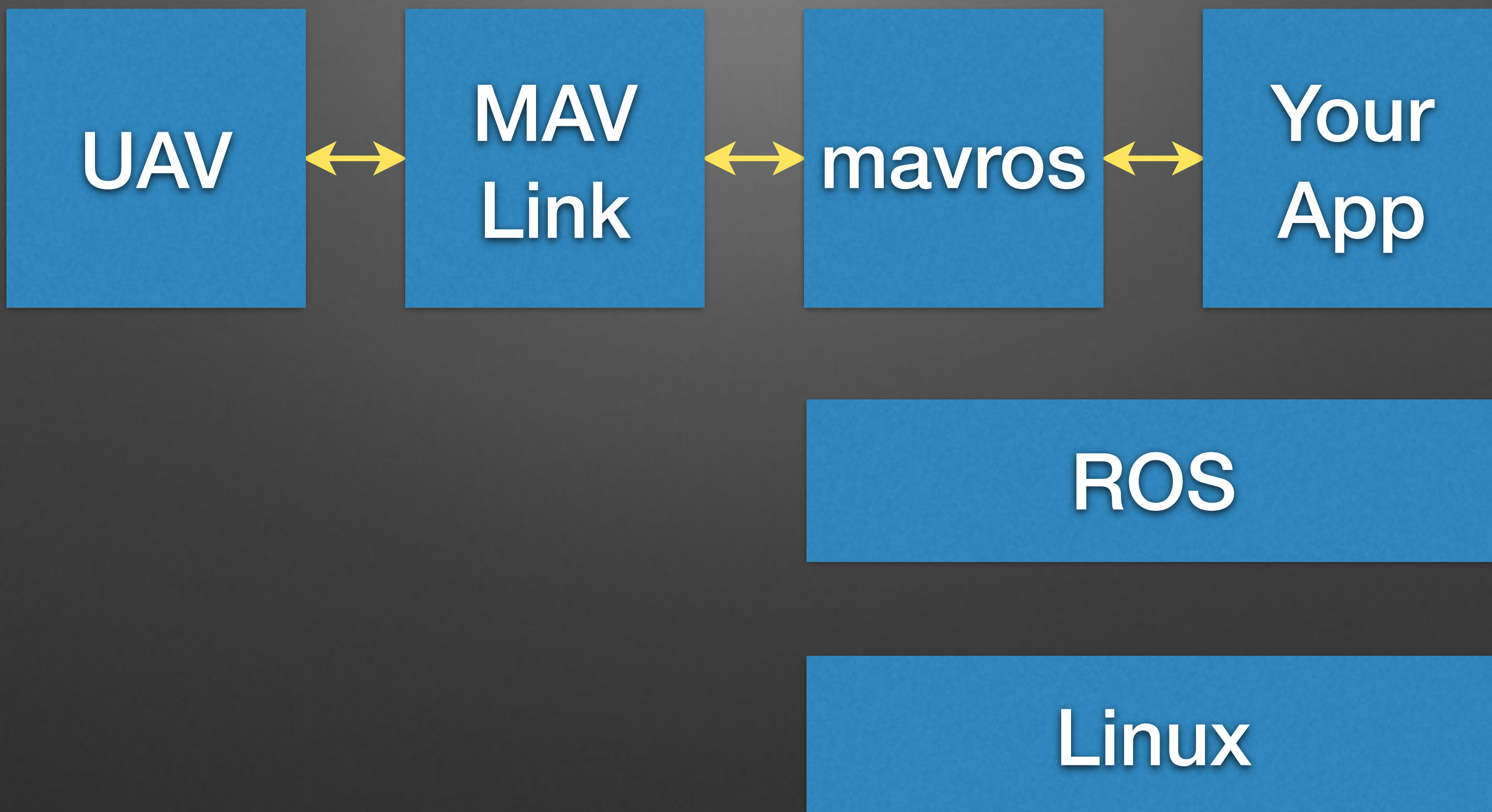
Flying Penguins

Embedded Linux applications for autonomous UAVs

Clay McClure

github.com/claymation

Roadmap



autopilot


OTTO
IS MY CO-PILOT

px4 autopilot

APM
MULTIPLATFORM AUTOPILOT

RC input
motor mixing
stabilization
telemetry
missions
failsafes


FLIGHT DATA
FLIGHT PLAN
INITIAL SETUP
CONFIG/TUNING
SIMULATION
TERMINAL
HELP
DONATE
COM3
115200
CONNECT
Distance: 0.7989 km
Prev: 522.46 m AZ: 67
Home: 462.94 m
1
2
3
4
5
Home
Zoom
Action
GEO
-35.040907
117.832747
11.40
Grid
View KML
GoogleSatelliteMa
Status: loaded tiles
Load WP File
Save WP File
Read WPs
Write WPs
Home Location
Lat -35.04173272
Long 117.8277683
Alt (abs) 38
Waypoints
WP Radius 2
Loiter Radius
Default Alt 100
Absolute Alt
Verify Height
Add Below
Alt Warn 20
Command Lat Long Alt Delete Up Down Grad % Dist AZ
1 WAYPOINT 0 0 0 0 -35.0407928 117.8277898 100 X 95.7 104.5 1
2 WAYPOINT 0 0 0 0 -35.0406786 117.8260410 100 X 0.0 159.7 275
3 WAYPOINT 0 0 0 0 -35.0417239 117.8251612 100 X 0.0 141.2 215
4 WAYPOINT 0 0 0 0 -35.0428395 117.8259873 100 X 0.0 145.1 149
5 WAYPOINT 0 0 0 0 -35.0427165 117.8274572 100 X 0.0 134.5 84

AUTO
PILOT

≠

AUTO
NOMOUS

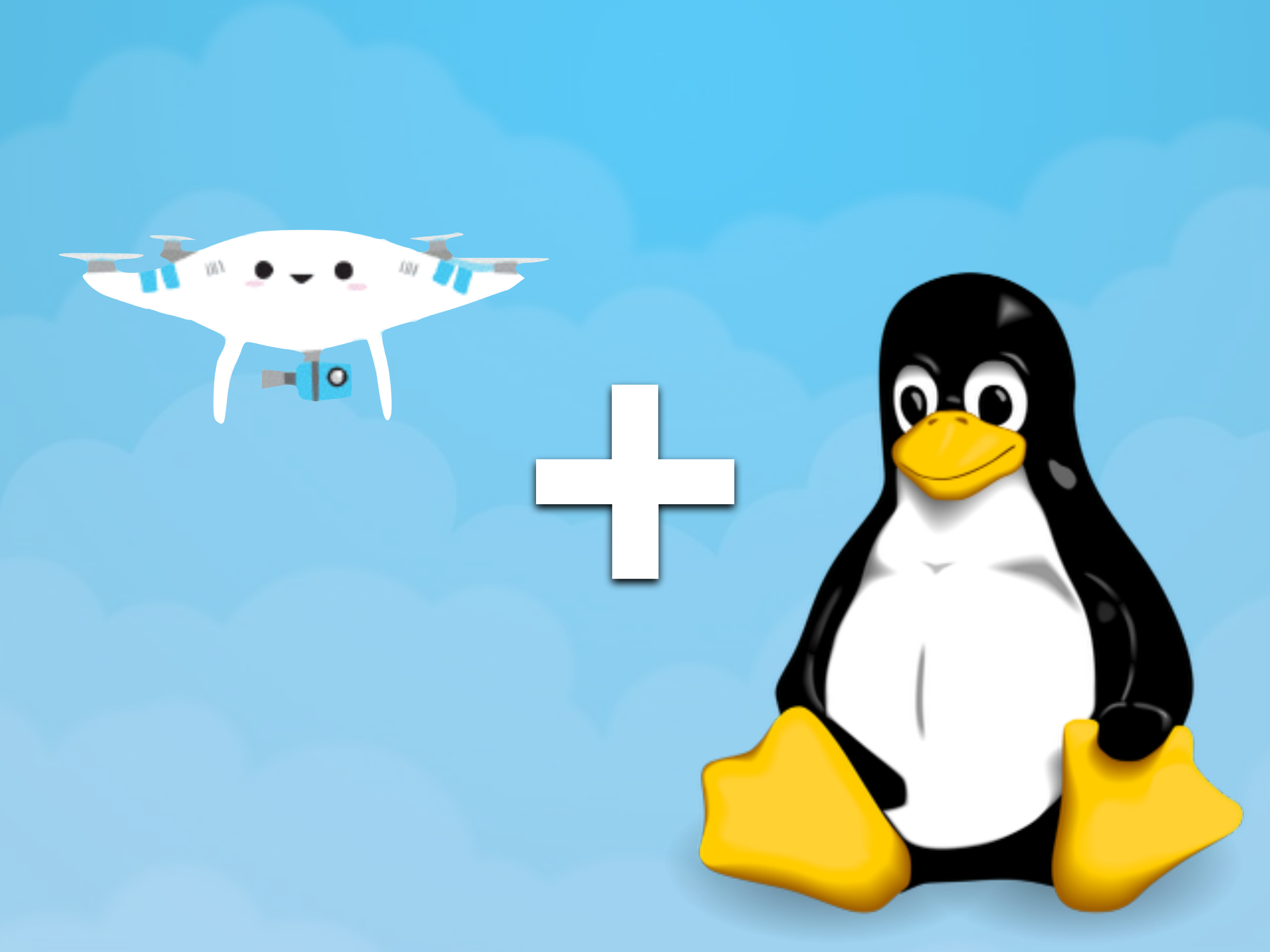
"system finds its own goal positions"

