AIAA SciTech Forum
6-10 January 2020, Orlando, FL
AIAA Scitech 2020 Forum

10.2514/6.2020-2072

Longitudinal control of transition to powered flight for a parachute-dropped multirotor

Tomas I. Opazo * and Jack W. Langelaan†
The Pennsylvania State University, University Park, PA 16802, USA

The problem of controlling the safe transition of a multirotor vehicle between its release from a parachute to the moment it reaches steady state flight is analyzed. The problems of optimal trajectory, controller tuning, safety and stability to initial conditions are also studied. The proposed solution involves a two step process to find a safe controller. First, an optimal control problem is defined and then transcribed, using a direct collocation method, into a nonlinear programming problem. Second, the trajectory error for a subset of initial conditions is calculated, and the worst case is defined as the overall cost. This emerging Min-Max problem is then solved using particle swarm optimization to obtain the best set of controller gains. In order to evaluate performance, a cost function based on margins between the vehicle's state and their respective maximum allowable values was defined. Monte Carlo simulations were ran over the space of possible initial conditions. The combination of optimal trajectory and particle swarm derived controller gains results in an average state cost reduction of 23% when compared to a near-hover controller derived using Ziegler–Nichols method.

I. Introduction

There have been several examples of uninhabited fixed-wing aircraft released from balloons to enable high-altitude launch (see for example [1] and [2] and figure 1).

Fig. 1 Left: balloon-launched high altitude glider (from [1]); right: *Dragonfly* lander descending through Titan's atmosphere prior to the start of powered flight (image credit: Johns Hopkins APL).

The extension to parachute-based deployment of rotary-wing UAS has the potential to be operationally useful in several scenarios. For example, camera-equipped multirotors could be released from high altitude and glide descend under a parachute to the desired operations area without expending their battery. This would allow longer operational time at the destination. The current concept for Dragonfly (the NASA New Frontiers mission to Saturn's moon Titan) envisions traditional aeroshell and parachute-based entry and descent through Titan's upper atmosphere followed by release of the Dragonfly lander from the parachute and transition to powered flight and powered landing [3].

*Graduate Student, Department of Aerospace Engineering, Student Member AIAA
†Associate Professor, Department of Aerospace Engineering, Associate Fellow AIAA

The release of a rotorcraft from a parachute has the potential to be challenging. Besides mechanical considerations (e.g. preventing tangling of the shroud lines in rotors), aeromechanical challenges such as the avoidance of vortex ring state and the potential for high advance ratios must be addressed. At the same time, constraints on rotor speed, motor torque and power, and (since most multirotors are battery powered) battery current must be satisfied.

Here the transition to powered flight (TPF) is defined as the period between the vehicle's release from the parachute and its attaining a steady state (i.e. constant velocity, zero angular rate) flight condition. The TPF problem is cast as a trajectory optimization from the release condition to the final steady state condition. This optimal trajectory is then passed to the closed-loop flight control system that seeks to follow the trajectory.

Note that while the initial condition will be known at the moment of release, it may not be known with precision at the time of trajectory planning: depending on the planning method and the degree of complexity required of the system models in the planner, it may not be possible to plan in real time. This paper determines an optimal TPF from a nominal release condition and then uses the flight control system to track that trajectory from the actual release condition. Thus the controller gains are a part of the trajectory planning process: in addition to the trajectory, this paper finds a set of control gains that maximize safety over the space of possible release conditions. Design of control architectures is beyond the scope of this paper. Rather, this work assumes a system of nested PID loops is used (similar to that implemented on the PX4 flight stack in the Pixhawk autopilot).

To find the optimal trajectory and set of gains, a two step process is followed. First, a nonlinear optimal control problem (OCP) that calculates a nominal TPF trajectory is defined and solved using direct collocation. This nominal trajectory is assumed to begin from some nominal initial condition and end at the desired steady state flight condition. Second, using a set of off-nominal initial conditions (e.g. off-nominal initial pitch and pitch rate), particle swarm optimization is used to find the set of PID gains that minimizes the maximum deviation from the nominal trajectory. This combination of nominal trajectory and of PID gains is then tested in Monte Carlo simulations.

