PyCopter's technical note and final assignments

Guidance Navigation and Control course

I. Introduction

This note explains the theory behind the PyCopter simulator[1] so that the students can implement and test the performance of a GNC algorithm for a team of rotorcraft. The Section II describes the employed model for the rotorcraft in the simulator. The Section III introduces the controllers available in the simulator so that they can be employed to command the rotorcraft. The last Section IV suggests some topics for the final assignment. The group of students must choose one topic among the list or propose a new one to the teacher.

II. The quadcopter

A quadcopter is a multirotor helicopter that is lifted and propelled by four rotors. Quadcopters are classified as rotorcraft, as opposed to fixed-wing aircraft, because their lift is generated by a set of propellers spinned by rotors. We consider that the quadcopter has a two pairs of identical fixed-pitched propellers: two clockwise and two counter-clockwise. By changing the angular speed of each rotor it is possible to specifically generate torques and thrust along the vertical body axis to induce motion in the 3D space. We will consider a typical quadcopter of one kilogram of mass and a diameter of about one meter similarly as the one in Figure 1.

The first step to control a vehicle is to model it. Since we are interested in controlling a formation of quadcopters with respect to Earth, we need first to define the corresponding frames of coordinates and their relations. Secondly, we will set a *good enough* model for the considered quadcopter. A good model in this context means that we can omit some aerodynamics effects while having *decent* accuracy on the simulation of the demanded vehicle's maneuvers.

A. Frames of coordinates

We set the origin of the Earth-centered Earth-fixed frame O_E at the center of the planet, the Z_E axis is aligned with the rotational axis of the planet and the X_E axis is pointing to the zero-longitude meridian. Therefore, O_E is rotating together with the planet. We define the Navigation frame O_N to the axes defining a tangent plane on the Earth's surface at the geodetic longitude l and latitude ϕ_g coordinates with respect to O_E. The axis X_N and Y_N points to the geographic North and East respectively. We define the Quad frame O_Q with the axis Z_Q pointing down. The attitude or alignment of the O_Q with respect to O_N is given by the following sequence of rotations:

1) Right-handed rotation about the Z_N, positive *yaw* angle ψ.
2) Right-handed rotation about the Y_N, positive *pitch* angle θ.
3) Right-handed rotation about the X_N, positive *roll* angle ϕ.

Therefore, we can consider the following rotational matrix relating both navigation and quadcopter frames

$$ {}^Q_N R = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ -c\phi s\phi + s\phi s\theta c\psi & c\phi c\psi + s\phi s\theta s\psi & s\phi c\theta \\ s\phi s\psi + c\phi s\theta c\psi & -s\phi c\phi + c\phi s\theta s\psi & c\phi c\theta \end{bmatrix}, \tag{1} $$

where we have denoted by s and c the sine and the cosine functions respectively.

In this section we consider a fixed O_N close to the team of quadcopters. In fact since the Earth's radius is about 6371 km, then for our purposes we can consider that the Earth is flat about a kilometer around of O_N. We consider that each

[1] https://github.com/noether/pycopter

Fig. 1: Flying quadcopter BeBop from Parrot company.