where to go

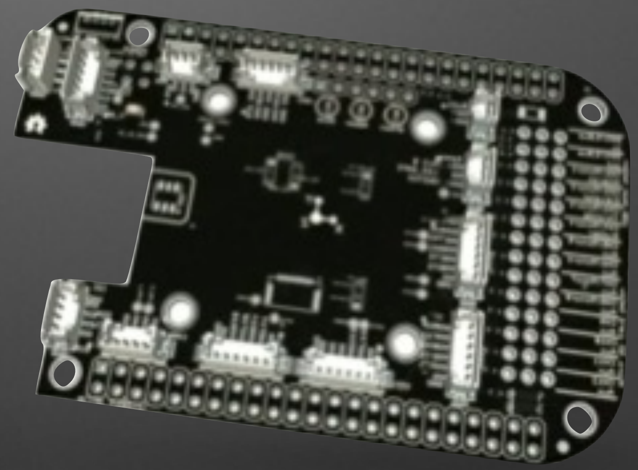
how to get there

what to do next

SO MANY ALGORITHMS, SO LITTLE COMPUTER



Autopilot *runs on* Linux

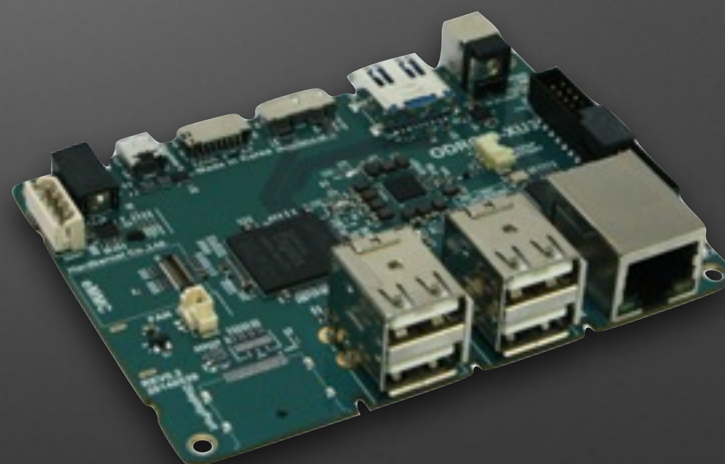


Autopilot *talks to* Linux

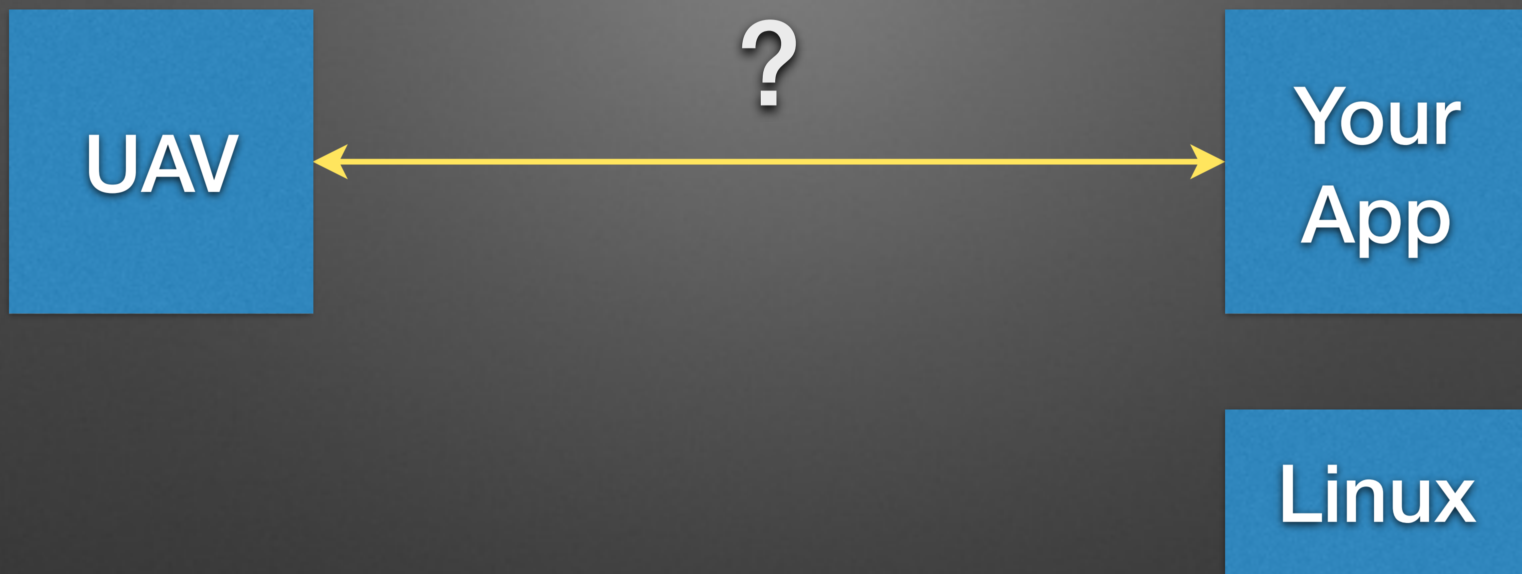
ODROID

ODROID-XU3 Lite

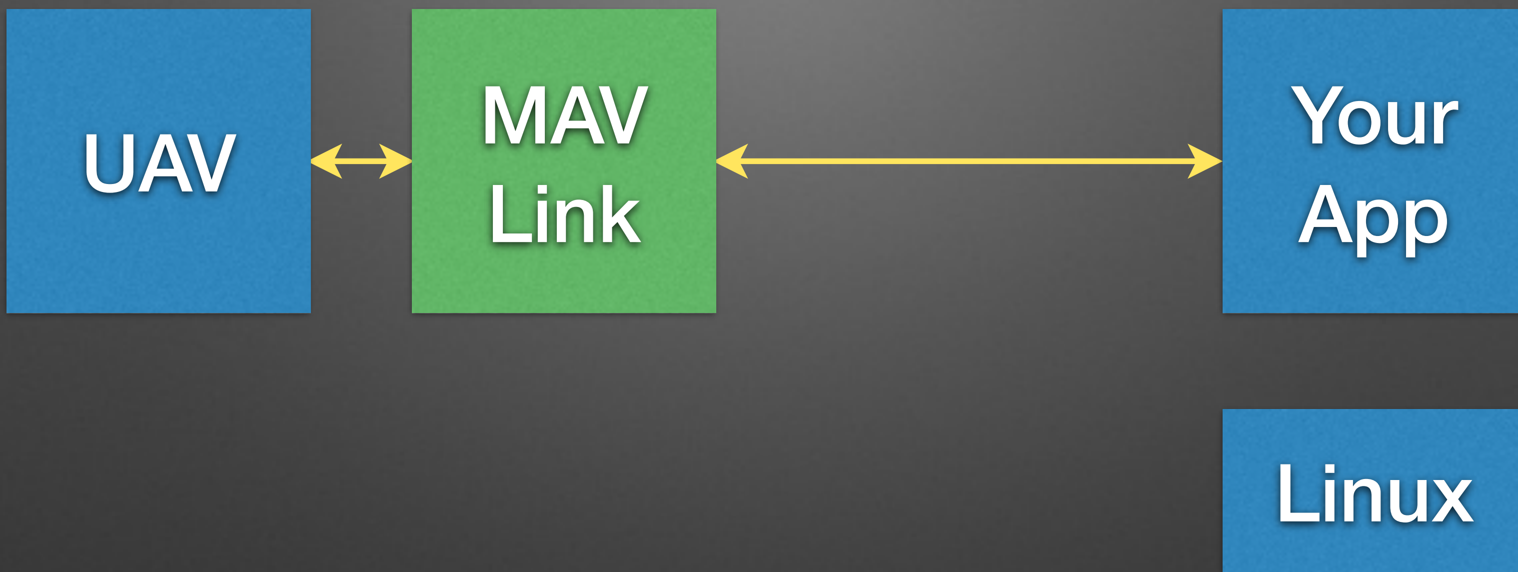
- Samsung Exynos5422 octa core
  - 4x Cortex™-A15 2.0GHz
  - 4x Cortex™-A7 1.4GHz
- 2 GB RAM
- 32+ GB flash
- 4x USB 2.0 + 1x USB 3.0