The remainder of this paper is organized as follows. Section II covers the problem description and transition to powered flight requirements. Section III details the setup of the vehicle and actuator model. Section IV explains the synthesis of two different PID controllers. Section V covers the safety evaluation and stability of each controller under different initial conditions. Conclusion are presented in Section VI.

II. Problem description

The problem under consideration is transition to a steady state powered flight condition from a steady state (unpowered) descent under a parachute. Note that in principle a negatively-buoyant balloon could also be used during the descent if a parachute cannot provide a low enough descent velocity.

Figure 2 shows a schematic of the problem. The rotorcraft releases from the parachute at an initial state $\mathbf{x}_0$. To ensure clearance from the parachute and associated support structure, a delay of t_d seconds is imposed before the rotors can begin to spin (so the rotorcraft is in free fall, acted upon by gravity and aerodynamic forces on the body). At this point the rotorcraft begins a dynamic maneuver to attain steady powered flight. The terminal condition is assumed to be a steady state (trimmed) flight condition (i.e. constant velocity). The focus is on determining a safe trajectory from release to steady state powered flight. While the motivating example is TPF for the Dragonfly lander, the method is general. Note also that vehicle parameters and constraints mentioned throughout this paper are notional, not specific to the Dragonfly lander.

In the case of an aeroshell deployment representative of that currently envisioned for Dragonfly, rotor blades are not free to spin until after release because of space constraints [4], [5]. Upon release, a safety delay of 0.5 s will produce an aeroshell-vehicle separation of approximately 0.16 m (1 s delay producing approx. 0.67 m separation under Titan's gravitational acceleration of 1.35 m/s^2). It is only at this point that motors can start spinning. The net effect is an increase in descent velocity of 0.6 m/s (1.3 m/s in the case of 1 s delay). The TPF must take this delay into account.

Vortex Ring State is a condition where the flow field surrounding the rotor is moving in the same direction of the produced thrust. A rotor descending vertically is an example if this. Turbulent Wake States is another condition that arises when the flow field velocity along the thrust direction is even higher. This condition is characterized by large recirculation and turbulence. The rotor acts similar to a circular plate in an up-moving airflow. From the point of view of the actuator (i.e. rotor), Vortex Ring State and Turbulent Wake States are undesired because the rotor lift over drag ratio worsens and fluctuates [6] [7]. This reduces control authority and makes the vehicle less stable. Potential problems with structural integrity of the blades due to vibration are also a concern [8].

After its release from the parachute and safety delay Dragonfly's downward velocity has the potential to put it well within Vortex Ring State, and actually crossing into Turbulent Wake State (according to definitions in [9]). Therefore, a

