Auterion

Overview of multicopter control from sensors to motors

Matthias Grob and Mathieu Bresciani
Auterion

July 6, 2020

Matthias Grob

GitHub: MaEtUgR
PX4 maintainer since 2017
Focus: Multicopter flight control

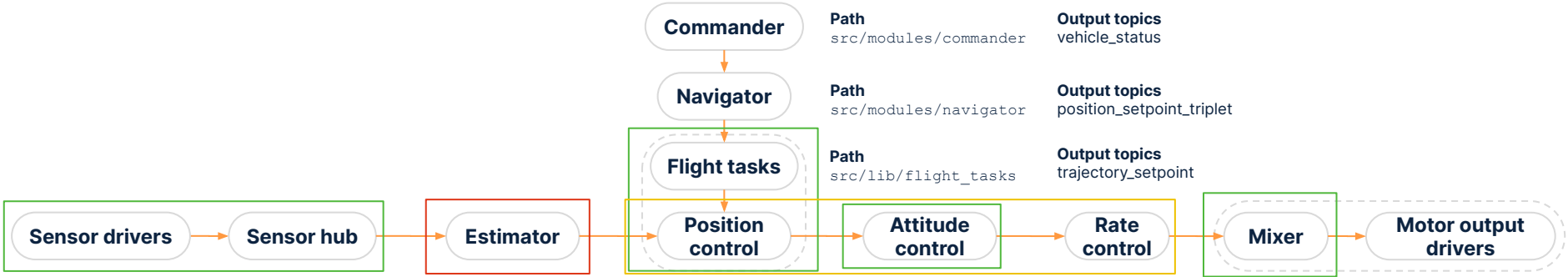
Mathieu Bresciani

GitHub: bresch
PX4 maintainer since 2018
Focus: estimation and multicopter control

Outline

1. **Multicopter overview**
2. **IMU pipeline**
   - Estimation and control paths
3. **Flight task update**
   - Setpoint continuity
   - Acceleration setpoints
4. **Quaternion attitude control**
5. **Control allocation (aka mixing)**
   - In a nutshell
   - Airmode