Roadmap (so far)



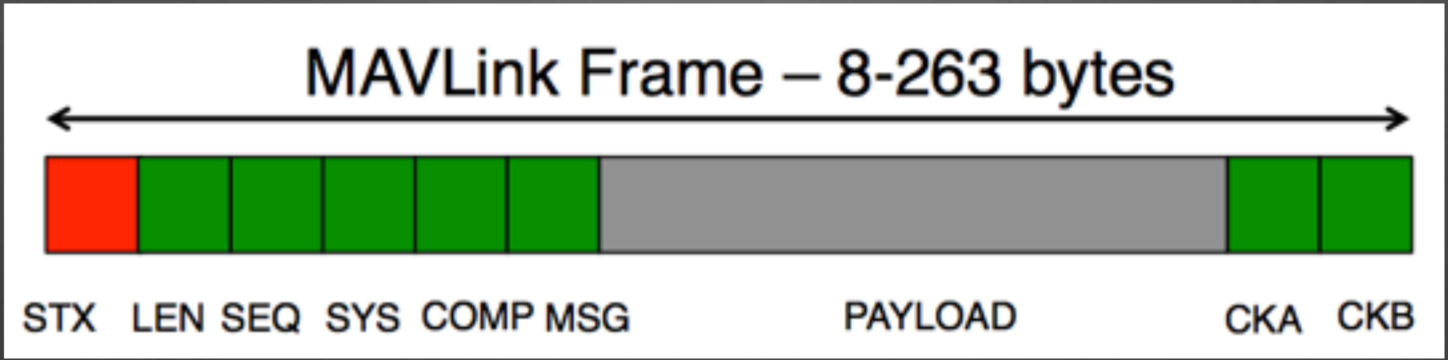
Roadmap (so far)



MAVLink is the HTTP of drones

(it’s also the libcurl)

MAVLink


MAVLink Frame – 8-263 bytes
STX
LEN
SEQ
SYS
COMP
MSG
PAYLOAD
CKA
CKB

status

configuration

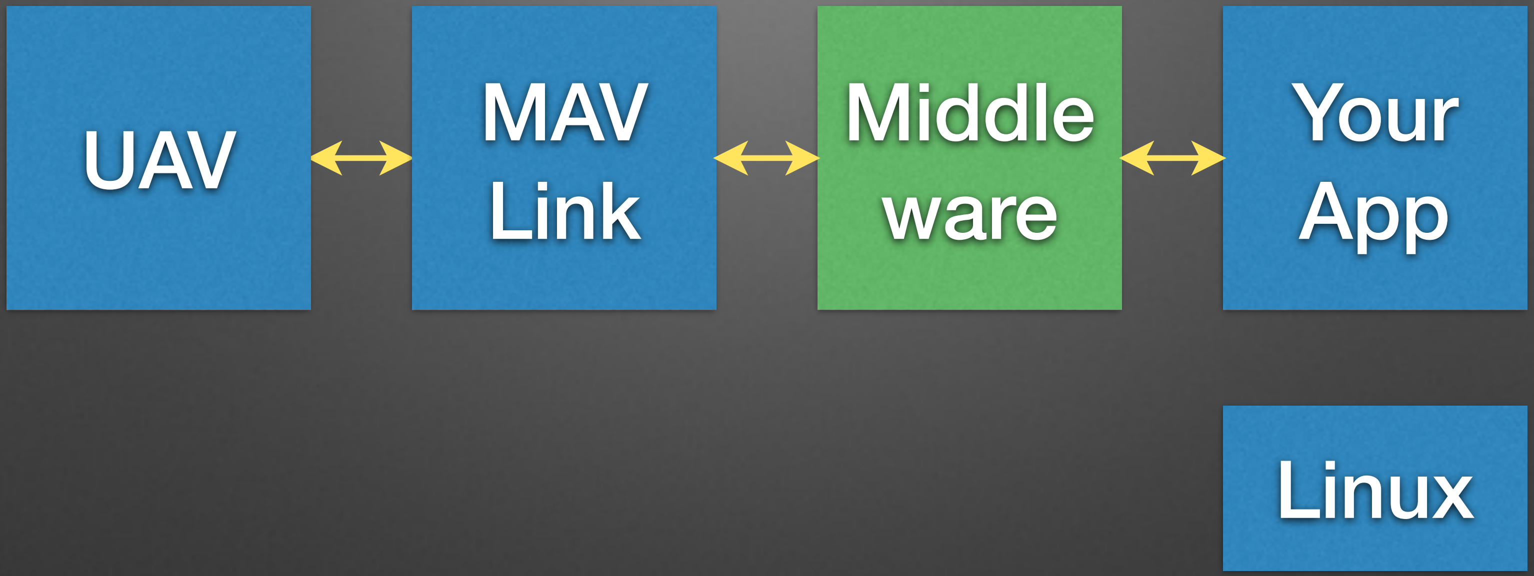
position / attitude

setpoints

missions

Roadmap (so far)

Roadmap (so far)