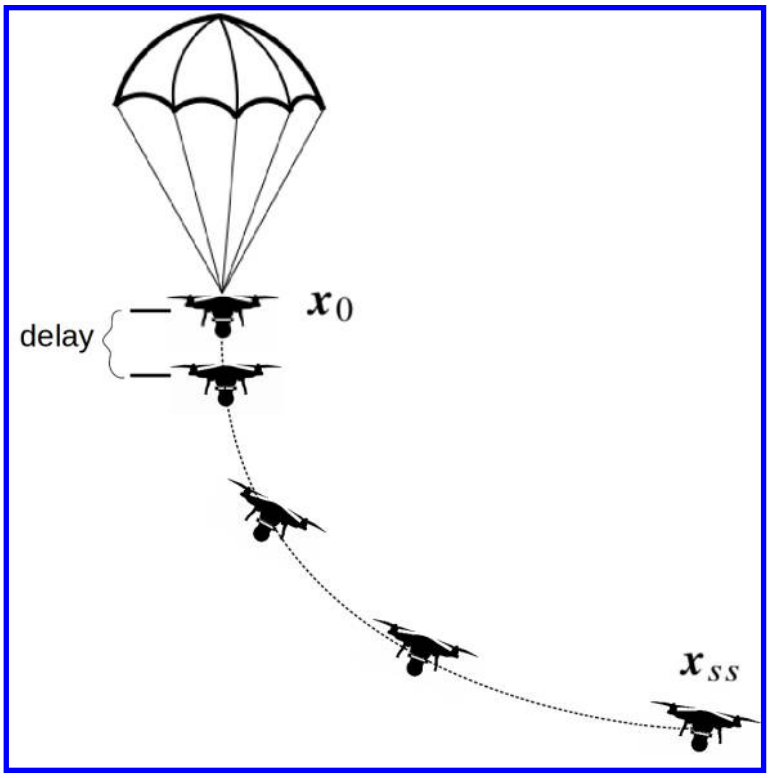

Fig. 2 Schematic of transition to powered flight trajectory

Table 1 Vehicle safety limits

Variable	Value	Units
Pitch	45	deg
Pitch rate	90	deg/s
Airspeed	12	m/s
Vertical acceleration	3	m/s^2

requirement for the TPF is to get the vehicle away from this condition by pitching the nose down (so that there is a significant component of edge-wise flow to the rotor and the wake is carried away) and then pulling out of the resulting dive.

Because gusts and uncertainties in atmospheric conditions can result in off-nominal conditions, TPF must be robust to uncertainties and variations in initial conditions. At the other end of the state trajectory the maneuver is considered complete whenever the target velocity (state) is achieved with a 1% error. And the target state is situated within the safe flight envelope of the vehicle.

Limits on pitch angle and pitch rate are defined to ensure envelope protection. Limits on motor current are also present. Table 1 shows the main vehicle limits.

The controller consists of a series of nested PID loops distributed in an outer loop controlling velocity, and an inner loop controlling angles, plus an allocation unit (mixer). Its input are the target velocity and vehicle state. Its output is a throttle command for each motor. This configuration is a common alternative covered in [10] and is also implemented in popular autopilots as PX4 (`http://px4.io/`). In the PX4 architecture and also in the architecture implemented in this paper, longitudinal and lateral motion (velocity and angles) are controlled using the same set of gains. Vertical motion (heave) is controlled separately. Note that Figure 3 shows control over roll, pitch, and yaw. Since the focus here is on longitudinal degrees of freedom, roll and yaw are set to zero.

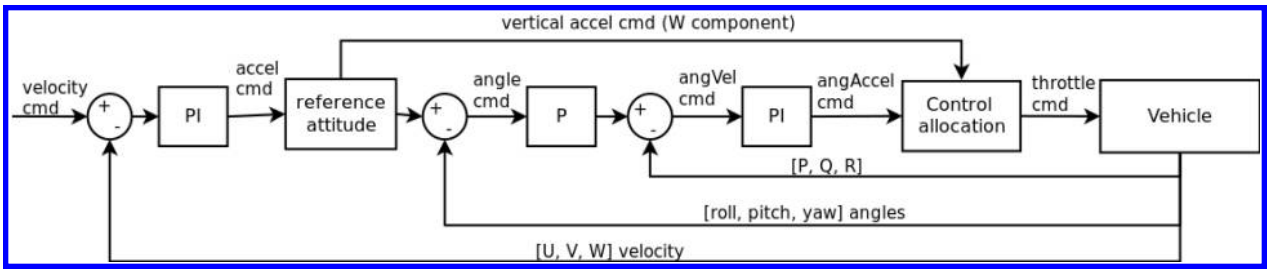

Fig. 3 PID architecture used in this work (PX4 autopilot)

III. Vehicle model

The vehicle's body is assumed to be rigid. Rotors are also assumed to be stiff (i.e. lead/lag and flap dynamics are assumed to be zero). An electromechanical model is implemented for the electronic speed control (ESC) and motors.