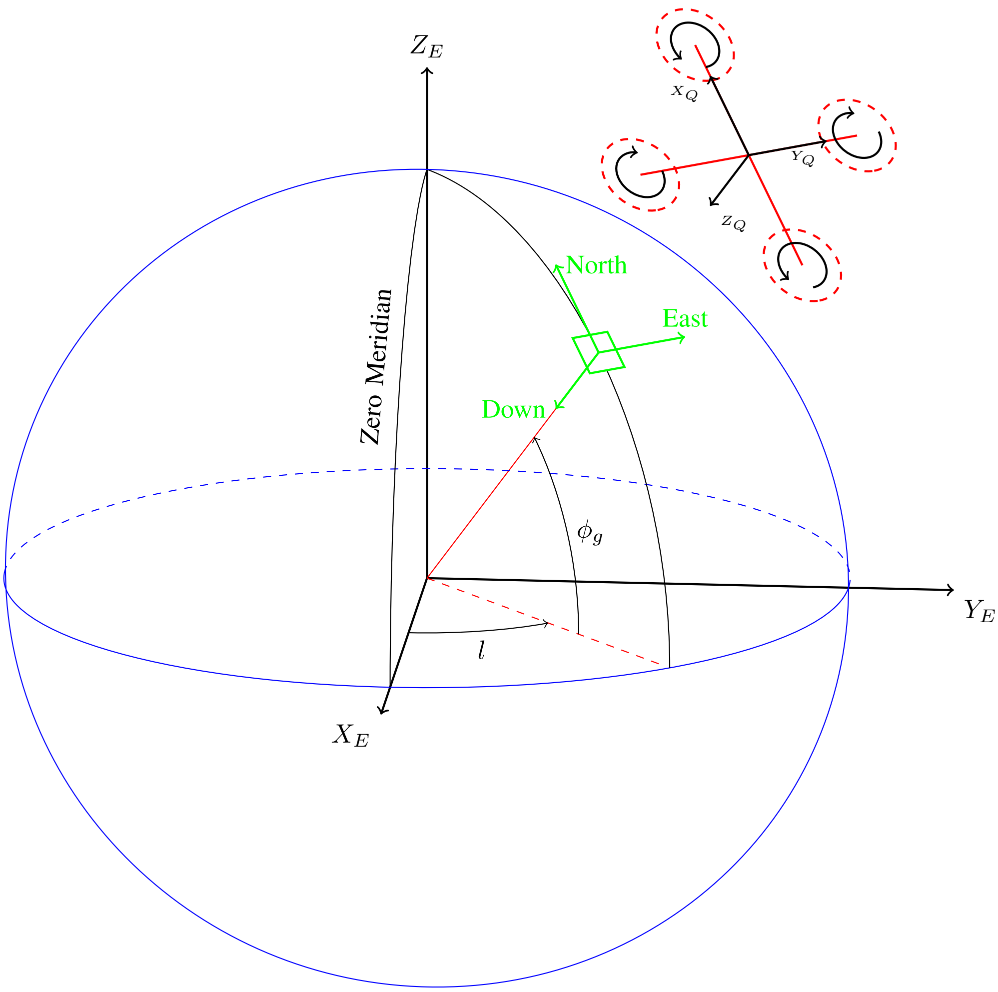

Fig. 2: The frame of coordinates ECEF (origin O_E) is at the center of the Earth and it rotates together with the planet. The Navigation coordinates in green color (origin O_N) is defined by the axes forming a tangent plane to the Earth's surface at the geodetic longitude l and latitude ϕ_g. The axes X_N and Y_N point to the geographic North and East respectively. The alignment of the body frame (origin O_Q) with respect to the Navigation frame is given by the three attitude angles ϕ, θ and ψ.

quadcopter is able to measure via a localization device, its altitude h with respect to the Earth's surface and the geodetic longitude l and latitude ϕ_g. The transformation coordinates between the geodetic system and O_E is given by

$$^{E}p = \begin{bmatrix} (N+h)\cos\phi_g \cos l \\ (N+h)\cos\phi_g \sin l \\ \big(N(1-e^2)+h\big)\sin\phi_g \end{bmatrix}, \tag{2}$$

where we have used the following constants defined by the World Geodetic System (WSG)-84 model [1]

$$\begin{aligned} a &\triangleq 6,378,137.0m \\ b &\triangleq 6,356,752m \\ e &= \frac{\sqrt{a^2+b^2}}{a} \\ N &= \frac{a}{\sqrt{1-e^2\sin^2\phi_g}}, \end{aligned}$$

therefore, the Cartesian coordinates of the quad with respect to O_N can be computed as

$$^{N}p_{\text{Quad}} = {}^{E}_{N}R({}^{E}p_{\text{Quad}} - {}^{E}p_{\text{NAV}}), \tag{3}$$

where the rotational matrix ${}^E_N R$ aligns the O_N frame to the O_E frame and is given by

$$ {}^E_N R = \begin{bmatrix} -\sin\phi_g^* \cos l^* & -\sin\phi_g^* \sin l^* & \cos\phi_g^* \\ -\sin l^* & \cos l^* & 0 \\ -\cos\phi_g^* \cos l^* & -\cos\phi_g^* \sin l^* & -\sin\phi_g^* \end{bmatrix}, $$

where l^* and ϕ_g^* are the geodetic longitude and latitude respectively of the fixed O_N. **The position of a rotorcraft in the PyCopter simulator is** ${}^N p_{\textbf{Quad}}$ in the North-East-Down (NED) convention. Take a look at the *physical variables* section in the quarrotor class in the quadrotor.py file.