Middleware

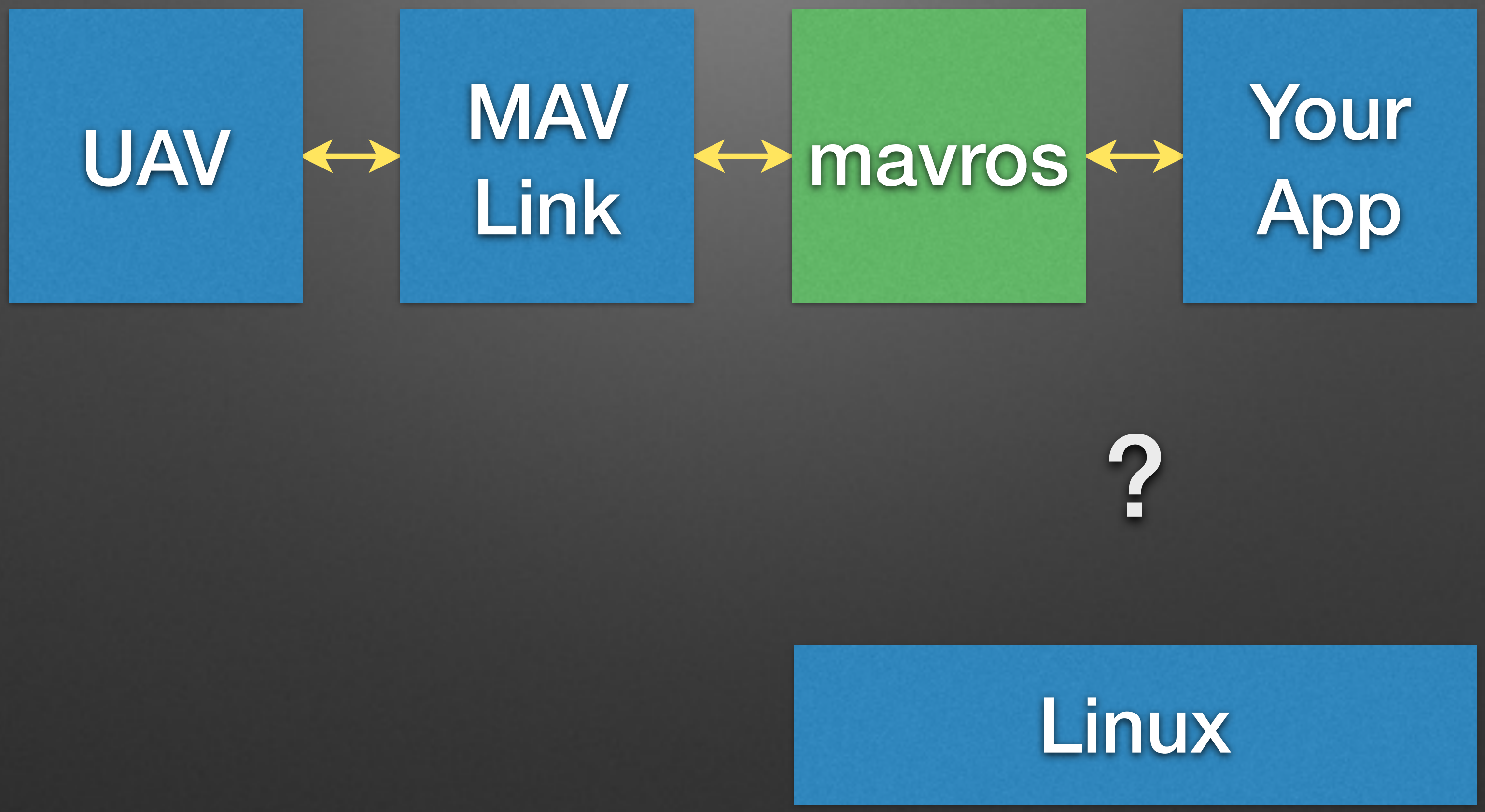
DroneAPI

- Python
- Go to Kevin Hester’s talk tomorrow

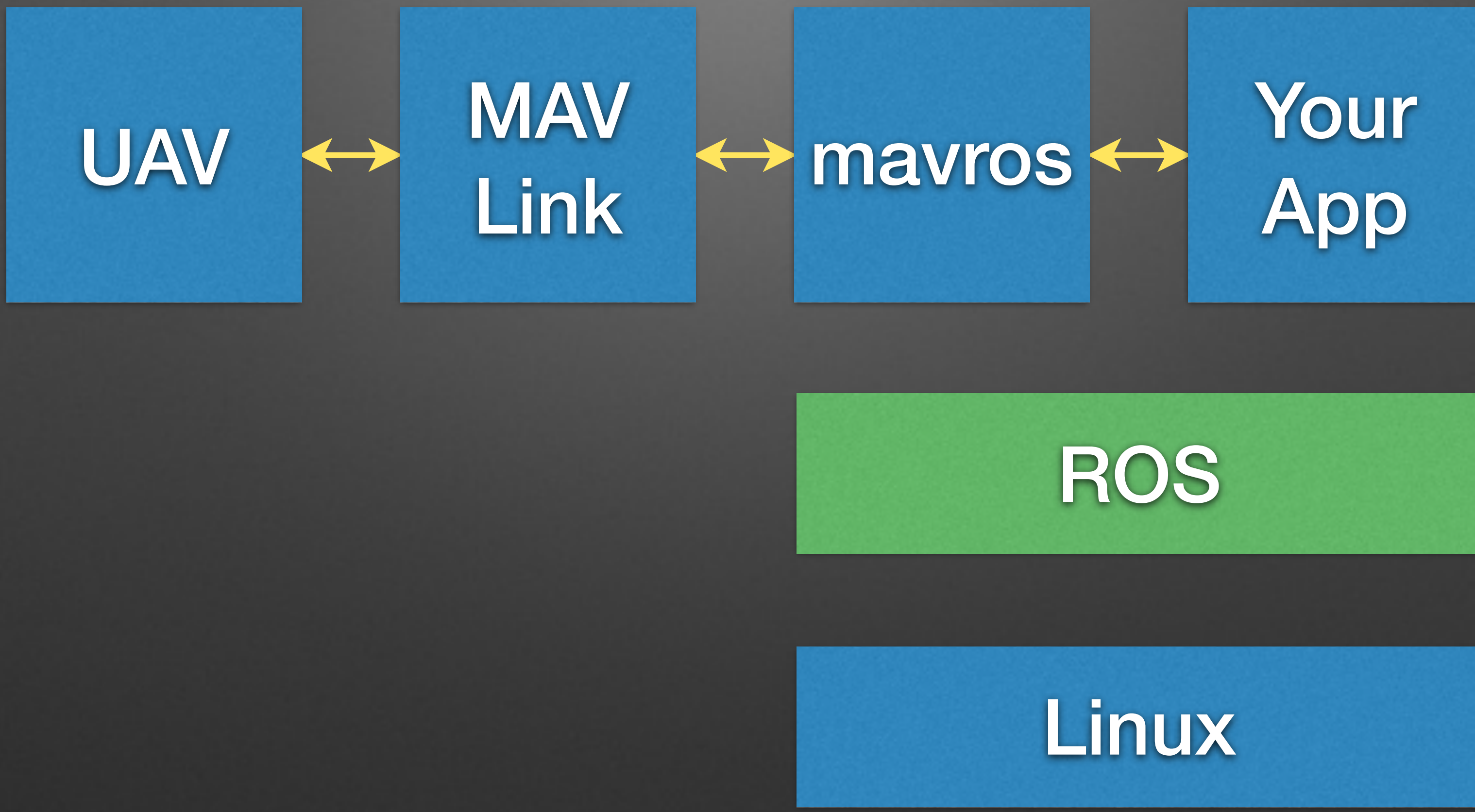
mavros

- Python, C++, Lisp (really)
- Access to a wealth of robotics research and tools

Roadmap (so far)



Roadmap (so far)