Referring to Figure 4, a vehicle with mass m_{body} and moment of inertia $\mathbf{J}_b$ is at a position $\mathbf{r}$ in an inertial north-east-down (NED) frame I. Orientation of the body frame B is defined by a quaternion $\mathbf{q} = [q_0\ q_1\ q_2\ q_3]^T$, where q_0 is the scalar and $[q_1\ q_2\ q_3]^T$ is the vector part. Actuators (i.e. motor/rotors) are located at positions $\mathbf{g}_i$ in the body frame. Each actuator is oriented along actuator frame R_i, defined with respect to the body frame, and produces thrust T_i. The rotation matrix from the i^{th} rotor frame to body frame is denoted by $\mathbf{R}_{i/B}$ and is assumed to be constant.

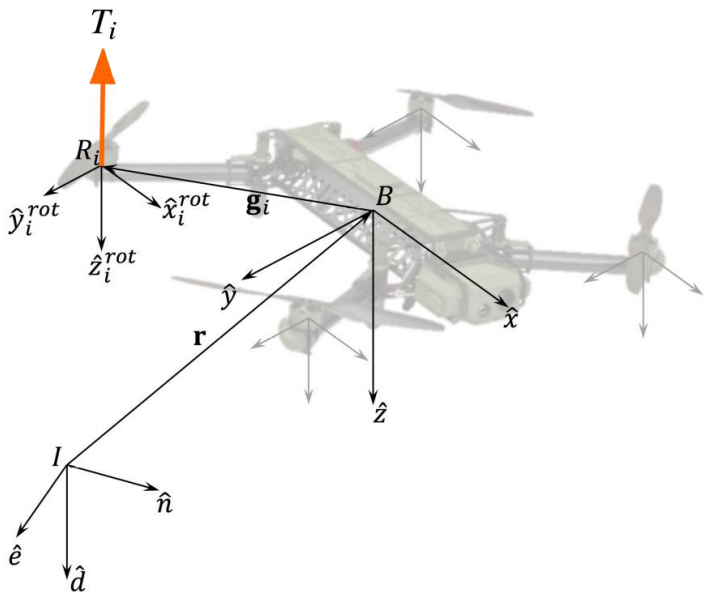

Fig. 4 Inertial frame, body fixed coordinates frame and Thrust T_i

The resulting equations of motion are

$$\dot{\mathbf{r}}^{ned} = \mathbf{v}^{ned} \tag{1}$$

$$\dot{\mathbf{q}}_{xyz/ned} = \frac{1}{2}\begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & Q & -P & 0 \end{bmatrix} \mathbf{q}_{xyz/ned} \tag{2}$$

$$\dot{\mathbf{v}}^{ned} = \frac{1}{m_{body}} \mathbf{F}_{net}^{ned} \tag{3}$$

$$\dot{\boldsymbol{\omega}}^{xyz} = \mathbf{J}_b^{-1}(\mathbf{M}_{net} - \boldsymbol{\omega}^{xyz} \times \mathbf{J}_b \boldsymbol{\omega}^{xyz} - \mathbf{M}_{rot}) \tag{4}$$

Where the state is defined as $\mathbf{x} = [\mathbf{r}^{ned}\ \mathbf{q}_{xyz/ned}\ \mathbf{v}^{ned}\ \boldsymbol{\omega}^{xyz}]^T$. When expressed in body frame, velocity is written as $\mathbf{v}^{xyz} = [U\ V\ W]^T$ and angular velocity as $\boldsymbol{\omega}^{xyz} = [P\ Q\ R]^T$.

The effect of the actuators on the vehicle's moment equation is $\mathbf{M}_{rot}$ and is discussed in following sections. The net moment acting on the vehicle is

$$\mathbf{M}_{net} = \mathbf{M}_{rotors} + \mathbf{M}_{aero} \tag{5}$$

Moment $\mathbf{M}_{aero}$ is taken from a lookup table as described in following sections. And $\mathbf{M}_{rotors}$ is the cross product of rotor location and rotor-produced thrust:

$$\mathbf{M}_{rotors} = \sum_{i=1}^{N} \mathbf{g}_i \times \mathbf{R}_{B/i} \begin{bmatrix} 0 \\ 0 \\ -T_i \end{bmatrix} \tag{6}$$