B. *Physical model*

In this section we proceed to obtain a physical model of the quadcopter. In particular, we will set a space-state model where the control inputs are the reference signals for the angular speed of the four propellers, and the states are the position ${}^N p_{\text{Quad}}$, the velocity ${}^N \dot{p}_{\text{Quad}}$, the attitude angles ψ, θ and ϕ and their corresponding velocities.

Let us start with the following kinematic relation

$$ \begin{bmatrix} \dot\phi \\ \dot\theta \\ \dot\psi \end{bmatrix} = \begin{bmatrix} 1 & t\theta s\phi & t\theta c\phi \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix}, \tag{4} $$

where t is the tangent function and P, Q and R are the standard symbols for the three angular velocities measured at the quadcopter frame O_Q, for example, from an Inertial Measurement Unit (IMU). We will further make the following assumption

***Assumption** 2.1:* The plane formed by the axes X_Q and Y_Q is almost parallel to the one formed by the axes X_N and Y_N.

In other words, we assume that ϕ and θ are sufficiently small angles such that

$$ \begin{bmatrix} \dot\phi \\ \dot\theta \\ \dot\psi \end{bmatrix} \approx \begin{bmatrix} P \\ Q \\ R \end{bmatrix}. \tag{5} $$

We continue with writing the angular dynamics at the frame O_Q. It is well known that such dynamics correspond to the rigid body dynamics or Euler's equations

$$ J\dot\omega = M - (\omega \times (J\omega)), \tag{6} $$

where J is the inertia tensor of the quadcopter, $\omega = \begin{bmatrix} P & Q & R \end{bmatrix}^T$ and $M = \begin{bmatrix} l & m & n \end{bmatrix}^T$ is the applied moments by the propellers. Based on the symmetry of the quadcopter we can consider the following inertia tensor

$$ J_Q \approx \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}, \tag{7} $$

where $J_{zz} \gg J_{xx} = J_{yy}$, and under Assumption 2.1 we can further write

$$ \begin{cases} J_{xx}\ddot\phi & \approx (J_{yy} - J_{zz})\dot\theta\dot\psi + l \\ J_{yy}\ddot\theta & \approx (J_{zz} - J_{xx})\dot\phi\dot\psi + m \\ J_{zz}\ddot\psi & \approx n. \end{cases} \tag{8} $$

Let us now write the translational dynamics of the quadcopter at the body frame O_Q

$$ \begin{cases} {}^Q\ddot{p}_x & = {}^Q\dot{p}_y R - {}^Q\dot{p}_z Q - g\sin\theta + \frac{{}^Q X_A + {}^Q X_T}{m} \\ {}^Q\ddot{p}_y & = {}^Q\dot{p}_z P - {}^Q\dot{p}_x R + g\sin\phi\cos\theta + \frac{{}^Q Y_A + {}^Q Y_T}{m} \\ {}^Q\ddot{p}_z & = {}^Q\dot{p}_x Q - {}^Q\dot{p}_y P + g\cos\phi\cos\theta + \frac{{}^Q Z_A + {}^Q Z_T}{m}, \end{cases} \tag{9} $$

where m is the mass of the quadcopter, g is the acceleration due to gravity, X_A, Y_A, Z_A are the aerodynamic forces present in the quadcopter and X_T, Y_T, Z_T are the forces due to the propellers. In addition to Assumption 2.1, we also consider the following

***Assumption** 2.2:* There is almost not wind around O_N and the translational velocity ${}^Q\dot{p}$ is sufficiently small such that the aerodynamic forces are negligible.