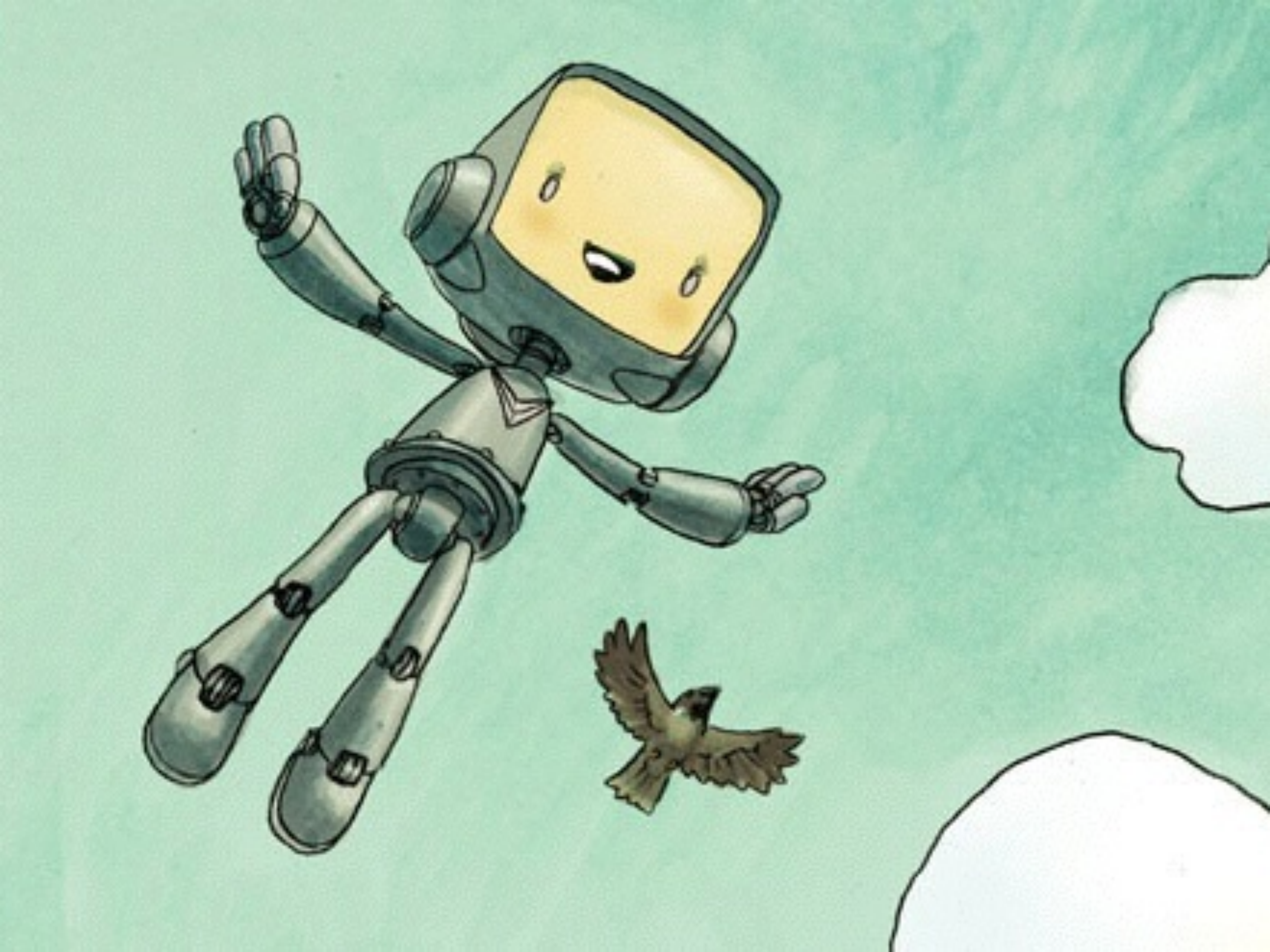
ROS
CRASH
COURSE

Robot Operating System

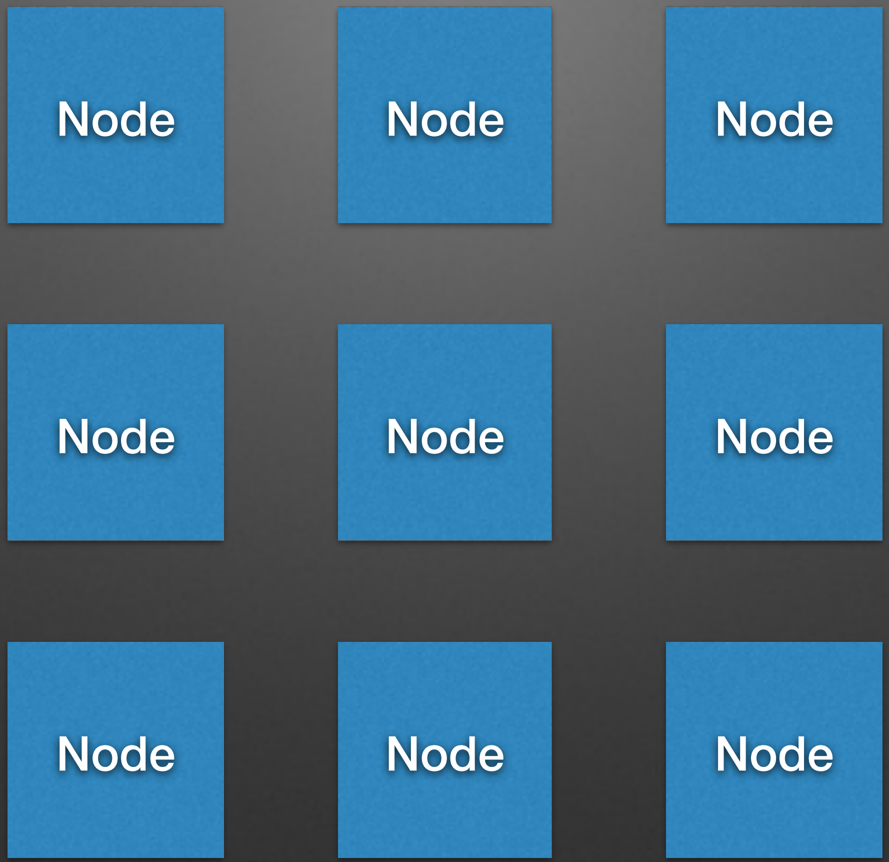
"ROS is an open-source, meta-operating system for your robot."

Robot Operating System

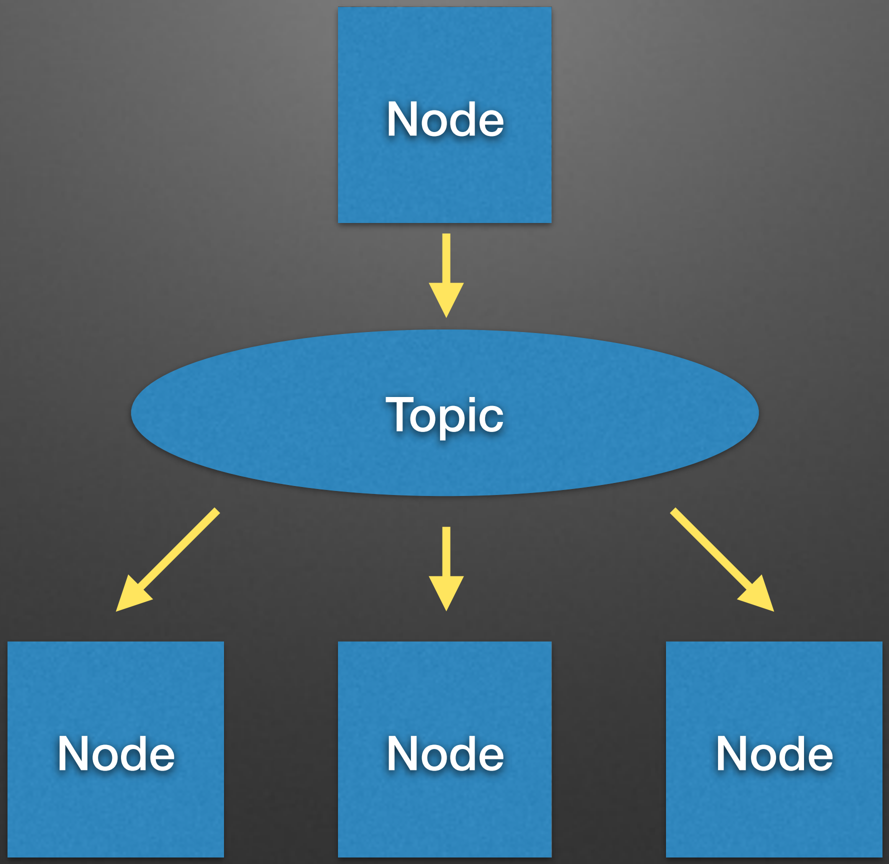
“ROS is an open-source, meta-operating system for your robot.”



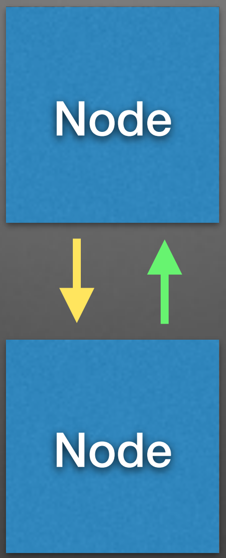
Nodes



Topics



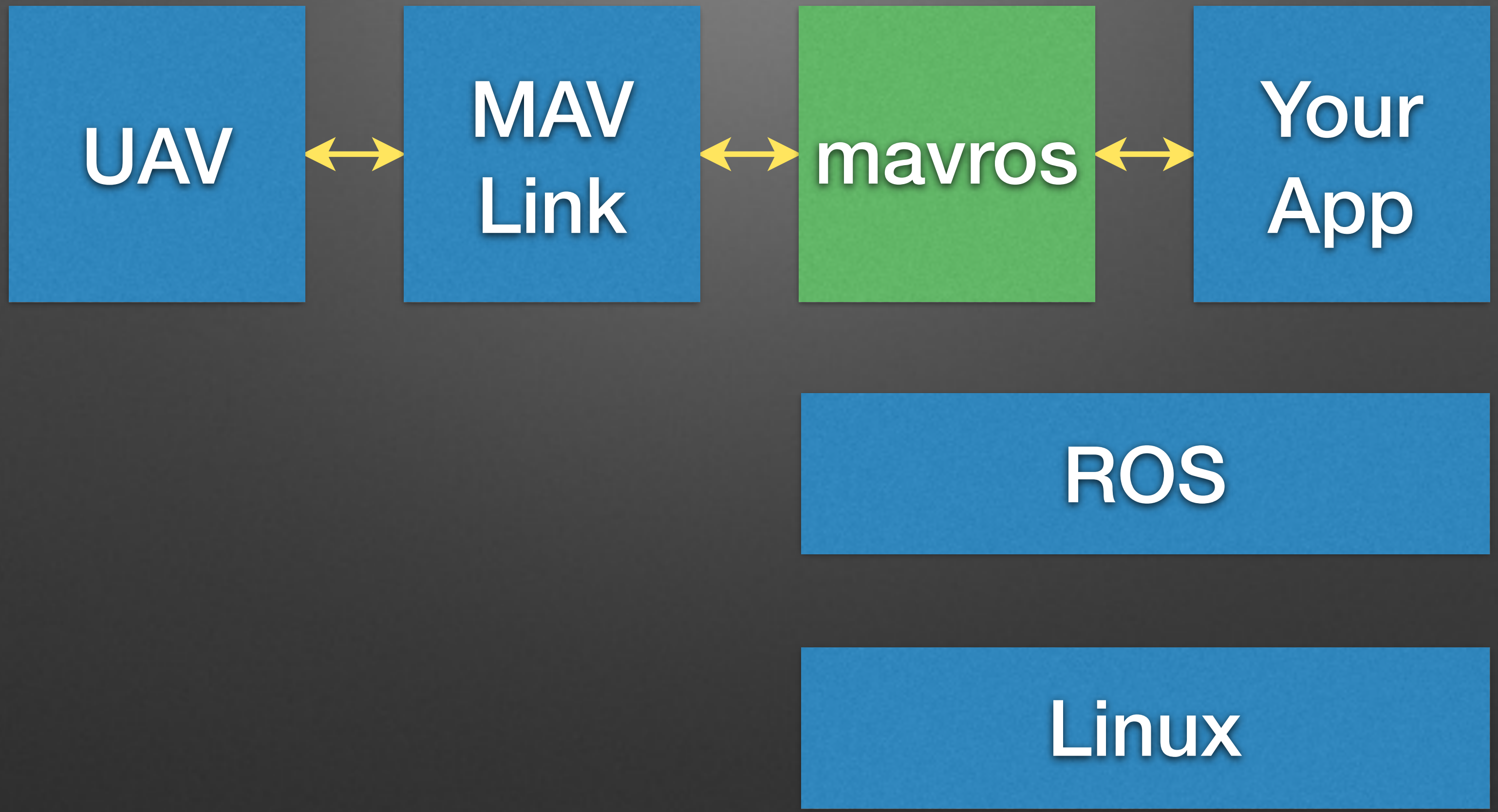
Services



but wait, that’s not all…

parameters
dynamic reconfig
coordinate frames
transformations
record/playback
visualization
logging