Multicopter overview

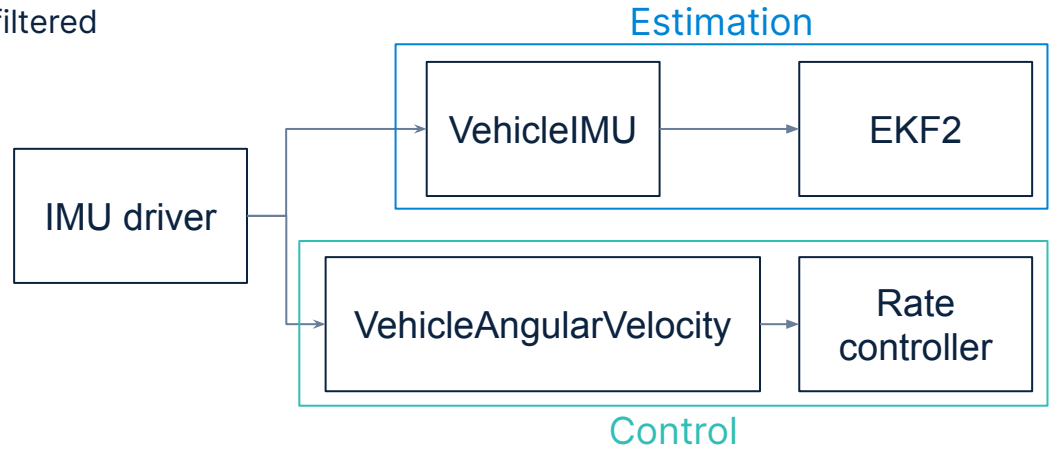


IMU pipeline

Two paths

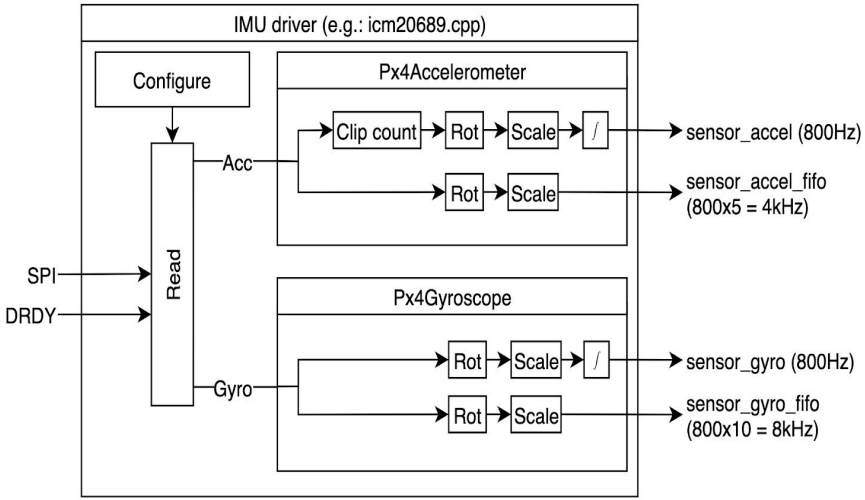
Estimation: low rate, unfiltered

Control: high rate, filtered

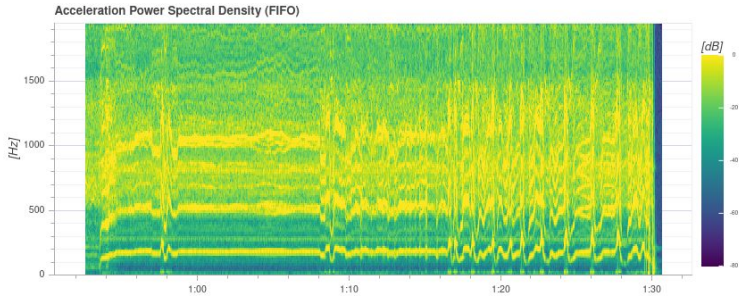


IMU pipeline

Driver



- Sensor_gyro frequency: IMU_GYRO_RATEMAX (400hz - 4khz) → rate controller frequency (default: 800Hz)
- Accelerometer clip counter used by EKF2 against asymmetric railing
- Activate FIFO logging using SDLOG_PROFILE
- Show FFT and spectrograms in logs.px4.io