Where N is the number of rotors. Body aerodynamic forces and rotor forces are computed in body coordinates and then transformed into inertial coordinates:

$$\mathbf{F}^{ned} = \begin{bmatrix} ({q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2) & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\ 2(q_1q_2 - q_0q_3) & ({q_0}^2 - {q_1}^2 + {q_2}^2 - {q_3}^2) & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & ({q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2) \end{bmatrix}^{-1} \mathbf{F}^{xyz} \tag{7}$$

In this work, a quadcopter configuration as shown in Figure 4 is assumed. Rotor thrust vector is defined as $\mathbf{T}_{rotor} = [T_1\ T_2\ T_3\ T_4]^T$. The direction of thrust with respect to the vehicle is opposite to $\hat{z}_i^{rot}$ and is also shown in figure 4. The net force acting on the vehicle is

$$\mathbf{F}_{net} = \mathbf{F}_{rotors} + \mathbf{F}_{aero} + \mathbf{F}_{gravity} \tag{8}$$

The net force of rotors in the vehicle is

$$\mathbf{F}_{rotors}^{xyz} = \sum_{i=1}^{N} \mathbf{R}_{B/i} \begin{bmatrix} 0 \\ 0 \\ -T_i \end{bmatrix} \tag{9}$$

Gravity force is $\mathbf{F}_{gravity}^{ned} = [0\ \ 0\ \ g]^T$ and $\mathbf{F}_{aero}$ is taken from a lookup table as described in following sections.

A. Body aerodynamics

In the work presented here, body aerodynamic forces and moments were determined using CFD, with body drag coefficient, lift coefficient, and pitching moment coefficient defined as functions of body angle of attack and accessed via lookup table. Since the focus here is on longitudinal motion, side force coefficient, rolling moment coefficient, and yaw moment coefficient are all set to zero. Hence, the dependence of drag, lift, and pitch moment coefficients on sideslip angle is neglected.

Similarly, CFD was used to calculate rotor forces and torques as functions of rotor speed, rotor relative airspeed, and rotor angle of attack. Net rotor aerodynamic torque on the vehicle is not considered, since this work is concern with longitudinal motion only and thus cancellation of torque between rotors is assumed. Note that the airspeed and angle of attack of the i^{th} rotor is

$$\mathbf{v}_i^{xyz} = \mathbf{v}^{xyz} + \boldsymbol{\omega}^{xyz} \times \mathbf{g}_i^{xyz} \tag{10}$$

$$\mathbf{v}_i^{xyz_i} = [U_i\ \ V_i\ \ W_i]^T = R_{xyz_i/xyz}\mathbf{v}_i^{xyz} \tag{11}$$

$$\alpha_i = \text{atan2}(W_i, U_i) \tag{12}$$

The expressions for body aerodynamic forces and moments are

$$F_{drag} = q\ S\ C_{drag} \tag{13}$$

$$F_{lift} = q\ S\ C_{lift} \tag{14}$$

$$M_{pitch} = q\ S\ b\ C_{Mpitch} \tag{15}$$

$$\mathbf{R}_{wind/xyx} = \begin{bmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & 1 \end{bmatrix} \tag{16}$$

$$\mathbf{F}_{aero} = \mathbf{R}_{xyz/wind} \begin{bmatrix} -F_{drag} \\ 0 \\ -F_{lift} \end{bmatrix} \tag{17}$$

$$\mathbf{M}_{aero} = \mathbf{R}_{xyz/wind} \begin{bmatrix} 0 \\ M_{pitch} \\ 0 \end{bmatrix} \tag{18}$$

Where airspeed is assumed to be $V_a = \|\mathbf{v}^{ned}\|$, reference area S, and reference span b and dynamic pressure $q = \frac{1}{2}\rho {V_a}^2$. Equation 16 is a special case of the general rotation matrix $\mathbf{R}_{wind/xyz}(\alpha, \beta)$ between body and wind coordinates frames when the side slip angle is set to zero.