Roadmap



mavros is the
Babel fish of drones

Topics

/mavros/state

/mavros/imu/data

/mavros/global_position/global

/mavros/local_position/local

/mavros/setpoint_position/local_position

/mavros/setpoint_velocity/cmd_vel

Services

/mavros/cmd/arming

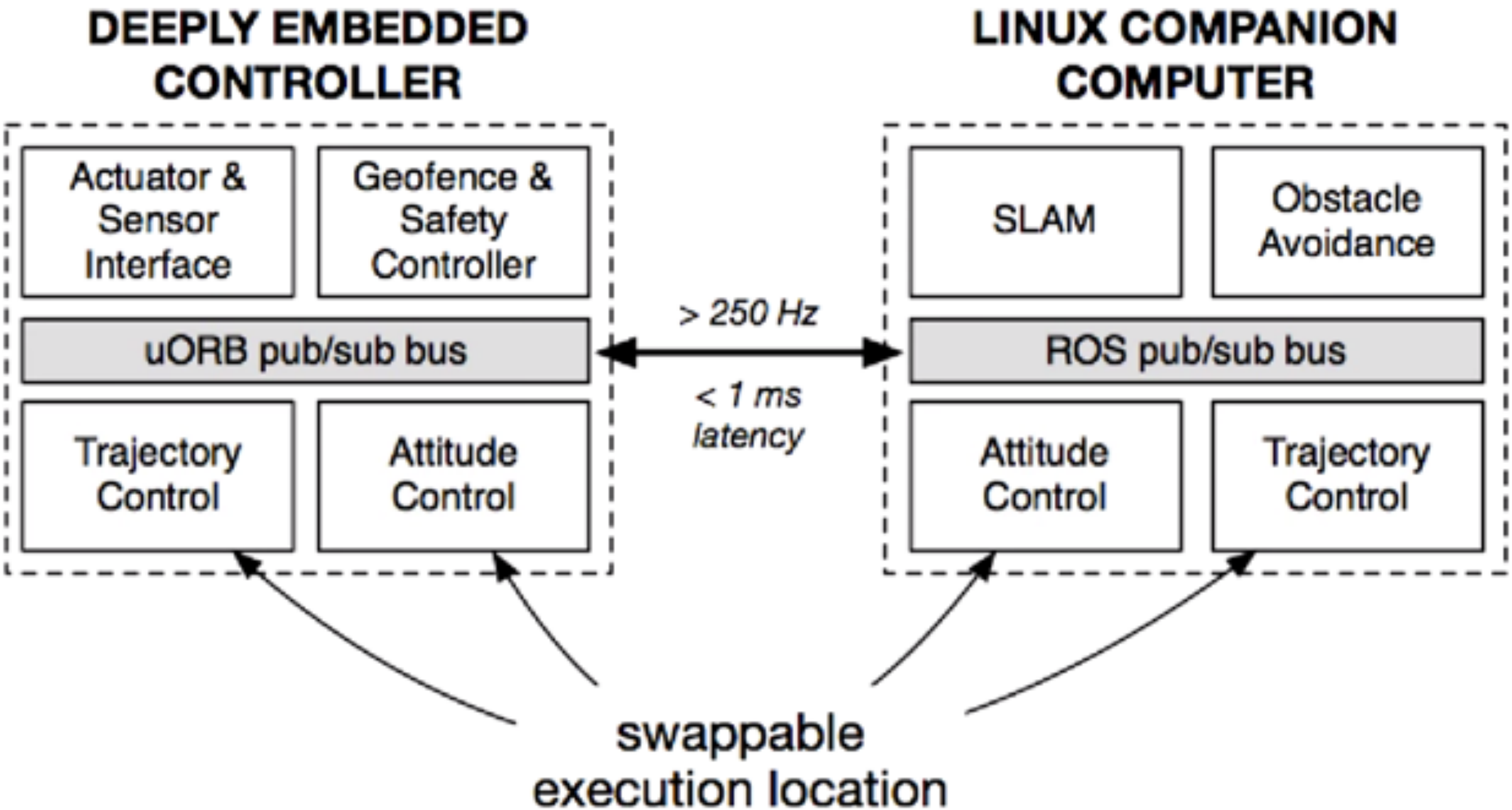
/mavros/cmd/land

/mavros/cmd/takeoff

/mavros/set_mode

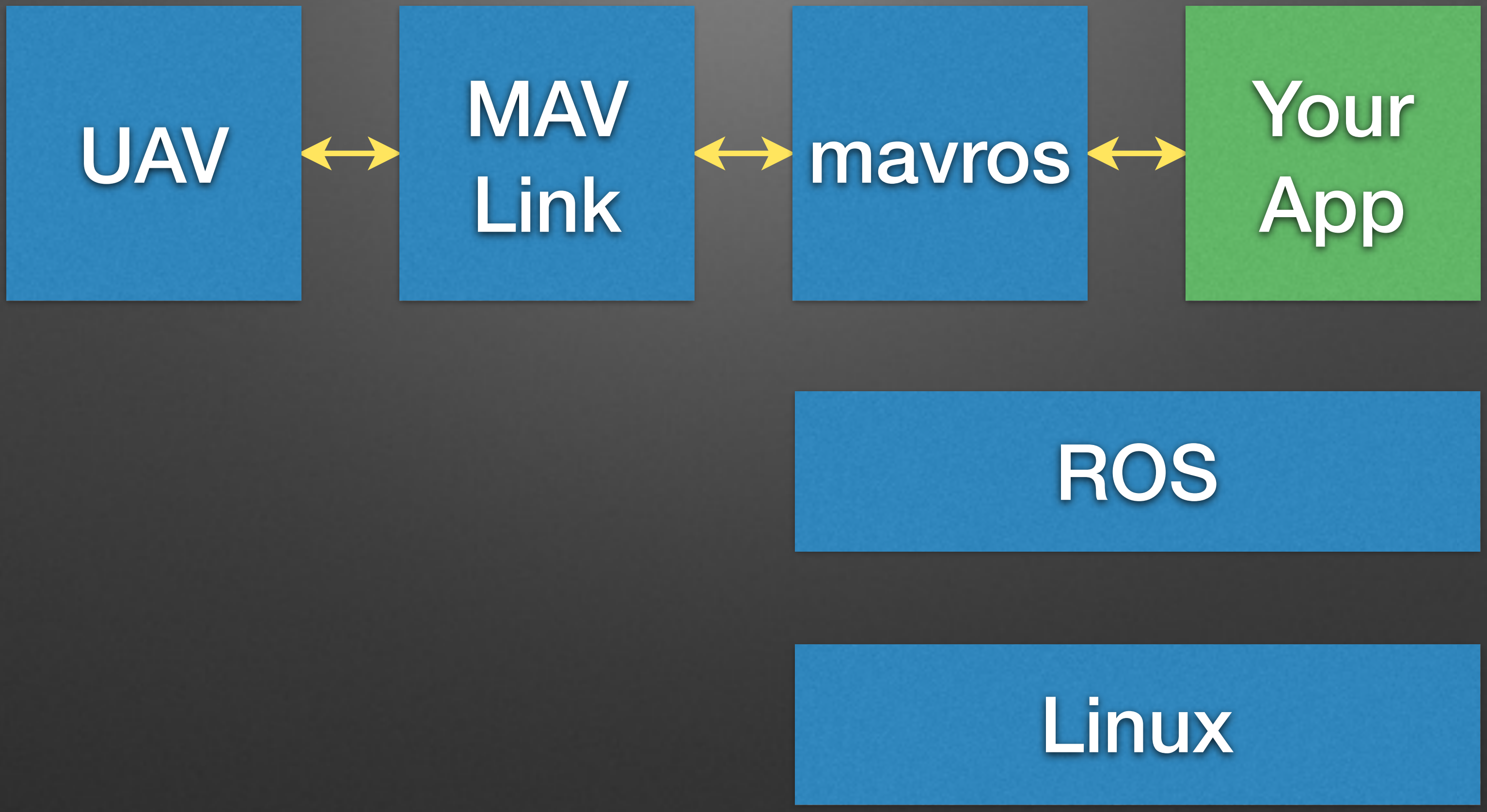
/mavros/set_stream_rate

PX4 + ROS



Credit: Kabir Mohammed

Roadmap



YAPL

Yet Another Precision Lander

Event-driven programming

- "Don't call me, I'll call you"
- Your application code responds to events
  - Message arrival
    - "my position is *(x, y, z)*"
  - Timer expiry
    - "it's time to run the control loop"