Under Assumption 2.2 we have that X_A, Y_A, Z_A are zero. Since the only thrust is given by the propellers oriented along Z_Q, then, we can write that ${}^QX_T$ and ${}^QY_T$ are zero and ${}^QZ_T = -T$, i.e., the total thrust. In combination with Assumption 2.1 from (1) and (9) we have that

$$\begin{cases} {}^N\ddot{p}_z & \approx g - T/m \\ \phi & \approx \frac{m}{T}\left({}^N\ddot{p}_x \sin\psi - {}^N\ddot{p}_y \cos\psi\right) \\ \theta & \approx \frac{m}{T}\left({}^N\ddot{p}_x \cos\psi + {}^N\ddot{p}_y \sin\psi\right). \end{cases} \tag{10}$$

The approximations in (10) show that the vertical and horizontal motions of the quadcopter can be decoupled. Therefore, once the quadcopter is stabilized at a desired altitude ${}^Np_z^*$ with a thrust T^* overcoming the gravity g, in order to travel with respect to O_N with desired accelerations ${}^N\ddot{p}_x^*$ and ${}^N\ddot{p}_y^*$ we need to set the roll and pitch to ϕ^* and θ^* respectively derived from (10), whose dynamics are given by (8). In the next subsection we will show how to relate the force and moment inputs T, l, m and n in (10) and (8) with the angular speed of the propellers, which is the actual control inputs of our vehicle.

C. Motors and propellers models

The propellers of the quadcopter are attached to a spinning motor in order to induce a thrust force. In aerodynamics, the force originated from the propeller i can be modeled by

$$F_i = \frac{1}{2}\rho C V_i^2, \tag{11}$$

where ρ is the air density, V_i is the speed of the propeller into the airflow and C is an aerodynamic coefficient depending on many factors such as the Reynolds number, *angle of attack* or pitch of the propeller, size, etc. For more details in this matter we refer the reader to the excellent book [2]. Once a particular propeller has been chosen for the quadcopter, in the steady-state conditions we can consider all these factors constant (up to calibration) in (11), leading to

$$F_i = k_F \omega_i^2, \tag{12}$$

where ω_i is the angular speed of the propeller i. While the induce moments l and m are due mostly due to the force difference in the X_Q and Y_Q axes respectively, the induced moment n around the Z_Q axis is due to *yawing moment* originated by the spinning of the four propellers, similar to (11) we can consider that

$$n_i = k_M \omega_i^2, \tag{13}$$

where k_M is the corresponding yawing moment coefficient that is constant (up to calibration) when the quadcopter is at a steady-state condition.

We will consider *brushless* motors for spinning the propellers. These motors within their nominal state have a transitory time much faster than the dynamics given in (8) and (10) for our considered quadcopter. Therefore, we will assume that the setting point ω_i^* for the motor i will be the actual ω_i of the propeller, and the total thrust T and moments l, m and n at the frame O_Q can be calculated as

$$\begin{bmatrix} T \\ l \\ m \\ n \end{bmatrix} = \begin{bmatrix} -k_F & -k_F & -k_F & -k_F \\ 0 & -dk_F & 0 & dk_F \\ dk_F & 0 & -dk_F & 0 \\ -k_M & k_M & -k_M & k_M \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}, \tag{14}$$

where d is the distance from O_Q to a propeller and for the moment n we have consider the clockwise and counterclockwise rotations of the four propellers. As we will see in the following section, the controllers of the quadcopter will demand to generate a particular T, l, m and n signals. For such a task we will need the motors to satisfy

$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \begin{bmatrix} -k_F & -k_F & -k_F & -k_F \\ 0 & -dk_F & 0 & dk_F \\ dk_F & 0 & -dk_F & 0 \\ -k_M & k_M & -k_M & k_M \end{bmatrix}^{-1} \begin{bmatrix} T \\ l \\ m \\ n \end{bmatrix}, \tag{15}$$

where it can be checked that the requested inverse always exists.

III. Controllers

In this section we will present a series of algorithms to control the attitude, the altitude/2D position and the velocity ${}^N\dot{p}$ of the quadcopter. For the final assignment, these controllers enables us to implement GNC algorithms whose outputs are desired velocities, since the control action u in a single-integrator model

$$\dot{x} = u, \tag{16}$$

can be considered as a desired velocity signal to be tracked.

