Class UAV

Class used to represent an UAV.

Parameters:

- **drone_ns** : str
  ROS namespace where the data from the PX4 autopilot and the MOCAP system is encapsulated.

- **mass** : str
  Mass of the drone.

- **radius** : str
  Radius of the vehicle.

- **height** : str
  Height of the drone.

- **num_rotors** : str
  Number of rotors of the drone.

- **thrust_curve** : str
  Thrust curve of the vehicle.

Attributes:

- **ekf.pos** : np.array of floats with shape (3,1)
  Position of the vehicle, in local NED coordinates, provided by the extended Kalman filter of the PX4 autopilot.

- **ekf.vel** : np.array of floats with shape (3,1)
  Linear velocity of the drone, in local NED coordinates, provided by the extended Kalman filter of the PX4 autopilot.

- **ekf.vel_body** : np.array of floats with shape (3,1)
  Linear velocity of the vehicle, in body NED coordinates, provided by the extended Kalman filter of the PX4 autopilot.

- **ekf.att_q** : np.array of floats with shape (4,1)
  Attitude of the vehicle, expressed in quaternions, provided by the extended Kalman filter of the PX4 autopilot.

- **ekf.att_euler** : np.array of floats with shape (3,1)
  Attitude of the drone, expressed in Euler angles, provided by the extended Kalman filter of the PX4 autopilot.

- **ekf.ang_vel** : np.array of floats with shape (3,1)
  Angular velocity of the vehicle provided by the extended Kalman filter of the PX4 autopilot.

- **sen.imu.acc_body** : np.array of floats with shape (3,1)
  Linear acceleration of the vehicle, in body NED coordinates, measured by the IMU.

- **sen.imu.ang_vel** : np.array of floats with shape (3,1)
  Angular velocity of the drone measured by the IMU.

- **sen.imu.mag** : np.array of floats with shape (3,1)
  Magnetic field vector, in body NED coordinates, measured by the IMU. Expressed in Teslas.

- **sen.mocap.pos** : np.array of floats with shape (3,1)
  Position of the vehicle, in local NED coordinates, provided by the motion capture system.

- **sen.mocap.att_q** : np.array of floats with shape (4,1)
  Attitude of the vehicle, expressed in quaternions, provided by the motion capture system.

- **sen.mocap.att_euler** : np.array of floats with shape (3,1)
  Attitude of the drone, expressed in Euler angles, provided by the motion capture system.

- **sen.gps.pos** : np.array of floats with shape (3,1)
  Position of the vehicle, in gps coordinates, provided by the GPS sensor.

- **sen.baro.pressure** : float
  Static pressure measured by the barometer.

- **sen.baro.temperature** : float
  Temperature, in degrees Kelvin, measured by the thermometer integrated in the barometer.

- **sen.baro.alt** : float
  Altitude of the vehicle, above mean sea level, computed through the barometric atmospheric pressure and the temperature.

- **sen.emu.rel_pos** : np.array of floats with shape (n,3,1)
  Relative position of each of the n neighbour vehicles, in local NED coordinates, provided by the emulated relative position sensor.

- **sen.emu.rel_vel** : np.array of floats with shape (n,3,1)
  Relative velocity of each of the n neighbour vehicles, in local NED coordinates, provided by the emulated relative velocity sensor.

- **act.group** : int
  States the group of the active motors and servos of the drone.

- **act.output** : list
  Stores the normalized values (0 to 1 or -1 to 1) applied to the mixer and/or motors and servos of the vehicle.

- **info.drone_ns** : str
  ROS namespace where the data from the PX4 autopilot and the MOCAP system is encapsulated.

- **info.mass** : float
  Mass of the drone.

- **info.radius** : float
  Radius of the drone.

- **info.height** : float
  Height of the drone.

- **info.num_rotors** : int
  Number of rotors of the drone.

- **info.thrust_curve** : str
  Thrust curve of the drone.

- **info.flight_mode** : str
  Current flight mode of the drone.

- **info.is_connected** : bool
  States if the system is connected to the PX4 autopilot.

- **info.is_armed** : bool
  Stores the armed state of the vehicle. If True, the drone is armed.

- **info.is_landed** : bool
  Stores the landed state of the vehicle. If True, the drone is landed.

- **info.battery** : float
  Remaining battery percentage.

Methods:

- **arm_drone()**
  Arms the drone, if it is not already armed.

- **start_offboard_mode()**
  Changes the flight mode of the PX4 autopilot of the drone to offboard.

- **start_offboard_mission()**
  Makes the vehicle ready for an offboard experiment by arming it and by changing the flight mode of its PX4 autopilot to offboard.

- **set_pos_yaw(pos, yaw, time)**
  Offboard method that sends position and yaw references to the PX4 autopilot of the the drone.

  Parameters:
  - pos : np.array of floats with shape (3,1)
    Desired position for the drone, in local NED coordinates.
  - yaw : float
    Desired yaw for the vehicle, in radians.
  - time: float
    Time, in seconds, during which the selected position and yaw references will be sent to the PX4 autopilot.