Nodes

- **Tracker**
  - Processes video stream, looks for landing pad
  - Publishes target position/velocity messages
- **Commander**
  - Subscribes to vehicle state and position messages
  - Subscribes to target tracker messages
  - Controls vehicle velocity

```python
class TrackerNode(object):

    def __init__(self):
        rospy.init_node("tracker")

        use_sim = rospy.get_param("~use_sim", False)
        camera_matrix = rospy.get_param("~camera_matrix")

        # ...

        self.image_publisher = \
            rospy.Publisher("tracker/image",
                sensor_msgs.msg.Image, queue_size=1)

        self.track_publisher = \
            rospy.Publisher("tracker/track",
                Track, queue_size=1)
```

```python
class TrackerNode(object):

    def __init__(self):
        rospy.init_node("tracker")

        use_sim = rospy.get_param("~use_sim", False)
        camera_matrix = rospy.get_param("~camera_matrix")

        # ...

        self.image_publisher = \
            rospy.Publisher("tracker/image",
                sensor_msgs.msg.Image, queue_size=1)

        self.track_publisher = \
            rospy.Publisher("tracker/track",
                Track, queue_size=1)
```

```
$ cat msg/Track.msg
# Whether we're tracking an object
std_msgs/Bool is_tracking

# Relative position and velocity of the tracked object
geometry_msgs/Vector3 position
geometry_msgs/Vector3 velocity
```

```python
def publish_track(self, position, velocity):
    msg = TrackStamped()
    msg.track.is_tracking.data = self.is_tracking

    if self.is_tracking:
        msg.track.position.x = position[0]
        msg.track.position.y = position[1]
        msg.track.position.z = position[2]
        msg.track.velocity.x = velocity[0]
        msg.track.velocity.y = velocity[1]
        msg.track.velocity.z = velocity[2]

    self.track_publisher.publish(msg)
```

```python
def publish_image(self, image):
    msg = self.image_bridge.cv2_to_imgmsg(image, "bgr8")
    self.image_publisher.publish(msg)
```

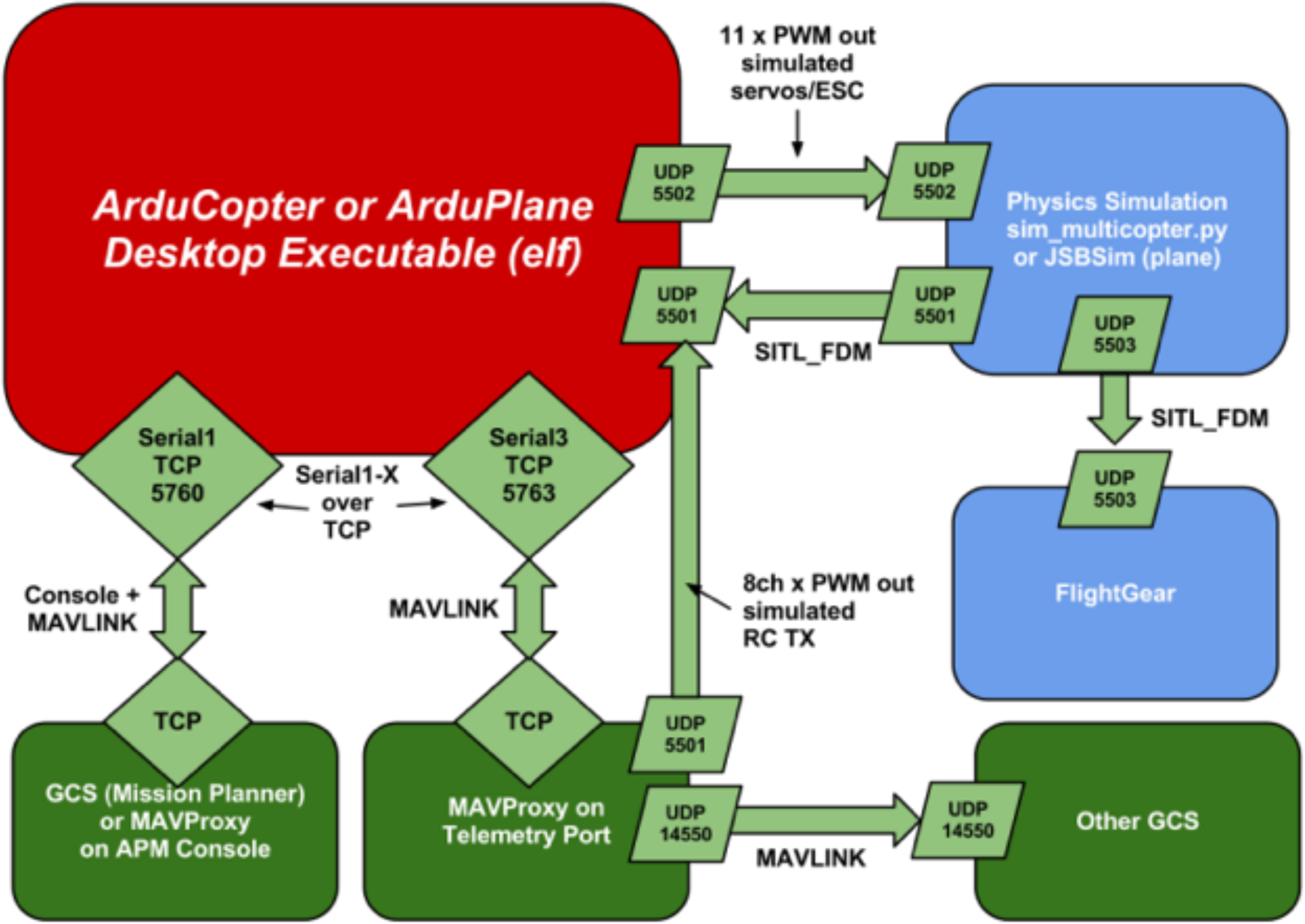
simulation

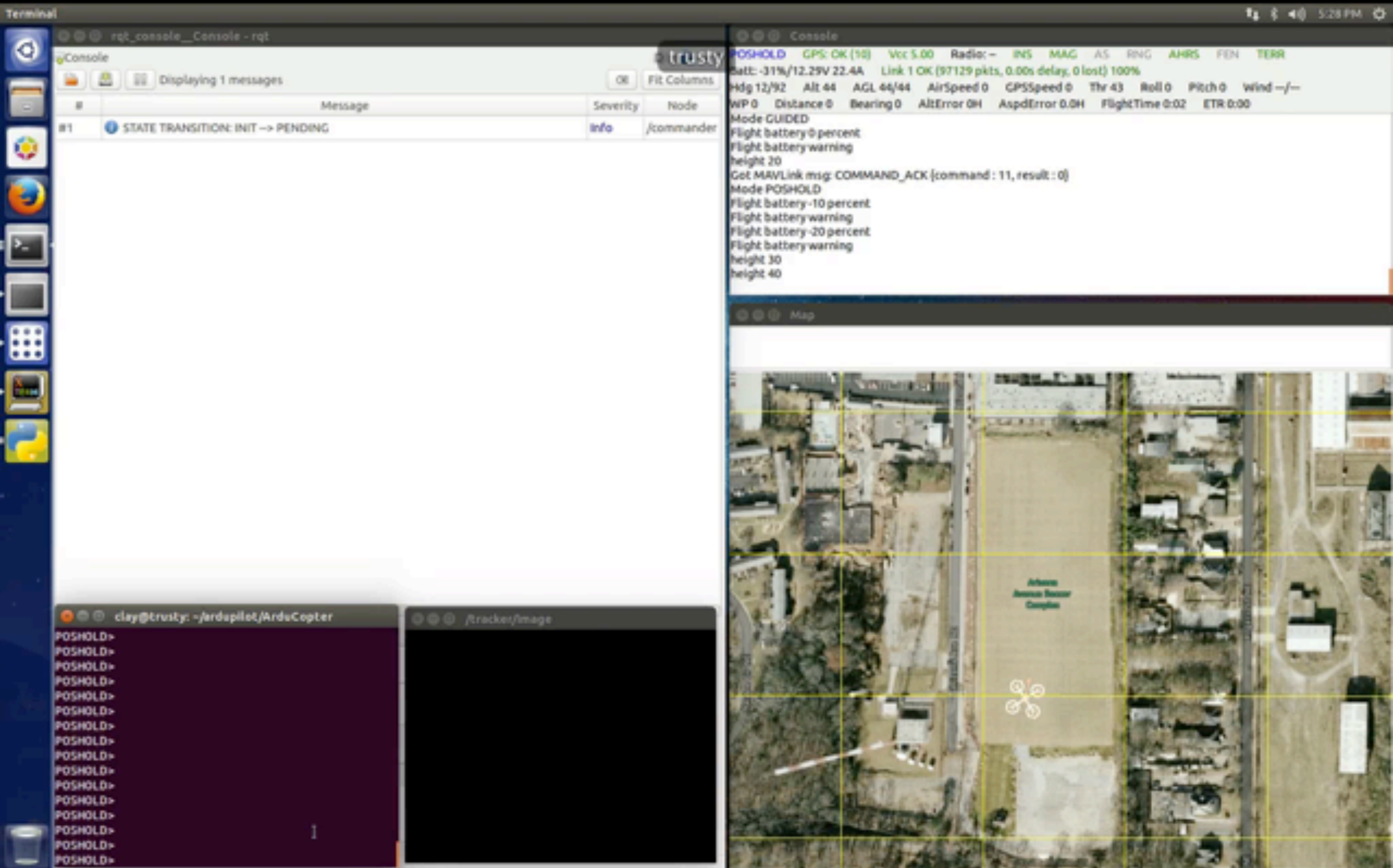
HITL

- Hardware in the loop
- Flight software runs on flight hardware
- Simulated sensor and control inputs