A. Attitude controller

In order to control the attitude we define first the following error signals

$$e_\phi \triangleq \phi - \phi^* \tag{17}$$
$$e_\theta \triangleq \theta - \theta^* \tag{18}$$
$$e_\psi \triangleq \psi - \psi^*, \tag{19}$$

where ϕ^*, θ^* and ψ^* are the desired attitude angles, for instance, generated by the system (10). We propose then the following controllers for the system (8)

$$l = -k_{p_\phi} e_\phi - (J_{yy} - J_{zz})\dot{\theta}\dot{\psi} - k_{d_\phi}\dot{\phi} \tag{20}$$
$$m = -k_{p_\theta} e_\theta - (J_{zz} - J_{xx})\dot{\phi}\dot{\psi} - k_{d_\theta}\dot{\theta} \tag{21}$$
$$n = -k_{p_\psi} e_\psi - k_{d_\psi}\dot{\psi}, \tag{22}$$

where $k_{\{p_\phi,p_\theta,p_\psi,d_\phi,d_\theta,d_\psi\}} \in \mathbb{R}^+$ are control gains. We then provide the corresponding stability result.

Theorem 3.1 *(**control_att** in PyCopter):* The origin of the attitude error signals e_ϕ, e_θ and e_ψ from the closed-loop system (8) with control laws (20)-(22) is globally asymptotically stable for any $k_{\{p_\phi,p_\theta,p_\psi,d_\phi,d_\theta,d_\psi\}} > 0$.

Proof: Consider the following Lyapunov function

$$V = \frac{k_{p_\phi}}{2}e_\phi^2 + \frac{k_{p_\theta}}{2}e_\theta^2 + \frac{k_{p_\psi}}{2}e_\psi^2 + \frac{J_{xx}}{2}\dot{\phi}^2 + \frac{J_{yy}}{2}\dot{\theta}^2 + \frac{J_{zz}}{2}\dot{\psi}^2, \tag{23}$$

whose time derivative satisfies by including (20)-(22)

$$\begin{aligned}\frac{\mathrm{d}V}{\mathrm{d}t} &= k_{p_\phi}e_\phi\dot{\phi} + k_{p_\theta}e_\theta\dot{\theta} + k_{p_\psi}e_\psi\dot{\psi} + J_{xx}\dot{\phi}\ddot{\phi} + J_{yy}\dot{\theta}\ddot{\theta} + J_{zz}\dot{\psi}\ddot{\psi} \\ &= k_{p_\phi}e_\phi\dot{\phi} + k_{p_\theta}e_\theta\dot{\theta} + k_{p_\psi}e_\psi\dot{\psi} \\ &+ \Big((J_{yy} - J_{zz})\dot{\theta}\dot{\psi} + l\Big)\dot{\phi} + \Big((J_{zz} - J_{xx})\dot{\phi}\dot{\psi} + m\Big)\dot{\theta} + n\dot{\psi} \\ &= -k_{d_\phi}\dot{\phi}^2 - k_{d_\theta}\dot{\theta}^2 - k_{d_\psi}\dot{\psi}^2,\end{aligned} \tag{24}$$

which is clearly non-increasing in the compact set

$$\mathcal{Q} \triangleq \Big\{e_\phi, e_\theta, e_\psi, \dot{\phi}, \dot{\theta}, \dot{\psi} : \frac{k_{p_\phi}}{2}e_\phi^2 + \frac{k_{p_\theta}}{2}e_\theta^2 + \frac{k_{p_\psi}}{2}e_\psi^2 + \frac{J_{xx}}{2}\dot{\phi}^2 + \frac{J_{yy}}{2}\dot{\theta}^2 + \frac{J_{zz}}{2}\dot{\psi}^2 \leq V(0)\Big\}, \tag{25}$$

and by invoking LaSalle's invariance principle, we can conclude the asymptotic stability of the signals $e_\phi, e_\theta, e_\psi, \dot{\phi}, \dot{\theta}$ and $\dot{\psi}$ to the origin. ■

Note that the Theorem 3.1 guarantees an upper bound given by (25) for the attitude angles, and this uppder bound can be shaped by the gains $k_{p_{\phi,\theta,\psi}}$ while the rate of convergence can be shaped by the gains $k_{d_{\phi,\theta,\psi}}$. In other words, it helps to keep Assumptions 2.1 and 2.2 satisfied.

