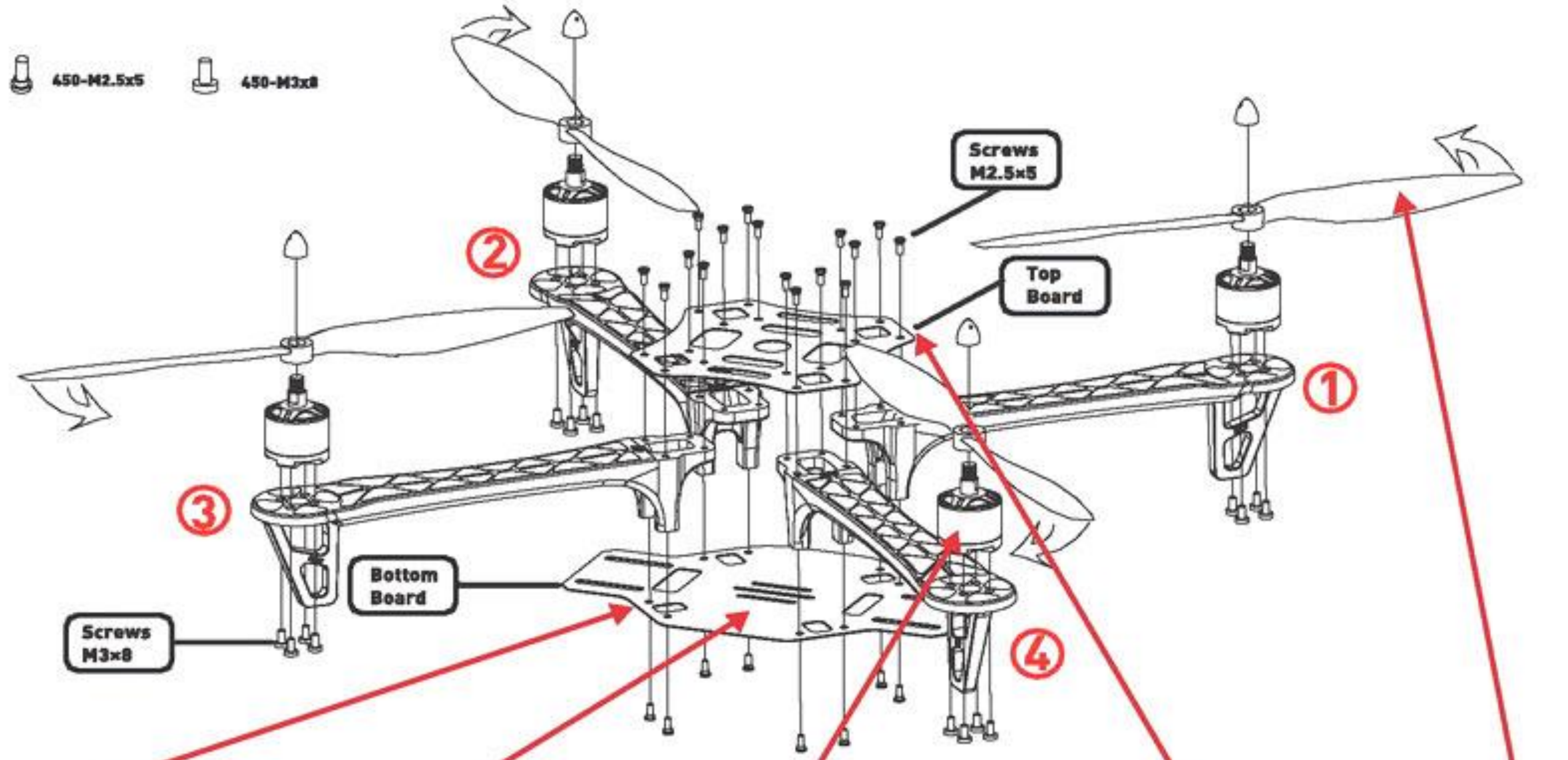
Quadrotor attitude control

Our quadcopter



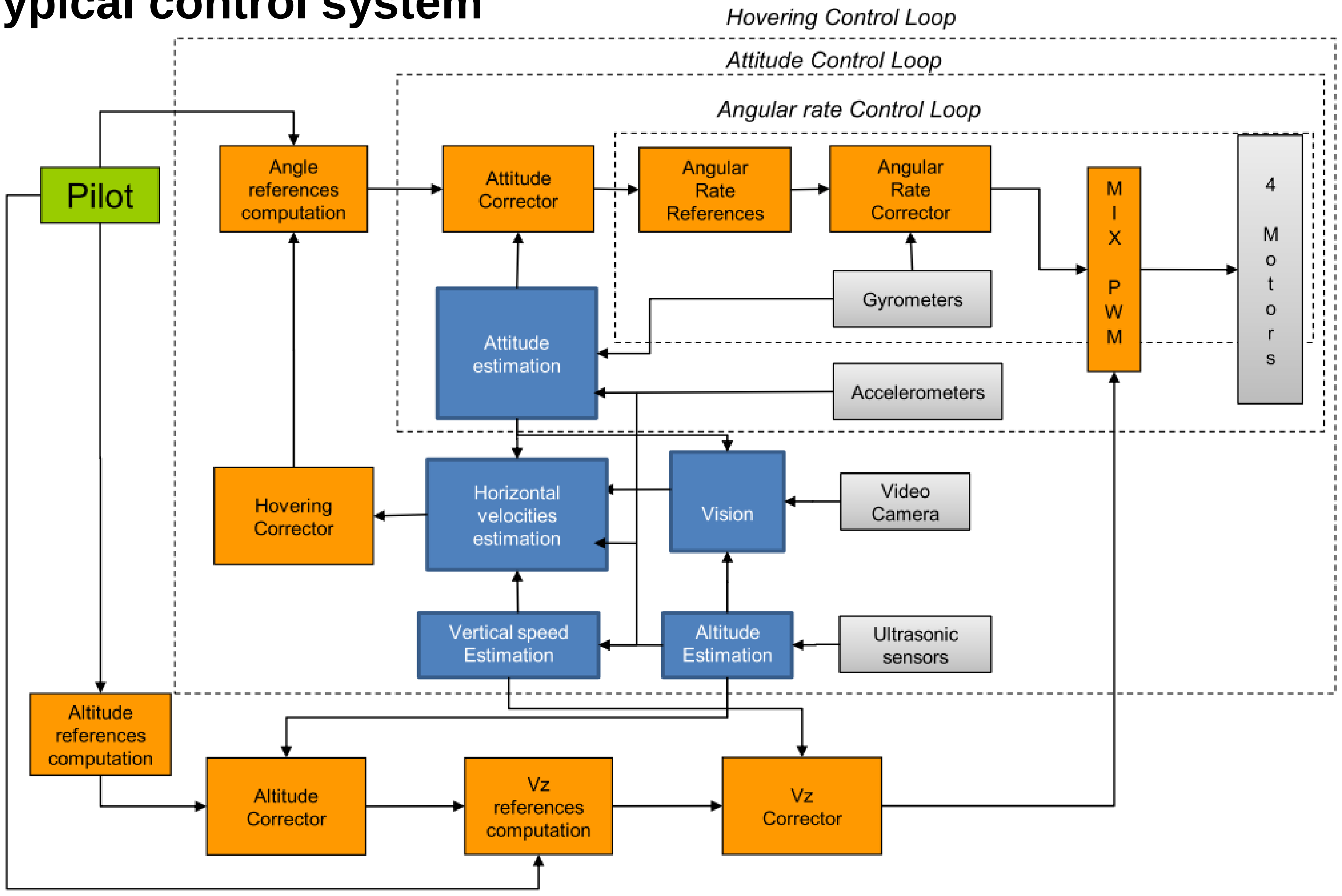
Step 1 Install bottom board → **Step 2** Install autopilot system → **Step 3** Install motors and ESCs → **Step 4** Install top board → **Step 5** Install propellers

Install screws by appropriate force to prevent breaking threads; use adequate screw glue for installing screws.