SITL

- Software in the loop
- Flight software runs on (Linux) desktop
- Simulated sensor and control inputs and HAL


ArduCopter or ArduPlane
Desktop Executable (elf)
11 x PWM out
simulated
servos/ESC
UDP
5502
UDP
5502
Physics Simulation
sim_multicopter.py
or JSBSim (plane)
UDP
5501
UDP
5501
SITL_FDM
UDP
5503
SITL_FDM
UDP
5503
FlightGear
Serial1
TCP
5760
Serial1-X
over
TCP
Serial3
TCP
5763
Console +
MAVLINK
MAVLINK
8ch x PWM out
simulated
RC TX
TCP
TCP
UDP
5501
GCS (Mission Planner)
or MAVProxy
on APM Console
MAVProxy on
Telemetry Port
UDP
14550
UDP
14550
MAVLINK
Other GCS


Terminal
5:28 PM
rqt_console__Console - rqt
Console
trusty
Displaying 1 messages
Fit Columns
#
Message
Severity
Node
#1
STATE TRANSITION: INIT --> PENDING
Info
/commander
Console
POSHOLD GPS: OK (10) Vcc 5.00 Radio: -- INS MAG AS RNG AHRS FEN TERR
Batt: -31%/12.29V 22.4A Link 1 OK (97129 pkts, 0.00s delay, 0 lost) 100%
Hdg 12/92 Alt 44 AGL 44/44 AirSpeed 0 GPSSpeed 0 Thr 43 Roll 0 Pitch 0 Wind --/--
WP 0 Distance 0 Bearing 0 AltError 0H AspdError 0.0H FlightTime 0:02 ETR 0:00
Mode GUIDED
Flight battery 0 percent
Flight battery warning
height 20
Got MAVLink msg: COMMAND_ACK {command : 11, result : 0}
Mode POSHOLD
Flight battery -10 percent
Flight battery warning
Flight battery -20 percent
Flight battery warning
height 30
height 40
Map
clay@trusty: ~/ardupilot/ArduCopter
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
POSHOLD>
/tracker/image

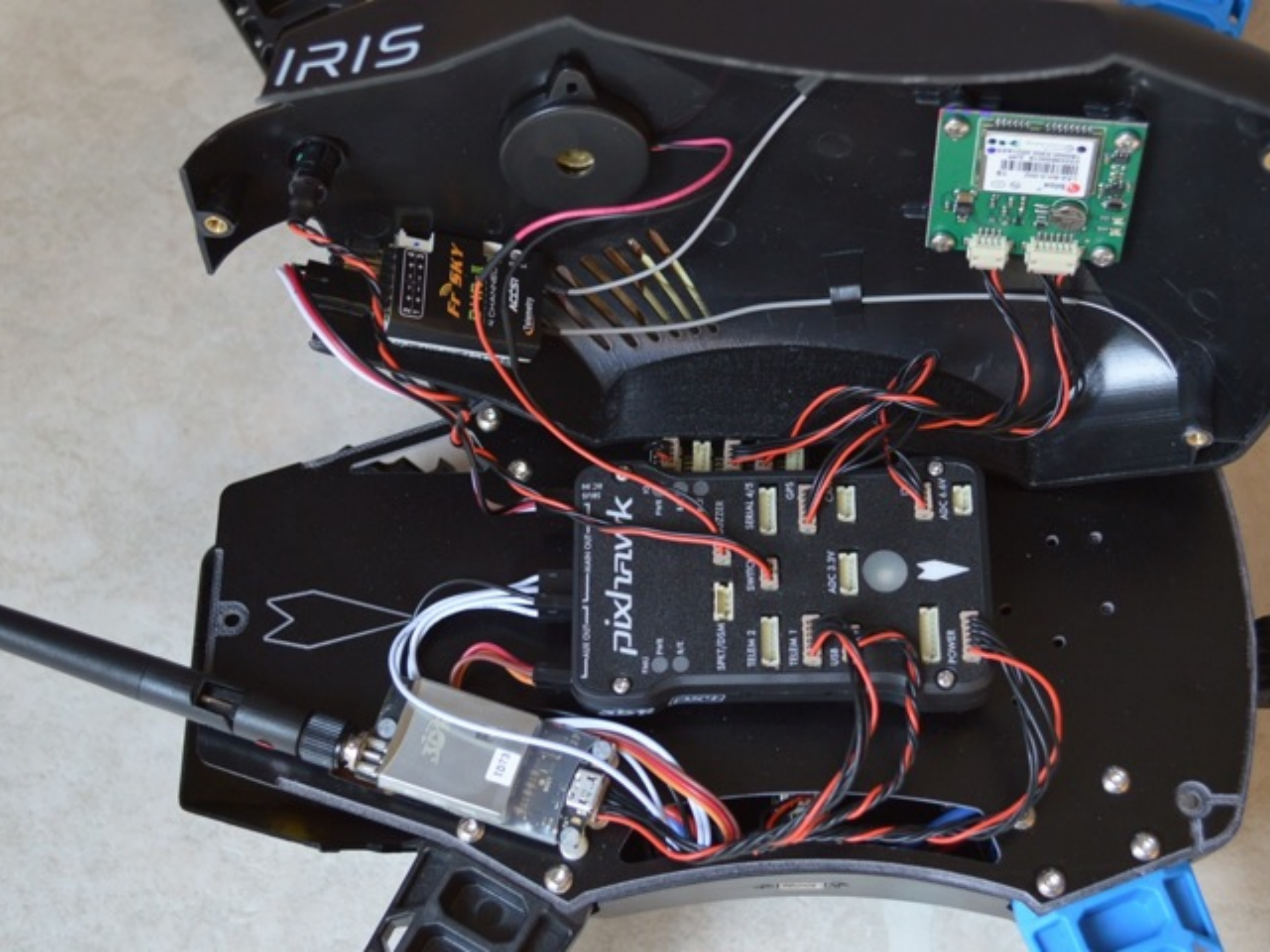

3D Software in the Loop Simulation (SITL)
16

“In theory there is no difference between theory and practice.

In practice there is.”

~ Yogi Berra

Practical Considerations


IRIS
FrSKY
pixhawk
SERIAL 4/5
GPS
ADC 6.6V
ADC 3.3V
SPKT/DSM
TELEM 2
TELEM 1
USB
POWER

Connections

- UART recommended
    - Requires 6-pin DF-13, possibly a level shifter
- USB works for me
    - Use hot glue gun
- `sudo apt-get remove modemmanager`

Power

5V 5A UBEC

ODROID
+
USB camera
+
WiFi
+
3S LiPo
=
5 hours

Launch files

- ROS feature that makes it easy to start and manage multiple nodes and their parameters

- `roslaunch lander lander.launch`

Startup

- Use ubuntu's upstart to launch ROS + mavros + application nodes
- `rosrun robot_upstart install \`
  `lander/launch/lander.launch`

Telemetry

- MAVLink + 3DR radio
- WiFi
  - Ad-Hoc mode (`man wireless`)
  - Need high-gain antenna and a tracker (helper)
  - `sudo apt-get remove wpasupplicant`
- GSM?

Coordinate Frames

- Global / Local
  - NED
  - ENU
- Body-fixed
- tf library

In closing…


search and rescue
conservation
3D mapping
cinematography
inspection
delivery
real estate
agriculture
disaster relief
law enforcement
pipeline monitoring
atmospheric research

What will you make?

For more information…

ros.org
ardupilot.com
pixhawk.org/start
pixhawk.ethz.ch/mavlink
github.com/mavlink/mavros
github.com/claymation/lander