B. Altitude controller

In order to control the altitude of the quadcopter, from (10) we have to deal with the gravity g. For now we will consider that g is unknown and we will relax Assumption 2.2 by including non-linear drag forces. Therefore we want to design a control input $u \in \mathbb{R}$ for the following system derived from (9)

$$\begin{cases}\dot{z} &= v_z \\ \dot{v}_z &= -C_D v_z|v_z| + g + u,\end{cases} \tag{26}$$

where z is the altitude with respect to O_N (recalling that negative means *up*), $C_D(t) \in \mathbb{R}^+$ is a positive state-dependent time-varying drag coefficient and the control input is related to the quadcopter by $u = \frac{T}{m}$. Let us define the following error signals

$$e_z = z - z^* \tag{27}$$
$$e_{\xi_g} = \xi_g - g, \tag{28}$$

where z^* is the desired altitude and $\xi_g \in R$ is the state of an estimator for compensating the gravity with dynamics

$$\dot{\xi}_g = u_{\xi_g}. \tag{29}$$

We propose the altitude controller with estimator

$$u = -\xi_g - k_p\, e_z - k_d\, v_z \tag{30}$$

$$u_{\xi_g} = v_z, \tag{31}$$

where $k_p, k_d \in \mathbb{R}^+$ are positive gains, and we obtain the following closed-loop system

$$\begin{cases} \dot{e}_z &= v_z \\ \dot{v}_z &= -C_D v_z |v_z| + g - \xi_g - k_p\, e_z - k_d\, v_z \\ \dot{e}_{\xi_g} &= v_z. \end{cases} \tag{32}$$

Theorem 3.2 *(**set_(a or v)_2D_alt** in PyCopter):* The origin of the closed-loop system (32) is globally asymptotically stable for any $k_p, k_d > 0$.

Proof: Consider the following candidate Lyapunov function

$$V = \frac{k_p}{2} e_z^2 + \frac{1}{2} v_z^2 + \frac{1}{2} e_{\xi_g}^2, \tag{33}$$

whose time derivative satisfies

$$\begin{aligned} \frac{\mathrm{d}V}{\mathrm{d}t} &= k_p\, e_z \dot{z} + v_z \dot{v}_z + e_{xi_g} \dot{\xi}_g \\ &= -C_D v_z^2 |v_z| + (k_p\, e_z + g - \xi_g - k_p\, e_z - k_d\, v_z) v_z + e_{\xi_g} v_z \\ &= -C_D v_z^2 |v_z| - k_d\, v_z^2, \end{aligned} \tag{34}$$

which is clearly non-increasing in the compact set

$$\mathcal{Q} \triangleq \Big\{ e_z, v_z, e_{\xi_g} : k_p\, e_z^2 + v_z^2 + e_{\xi_g}^2 \leq 2V(0) \Big\}, \tag{35}$$

and by invoking LaSalle's invariance principle we can conclude that the origin of the closed-loop system (32) is globally asymptotically stable. ■

We first note that k_d is related to the rate of convergence and that k_p is related with the maximum possible values of the signals e_z, v_z and e_{ξ_g}. The estimator ξ_g in Theorem 3.2 acts as an integral controller but with the advantage that its value converges to the actual gravity. Nowadays the value of gravity can be obtained from very accurate models [3] and therefore one can use a feedforward compensation in (30). In such a case, one can employ the estimator ξ_g for calibrating k_F in (12) since $u = \frac{T}{m}$ in (26) and the generation of T is in open-loop.

Remark 3.3: For steering the quadcopter to a 3D position with respect to O_N, for the X and Y coordinates we can still employ the same controller as in Theorem 3.2 but without needing an estimator since gravity only is present in the Z axis.

C. Velocity controller

Since we have already decoupled the horizontal and the vertical motion of the quadcopter in (10), we will consider to control the *North* and $East$ velocities in ${}^N\dot{p}$ of the quadcopter and again we introduce drag forces, namely

$$\dot{v} = -C_D v ||v|| + u_v, \tag{36}$$

where $v \in \mathbb{R}^2$ is the stacked vector of the North and East velocities in O_N, $C_D \in R^+$ is a positive constant drag coefficient and $u \in \mathbb{R}^2$ is the control input. Note that this control signal u_v will be employed for ${}^N\ddot{p}_x$ and ${}^N\ddot{p}_y$ in (10). Let us define the following error signals