B. Actuator dynamics

In most multirotors and also in this work, the actuators are rotors (propellers) spun by motors. This electromechanical system is driven by an Electronic Speed Controllers (ESC) that keeps the RPM (angular velocity) at the desired level. The input variable to the ESC is throttle δ and its relationship with RPM is approximately linear. Output variables are: effective motor voltage V_{motor}, current I_{motor} and torque Q_{motor}. Torque produced by the motor is limited by the input current through the K_t motor constant. Input current is limited by RPM and maximum voltage through the K_e motor constant. A more detailed study of these motors, and Brushless Permanent Magnet ones in particular, can be found in Chapter 4.4 and 4.8 of [11] and in [12].

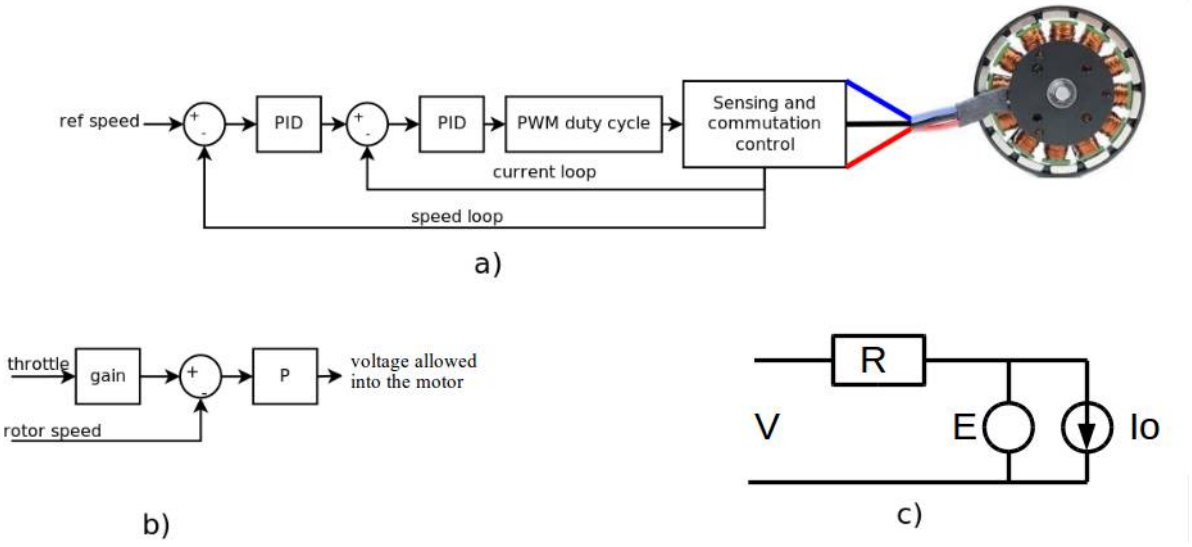

Fig. 5 (a) Block diagram of electronic speed controller; (b) ESC as implemented in the dynamic model; (c) electrical model of motor

1. Quasi-static assumption

Both the aerodynamic loads and the actuator forces/moments are a function of the vehicle/rotor state. In this work a quasi-static assumption is made, meaning that no maneuver is fast enough to cause the vehicle/rotor to change its state in a time interval comparable with the time scale required for the airflow field around the rotor to readjust [10]. The violation of this assumption is addressed by unsteady aerodynamic models, which are beyond the scope of this work.

On the other hand, motor torque Q_{motor} is upper bounded by the maximum current rating. Aerodynamic torque Q_{aero} increases with RPM. Since these are the main contributors of $\dot{\Omega}$ there is a natural, built-in limit on RPM change resulting in a time constant of about 1 second. This matches well with the quasi-static assumption and no further limitation is needed for the rotor.

2. Coupling of actuator and body dynamics

The introduction of rotors into the vehicle generates a coupling between each other's angular momentum. Chapter 10.9 of [13] shows in detail how rotors are now a dynamic system embedded in the larger dynamics of the body. Now, rotor's state can change the attitude of the body. Yaw rotation in a multicopter is a good example of this.