- **set_vel_yaw(vel, yaw, freq)**
  Offboard method that sends velocity and yaw references to the PX4 autopilot of the the vehicle.

  Parameters:
  - vel : np.array of floats with shape (3,1)
    Desired linear velocity for drone, in local NED coordinates.
  - yaw : float
    Desired yaw for the vehicle, in radians.
  - freq : float
    Topic publishing frequency, in Hz.

- **set_vel_body_yaw_rate(vel_body, yaw_rate, freq):**
  Offboard method that sends velocity_body and yaw_rate references to the PX4 autopilot of the the drone.

  Parameters:
  - vel_body : np.array of floats with shape (3,1)
    Desired linear velocity for the drone, in body NED coordinates.
  - yaw_rate : float
    Desired yaw rate for the vehicle, in radians per second.
  - freq : float
    Topic publishing frequency, in Hz.

- **set_att_thrust(att, att_type, thrust, freq)**
  Offboard method that sends attitude and thrust references to the PX4 autopilot of the the vehicle.

  Parameters:
  - att : np.array of floats with shape (3,1) or with shape (4,1)
    Desired attitude for the vehicle, expressed in euler angles or in a quaternion.
  - att_type : str
    Must be equal to either 'euler' or 'quaternion'. Specifies the format of the desired attitude.
  - thrust : float
    Desired thrust value in newtons.
  - freq : float
    Topic publishing frequency, in Hz.

- **set_ang_vel_thrust(ang_vel, thrust, freq):**
  Offboard method that sends angular velocity and thrust references to the PX4 autopilot of the the drone.

  Parameters:
  - ang_vel : np.array of floats with shape (3,1)
    Desired angular velocity for the drone.
  - thrust : float
    Desired thrust value in newtons.
  - freq : float
    Topic publishing frequency, in Hz.

- **set_act(self, group, output, freq)**
  Offboard method that sets the values of the mixers and/or actuators of the vehicle.

  Parameters:
  - group: int
    Desired control group.

- output : list
  Desired output values for the mixers and/or actuators of the drone.
- freq : float
  Topic publishing frequency, in Hz.

- **disarm_drone()**
  Disarms the vehicle, if it is not already disarmed.

- **auto_land()**
  Lands the drone, changing its flight mode to auto-land.

- **init_telemetry()**
  Manages topic subscriptions.

- **update_position(msg)**
  Updates the variable that stores the position of the drone, in local NED coordinates, provided by the extended Kalman filter of the PX4 autopilot.

- **update_velocity(msg)**
  Updates the variable that stores the linear velocity of the vehicle, in local NED coordinates, provided by the extended Kalman filter of the PX4 autopilot.

- **update_velocity_body(msg)**
  Updates the variable that stores the linear velocity of the vehicle, in body NED coordinates, provided by the extended Kalman filter of the PX4 autopilot.

- **update_attitude(msg)**
  Updates the variables that store the attitude of the vehicle provided by the extended Kalman filter of the PX4 autopilot.

- **update_angular_velocity(msg)**
  Updates the variable that stores the angular velocity of the vehicle provided by the extended Kalman filter of the PX4 autopilot.

- **update_imu(msg)**
  Updates the variables that store the linear acceleration and the angular velocity of the drone measured by the IMU.

- **update_mag(msg)**
  Updates the variable that stores the magnetic field vector, in body NED coordinates, measured by the IMU.

- **update_mocap(msg)**
  Updates the variables that store the position and attitude of the drone provided by the motion capture system.

- **update_gps(msg)**
  Updates the variable that stores the raw measurements provided by the GPS sensor.

- **update_baro_pressure(msg)**
  Updates the variable that stores the static pressure measured by the barometer.

- **update_baro_temperature(msg)**
  Updates the variable that stores the temperature, in degrees Kelvin, measured by the thermometer integrated in the barometer.

- **update_baro_altitude(msg)**
  Updates the variable that stores the altitude of the vehicle, above mean sea level, computed through the barometric atmospheric pressure and the temperature.

- **update_relative_positions(msg)**
  Updates the variable that stores the relative position of each of the n neighbour vehicles, in local NED coordinates, provided by the emulated relative position sensor.

- **update_relative_velocities(msg)**
  Updates the variable that stores the relative velocity of each of the n neighbour vehicles, in local NED coordinates, provided by the emulated relative velocity sensor.

- **update_actuator(msg)**
  Updates the variables that store the group and the current normalized values (0 to 1 or -1 to 1) applied to the mixer and/or motors and servos of the vehicle.

- **update_status(msg)**
  Updates the variables that store the current flight mode of the PX4 autopilot, the system status, and the armed state of the vehicle.

- **update_landed(msg)**
  Updates the variable that stores the landed state of the drone.

- **update_battery(msg)**
  Updates the variable that stores the remaining battery percentage.