$$e_v = v - v^* \tag{37}$$

$$e_{\xi_{C_D}} = \xi_{C_D} - C_D, \tag{38}$$

where $v^* \in \mathbb{R}^2$ is the desired velocity and $\xi_{C_D} \in \mathbb{R}$ is the state of an estimator for estimating the drag coefficient with dynamics

$$\dot{\xi}_{C_D} = u_{\xi_{C_D}}. \tag{39}$$

Consider the control law with estimator

$$u_v = \xi_{C_D} v ||v|| - k_v e_v \tag{40}$$

$$\dot{\xi}_{C_D} = -||v|| e_v^T v, \tag{41}$$

that leads to the following closed-loop system

$$\begin{cases} \dot{e}_v &= -C_D v ||v|| + \xi_{C_D} v ||v|| - k_v e_v \\ \dot{e}_{\xi_{C_D}} &= -||v|| e_v^T v. \end{cases} \tag{42}$$

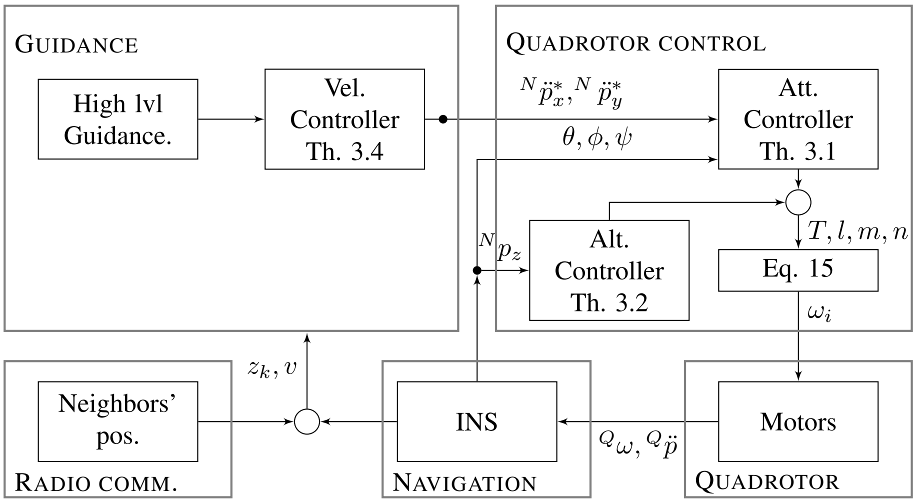

Fig. 3: Guidance, Navigation and Control system of the quadrotor. The Guidance block corresponds to the guidance system that generates the desired acceleration signals for the quadrotor in order to accomplish the assigned mission. The quadrotor control block corresponds to the control of the altitude and attitude of the vehicle to track the desired linear accelerations. The Inertial Navigation System (INS) estimates the actual state of the quadrotor which is necessary for the formation and quadrotor controls blocks. The relative positions between neighboring quadrotors could be obtained by communication from the neighbors' INS.

***Theorem** 3.4 (**set_v_2D_alt** in PyCopter):* The origin of the closed-loop system (42) is globally asymptotically stable for any $k_v > 0$.

Proof: Consider the following candidate Lyapunov function

$$V = \frac{1}{2}||e_v||^2 + \frac{1}{2}e_{\xi_{C_D}}^2, \tag{43}$$

whose time derivative satisfies

$$\begin{aligned}\frac{\mathrm{d}V}{\mathrm{d}t} &= e_v^T\dot{e}_v + e_{\xi_{C_D}}\dot{e}_{\xi_{C_D}} \\ &= e_{\xi_{C_D}}||v||e_v^T v - k_v||e_v||^2 - e_{\xi_{C_D}}||v||e_v^T v \\ &= -k_v||e_v||^2,\end{aligned} \tag{44}$$

which is clearly non-increasing in the compact set

$$\mathcal{Q} \triangleq \Big\{e_v, e_{\xi_{C_D}} : ||e_v||^2 + e_{\xi_{C_D}}^2 \leq 2V(0)\Big\}, \tag{45}$$

and by invoking LaSalle's invariance principle we can conclude that the origins of e_v and $e_{\xi_{C_D}}$ are globally asymptotically stable. ■

Remark 3.5: Although the Theorem 3.4 provides global stability for the velocity, we cannot forget that our whole system is based on the Assumption 2.2, i.e., sufficiently small linear speeds.