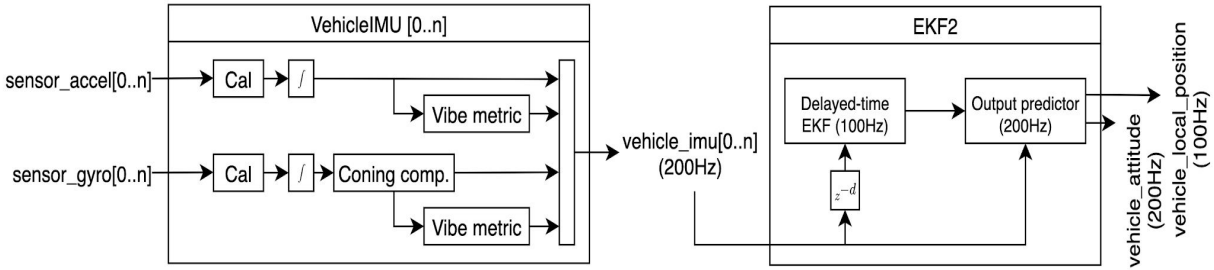


Auterion

IMU pipeline

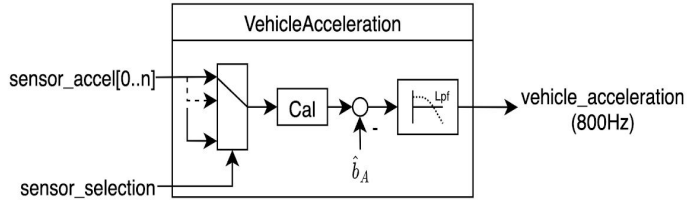
Estimation path

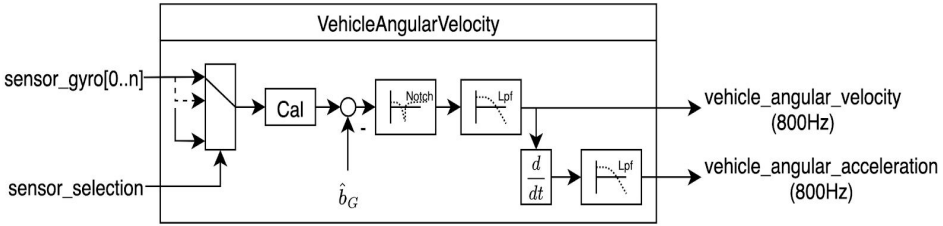
- Vehicle_imu and vehicle_attitude ODR: IMU_INTEG_RATE
- Delayed-time EKF frequency: 100Hz (hardcoded FILTER_UPDATE_PERIOD_MS)
- Vehicle_imu: 1 per IMU (New) - sensor_combined: voted IMU
- Multi-EKF (1 per IMU) coming soon...



IMU pipeline

Control path





Low-pass filters (Butterworth 2nd order) cutoffs:

- Acceleration: IMU_ACCEL_CUTOFF
- Angular velocity: IMU_GYRO_CUTOFF
- Angular acceleration: IMU_DGYRO_CUTOFF

Notch filter (gyro only):

- Notch frequency: IMU_GYRO_NF_FREQ
- Bandwidth: IMU_GYRO_NF_BW

Flight task update

- Setpoint continuity when switching mode / task
  - Problem: Vehicle state not enough
  - Idea: Hand over last setpoint set from previous task during switch

```
/**
 * Call once on the event where you switch to the task
 * @param last_setpoint last output of the previous task
 * @return true on success, false on error
 */
virtual bool activate(vehicle_local_position_setpoint_s last_setpoint);
```

  - New task takes over setpoints

- Acceleration setpoint execution
  - See next

Acceleration setpoints

- Any setpoint combination
  - Every dimension x, y, z, yaw
  - Horizontal setpoint pair x, y

Flight task output - Position control input

`trajectory_setpoint`

Local world frame

- **3D position**
- **3D velocity**
- **3D acceleration**
- 3D jerk [log]
- ~~3D thrust~~
- Yaw (heading)
- Yawspeed

- Velocity control output is acceleration
  - Gains rescaled

    MPC_{XY/Z}_VEL_{P/I/D}_ACC

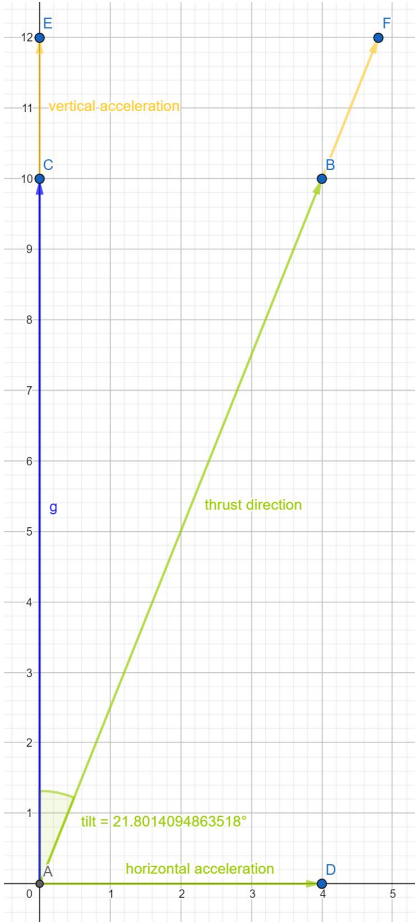
$$\frac{g}{hover\ thrust} \approx \frac{10\frac{m}{s^2}}{50\%} = 20$$

