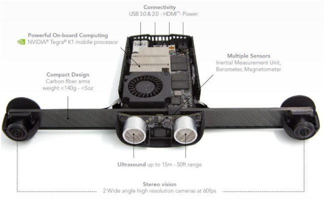
- Image quality improvements
- MAVLINK support to start streaming
- Piloting from Skycontroller 2
- Allow users to develop video plugins
- Write a fully Open Source version of the video pipeline

Parrot SLAM.dunk

Parrot SLAM.dunk

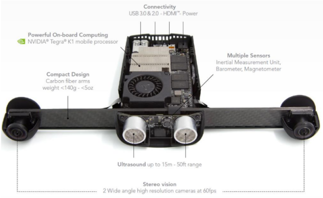
Parrot SLAM.dunk

- Development Kit for Autonomous Flight
- Tegra K1
- Stereo Cameras and sensors
- Ubuntu Desktop
- ROS support


Connectivity
USB 3.0 & 2.0 - HDMI™- Power
Powerful On-board Computing
NVIDIA® Tegra® K1 mobile processor
Multiple Sensors
Inertial Measurement Unit, Barometer, Magnetometer
Compact Design
Carbon fiber arms
weight <140g - <5oz
Ultrasound up to 15m - 50ft range
Stereo vision
2 Wide angle high resolution cameras at 60fps

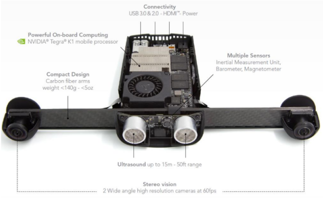
Parrot SLAM.dunk

- Development Kit for Autonomous Flight
- Tegra K1
- Stereo Cameras and sensors
- Ubuntu Desktop
- ROS support


Connectivity
USB 3.0 & 2.0 - HDMI™- Power
Powerful On-board Computing
NVIDIA® Tegra® K1 mobile processor
Multiple Sensors
Inertial Measurement Unit,
Barometer, Magnetometer
Compact Design
Carbon fiber arms
weight <140g - <5oz
Ultrasound up to 15m - 50ft range
Stereo vision
2 Wide angle high resolution cameras at 60fps

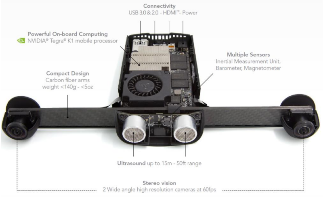
Parrot SLAM.dunk

- Development Kit for Autonomous Flight
- Tegra K1
- Stereo Cameras and sensors
- Ubuntu Desktop
- ROS support


Connectivity
USB 3.0 & 2.0 - HDMI™- Power
Powerful On-board Computing
NVIDIA® Tegra® K1 mobile processor
Multiple Sensors
Inertial Measurement Unit,
Barometer, Magnetometer
Compact Design
Carbon fiber arms
weight <140g - <5oz
Ultrasound up to 15m - 50ft range
Stereo vision
2 Wide angle high resolution cameras at 60fps

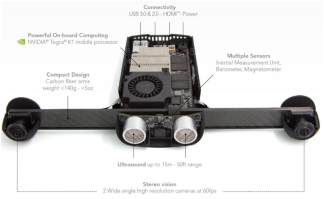
Parrot SLAM.dunk

- Development Kit for Autonomous Flight
- Tegra K1
- Stereo Cameras and sensors
- Ubuntu Desktop
- ROS support


Connectivity
USB 3.0 & 2.0 - HDMI™- Power
Powerful On-board Computing
NVIDIA® Tegra® K1 mobile processor
Multiple Sensors
Inertial Measurement Unit,
Barometer, Magnetometer
Compact Design
Carbon fiber arms
weight <140g - <5oz
Ultrasound up to 15m - 50ft range
Stereo vision
2 Wide angle high resolution cameras at 60fps

Parrot SLAM.dunk

- Development Kit for Autonomous Flight
- Tegra K1
- Stereo Cameras and sensors
- Ubuntu Desktop
- ROS support


Connectivity
USB 3.0 & 2.0 - HDMI™- Power
Powerful On-board Computing
NVIDIA® Tegra® K1 mobile processor
Multiple Sensors
Inertial Measurement Unit,
Barometer, Magnetometer
Compact Design
Carbon fiber arms
weight <140g - <5oz
Ultrasound up to 15m - 50ft range
Stereo vision
2 Wide angle high resolution cameras at 60fps

Conclusion

Questions ?