Remark 3.6: The controller in Theorem 3.4 can be extended to track a vertical velocity by incorporating the gravity g to (36). In such a case several options are available such as having only one estimator or two compensating drag and gravity separately.

Figure 3 shows the Guidance Navigation and Control (GNC) scheme for one quadrotor based on the presented results.

IV. ASSIGNMENTS

A. Coordinated motion of two rotorcraft with limited sensing

Consider two rotorcraft Q_1 and Q_2. The vehicle Q_1 knows its position $^Np_1$ but the vehicle Q_2 can only measure its altitude ^Np_z. Both vehicles can measure the distance $||^N(p_1 - p_2)||$ and their relative velocity $^N\dot{p}_1 - {^N\dot{p}_2}$. The team must be in formation by having the two vehicles with a relative position $^N(p_1 - p_2) = z$, where z is a vector with zero vertical component, i.e., $z = \begin{bmatrix} z_x & z_y & 0 \end{bmatrix}$. Please, design an algorithm such that the center of masses of the team tracks the following ellipse

$$f(p) = \left(\frac{x - x_o}{a}\right)^2 + \left(\frac{y - y_o}{b}\right)^2 - 1, \tag{46}$$

where $\begin{bmatrix} x_o & y_o \end{bmatrix}^T$ is the center of the ellipse and $a, b > 0$ are the length of its axes. The ellipse is located at a fixed altitude $c > 0$. An (overdetailed) report/reference on the guidance vector field explained in class [4].

B. Tracking and enclosing a target with limited sensing

Consider two rotorcraft Q_1 and Q_2, and a moving target Q_t at a constant altitude $t_z > 0$ with velocity $\dot{p}_t = \begin{bmatrix} v_x & v_y & 0 \end{bmatrix}$. Both vehicles Q_1 and Q_2 know the altitude of the target, and can measure their distance to the target, i.e., $||^N(p_{\{1,2\}} - p_t)||$, and the relative velocities ${}^N\dot{p}_{\{1,2\}} - {}^N\dot{p}_t$. Please, design an algorithm such that Q_1 and Q_2 describe a circumference around Q_t, and both Q_1 and Q_2 can control their intervehicle angle in the circumference. A reference on how to control the position of a team of vehicles on a circle [5].

C. Formation control for a team of rotorcraft with limited sensing

Consider an arbitrary number N of rotorcraft. Please, design a relative-position-based algorithm that takes as inputs: the desired deployment, the desired (parallel to the ground) velocity for the center of masses, and the desired altitude. Consider to include some rotorcraft that only can measure distances with respect to its neighbors, i.e., $||^N(p_i - p_j)||$, as well as the relative velocities ${}^N\dot{p}_i - {}^N\dot{p}_j$. Finally, consider the scenario where only one (leader) rotorcraft knows the desired velocity and wants the team to travel to different points in the space. An (overdetailed) report on formation control [6].

Appendix

Some advices for the assigment:

- The vehicles cannot handle arbitrarily high speeds. Watch out with yor commanded signals.
- To speed up simulations you might not want to play the animations but just the final plots from the logs.
- Try to understand first the given examples in PyCopter.
- Iterate your strategy to implement the algorithms and your final report with the teacher as much as you can.

References

[1] N. G.-I. Agency. (2012) World geodetic system. [Online]. Available: http://web.archive.org/web/20120402143802/https://www1.nga.mil/ProductsServices/GeodesyandGeophysics/WorldGeodeticSystem/Pages/default.aspx

[2] J. D. Anderson Jr, *Fundamentals of aerodynamics*. Tata McGraw-Hill Education, 1985.

[3] N. G.-I. Agency. (2008) Wsg84 earth gravitational model. [Online]. Available: http://earth-info.nga.mil/GandG/wgs84/gravitymod/

[4] H. G. de Marina, Y. A. Kapitanyuk, M. Bronz, G. Hattenberger, and M. Cao, "Guidance algorithm for smooth trajectory tracking of a fixed wing uav flying in wind flows," pp. 5740–5745, 2017.

[5] H. G. de Marina, Z. Sun, M. Bronz, and G. Hattenberger, "Circular formation control of fixed-wing uavs with constant speeds," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 5298–5303.

[6] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.