Acceleration setpoints

New strategy for attitude generation

- Problem: Collective thrust dynamics much faster than rotational dynamics
- Solution: Tilt independent of vertical acceleration
- Example
  - 4m/s² horizontal acceleration
    → tilt angle using gravity
  - 2m/s² upwards acceleration
    → adjust collective thrust

- Body z of attitude setpoint quaternion aligned with thrust direction



Flight task plans

Separate flight task

Instantiated in position control.

Inheritance → Libraries

Confusing structure limits reuse.

Goal: Sequential readability

Extend to “flight mode”

Allow rate and attitude setpoints to cover all modes.

Import navigator states

One “flight mode” for:
Takeoff, RTL, land, mission, ...

Use with other vehicle types

Can use “flight modes” since output is more flexible.

Hook up with state machine

Structure based on mode’s properties.

Quaternion attitude control

Principle

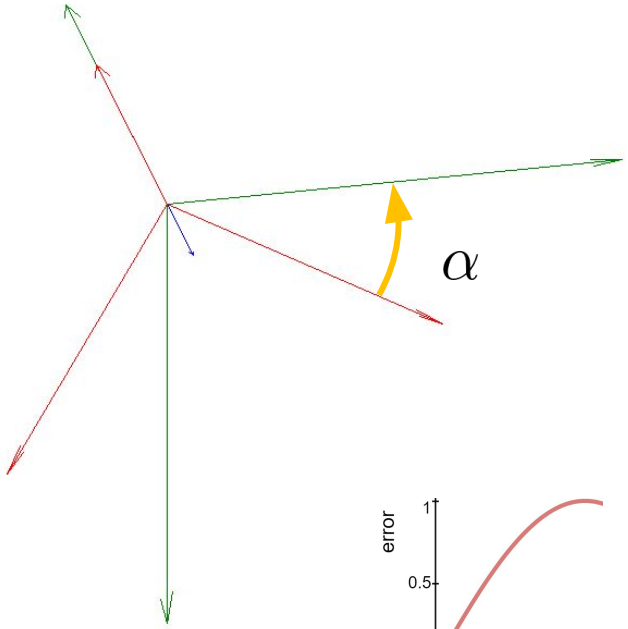
- Quaternion error

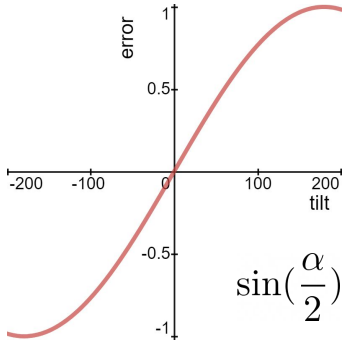
$$q_{setpoint} = q_{estimate} \cdot q_{error}$$

$$q_{estimate}^{-1} \cdot q_{setpoint} = q_{error}$$

- Angular velocity setpoint to correct

$$q_{error} = \begin{bmatrix} cos(\frac{\alpha}{2}) \\ sin(\frac{\alpha}{2})\vec{n} \end{bmatrix} \rightarrow \vec{\omega}_{setpoint}$$