Please wire neatly. make sure wires will not be cut by frame boards and propellers. Smooth out the boards edge if necessary.

Please install propellers after the configuration procedure of autopilot system. You can use 8in propellers if there is no payloads. Make sure the rotation direction of propellers are the same as the figure shows.

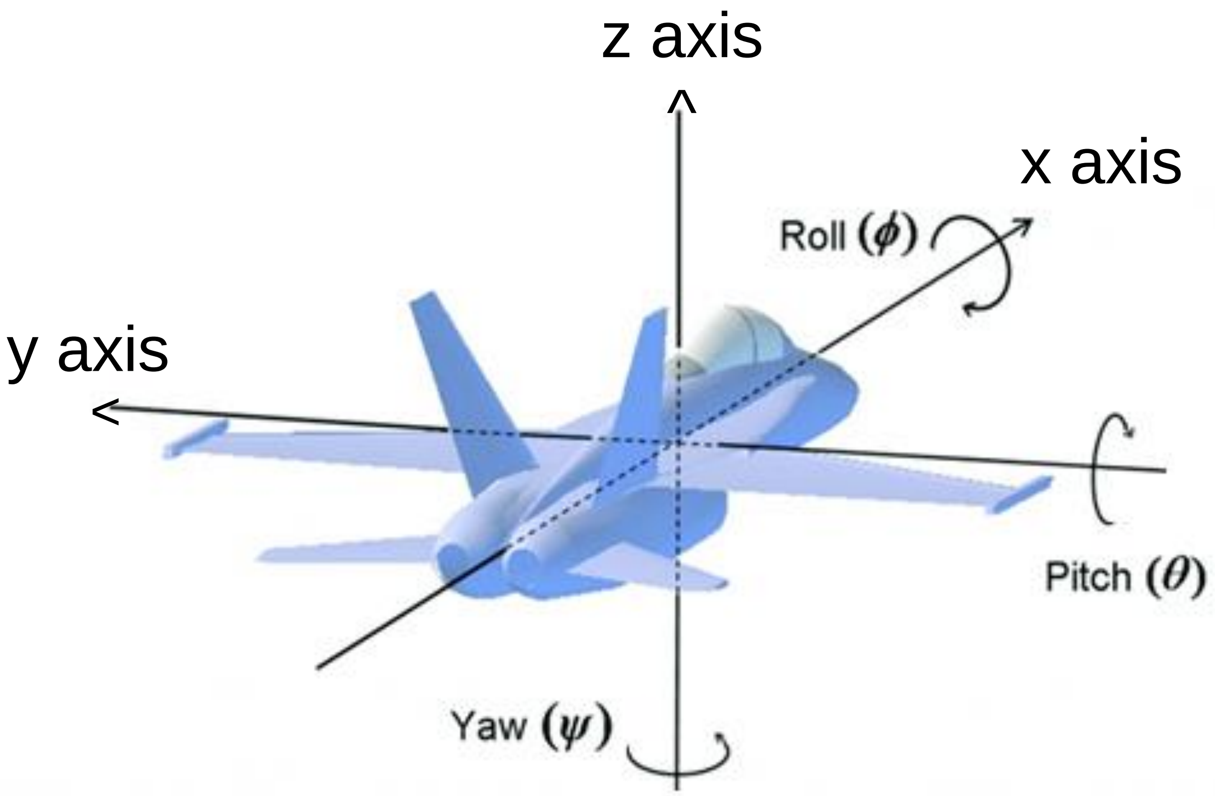
Typical control system



We'll implement the attitude control loop only

Attitude angles

Three angles must be controlled/stabilized: pitch, roll, yaw



Attitude control loop

- Probably, *quaternions* and/or *Euler angles* should be used to correctly describe and control the attitude
- however, in normal flight conditions, pitch and roll angles are small -> angles can be considered, and controlled, *independently*

- The system is nonlinear; nonlinear control theory should be applied
- for simplicity, we'll consider a PID controller instead