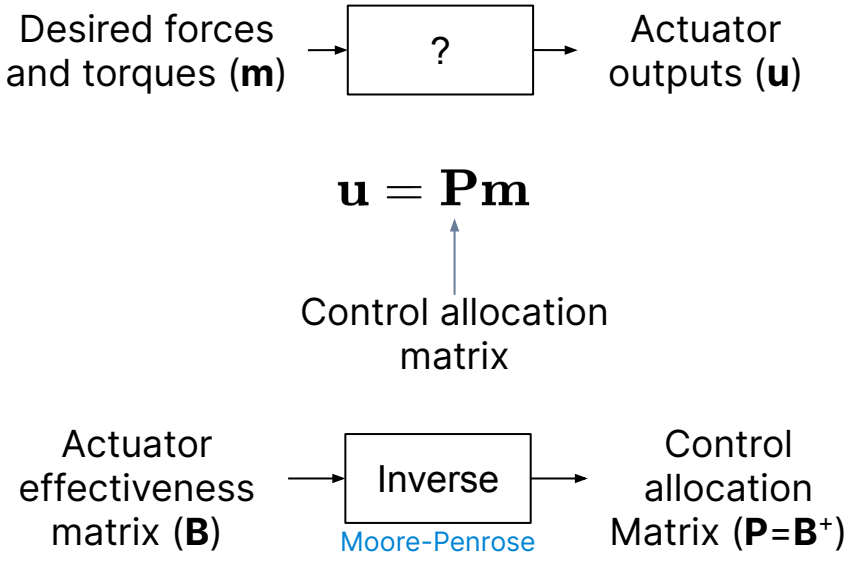




Nonlinear Quadrocopter Attitude Control
Dario Brescianini, Markus Hehn and Raffaello D'Andrea

Control allocation

In a nutshell



How to add a new geometry

1. Create new geometry file in src/lib/mixer/MultirotorMixer/geometries/foo.toml with a new key (e.g.: `key = "4fo"`) and add to CMakeLists.txt

Then in ROMFS/px4fmu_common/

2. Create new mixer file mixers/foo.main.mix with a line containing the new key:

```
R: 4fo 10000 10000 10000 0
```

3. Set the new mixer in init.d/airframes/myconfig

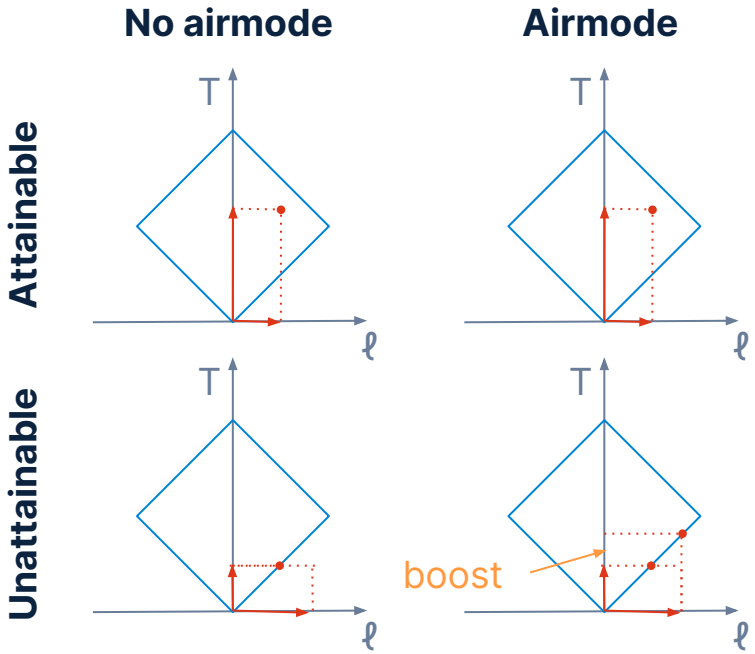
```
set MIXER foo
```

Control allocation

Airmode

- Parameter: MC_AIRMODE
- Use Roll/Pitch mode for VTOL planes and multirotors with weak yaw authority
- Use Roll/Pitch/Yaw mode for multirotors with strong yaw authority (e.g.: racers)
- For better performance, use thrust linearization: THR_MDL_FAC

Warning: Only activate airmode when the control loops are properly tuned!



Auterion

Thank you!

May your preflight check pass