- By the way, thrust is *not* proportional to propeller speed…

Attitude control loop

- As a *very very very first* approximation, for each angle α we can write

$I\, d^2\alpha/dt^2 = T$

with I moment of inertia and T torque

- If proportional only action is used

$T = - k(\alpha - \alpha_o)$

we get

$I\, d^2(\alpha - \alpha_o)/dt^2 = - k(\alpha - \alpha_o)$

-> oscillations! A derivative component is needed: PD action

Attitude control and motors actuation

front left FL CW	front right FR CCW	white arms
rear left RL CCW	rear right RR CW	red arms

Pitch: (RL+RR)-(FL+FR)

Roll: (FL+RL)-(FR+RR)

Yaw: (FL+RR)-(FR+RL)

Lift: (FL+FR+RL+RL)

so that

$out_{FL} = -P+R+Y+L$

$out_{FR} = -P-R-Y+L$

$out_{RL} = +P+R-Y+L$

$out_{RR} = +P-R+Y+L$

Attitude estimation with MPU-9150

- MPU-9150: 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer
- Pitch: data fusion of
  - angle given by the accelerometer on the xz plane
  - integration of angular velocity on the y axis
- Roll: data fusion of
  - angle given by the accelerometer on the yz plane
  - integration of angular velocity on the x axis
- Yaw: data fusion of
  - angle given by the compass on the xy plane
    - be careful to possible +/-360 deg discontinuities!
  - integration of the angular velocity on the z axis

Attitude estimation: data fusion

- As
  - gyro angular velocities must be integrated -> drift
  - accelerometer values are noisy
  - compass values are noisy

a Kalman filter should be used

- a simpler solution is with a *complementary filter:* for any of the 3 angles,

$$\alpha=(1-\varepsilon)\ \alpha_{gyro}+\varepsilon\ \alpha_{accelerometer/compass}$$

with ε around 0.01

Power electronics

- Brushless motors are powered by ESCs (Electronic speed control) circuits
  - each ESC actually implements the first level control loop (angular rate control loop) for its motor
  - PWM, 50 Hz, 5-10% duty cycle
- power is provided by a 3S lithium battery (11.1 V, around 2000 mAh)

MCU s/w

- Inputs:
  - commands from remote control (RC)
    - PWM, 50 Hz, 5-10% duty cycle
    - possibly using io,rise(), t.start(), io.fall(), t.read(), t.stop()
  - a couple of switches to set the control parameters
- outputs: commands to the 4 ESCs
- sampling frequency: at least 100 Hz
- control law: at least PD
- variables can probably all be treated in floating point: FRDM-KL25Z should be powerful enough

S/W (bare metal)

Init phase

- read MPU9150 to correct offsets
- init the ESCs and slowly start motors

Then for each sampling period

- read new 4 values from the RC (if any)
- read MPU9150 and update 3 attitude angles estimates
- compute 3 controller outputs for 3 angles
- add lift component to controller outputs and send them to ESCs

S/W development and project documentation

For any group project

- a repository (sych as github or gitlab) should be used
- we’ll simply use a shared folder on google drive

- documentation is fundamental!

Regulations

- Always wear eye protection glasses!!

- Outdoor flights require an A1+A3 driving licence (https://learningzone.eurocontrol.int/ilp/pages/catalogsearch.jsf?q=%7B!q.op%3DAND%7D%20*a1*%20*a3*&sidebarExpanded=true&rows=1

- … and the registration as drone driver (few euros) on the web site

https://www.d-flight.it/new_portal/

Possible further developments

- Better modelling
- Simulation of the drone dynamics, e.g. in Matlab
  - to tune the various parameters
  - for pilot training
  - *but accurate modelling is necessary*, and this is not trivial
- Memorization of control parameters on EEPROM
- Altitude (wrt ground) control using ultrasound distance meter
  - and automatic take-off and landing
- Bi-directional communication for telemetry (e.g. on battery charge status)
  - bidirectional radio connection is needed
  - e.g., NiceRF SV610