If the rotors create a non-zero angular momentum about the body center of mass, then the cross product of the vehicle angular velocity times the rotor's angular momentum will give rise to a gyroscopic moment (torque). Reversely, if one motor is trying to spin up a rotor, the torque it needs is going to change depending on the vehicle's angular momentum. This shows how both systems are coupled. The rotational dynamics of the vehicle is given by:

$$\mathbf{H}_{rot} = \sum_{i=1}^{N} \mathbf{J}_{rot-i}\boldsymbol{\Omega}_i \tag{19}$$

$$\mathbf{M}_{net} = (\mathbf{J}_b\dot{\boldsymbol{\omega}} + \dot{\boldsymbol{H}}_{rot}) + \boldsymbol{\omega} \times (\mathbf{J}_b\boldsymbol{\omega} + \mathbf{H}_{rot}) \tag{20}$$

$$\dot{\boldsymbol{\omega}} = (\mathbf{J}_b^{-1})(\mathbf{M}_{net} - \dot{\boldsymbol{H}}_{rot} - \boldsymbol{\omega} \times (\mathbf{J}_b\boldsymbol{\omega} + \boldsymbol{H}_{rot}) \tag{21}$$

Where $\mathbf{J}_b$ is the moment of inertia of the whole body and $\mathbf{J}_{rot-i}$ is the moment of inertia of the i^{th} rotor only. Therefore $\mathbf{M}_{rot}$ is defined as:

$$\mathbf{M}_{rot} = \dot{\boldsymbol{H}}_{rot} + \boldsymbol{\omega} \times \boldsymbol{H}_{rot} \tag{22}$$

On the other hand, the rotational dynamics of each motor is given by:

$$\mathbf{Q}_{net-i} = \mathbf{J}_{rot-i}(\dot{\boldsymbol{\Omega}}_i + \dot{\boldsymbol{\omega}}) + (\boldsymbol{\Omega}_i + \boldsymbol{\omega}) \times (\ \mathbf{J}_{rot-i}(\boldsymbol{\Omega}_i + \boldsymbol{\omega})\ ) \tag{23}$$

$$\dot{\boldsymbol{\Omega}}_i = (\mathbf{J}_{rot-i}^{-1})(\ \mathbf{Q}_{net-i} - \mathbf{J}_{rot-i}\dot{\boldsymbol{\Omega}}_i - (\boldsymbol{\Omega}_i + \boldsymbol{\omega}) \times (\ \mathbf{J}_{rot-i}(\boldsymbol{\Omega}_i + \boldsymbol{\omega})\ )\ ) \tag{24}$$

In this work, the effect of the rotor's angular momentum on the vehicle is neglected. Unless the vehicle is yawing the combined effect of rotors balances out and net angular momentum is zero. On the other hand, the effect of the vehicle's angular moment on the rotors is not ignored. If the rotors are constrained to rotate only around the z-axis of the xyz_i coordinates system, then the effect of the vehicle's rotation (as in a pitch-up maneuver) will produce an in plane moment on the rotor. Although not implemented in this work, this extra effect can be modelled as a friction term along the z-axis $\mathbf{Q}_{friction}$.

Now that $\mathbf{Q}_{motor}$, $\mathbf{Q}_{aero}$ and gyroscopic moments for each rotor are known the actuator dynamics are complete. The vehicle's input is the throttle command to each motor and the states are those of a rigid body plus rotor state.

IV. Trajectory planning and control for transition to powered flight

The TPF maneuver follows a two step process: first, a trajectory planner computes a safe, feasible trajectory from the initial nominal condition to the desired end state; second, a trajectory following controller tracks that trajectory. This implies two parts to generation of a safe TPF: a safe planned trajectory and a robust controller to follow that trajectory. It will be shown later that simply commanding an end condition (as a step input) leads to potentially unsafe transitions.

Here a nonlinear optimization that minimizes a quadratic cost function based on states and inputs is used to plan a nominal trajectory. To enable good tracking of this trajectory even from off-nominal initial conditions, particle swarm optimization (PSO) is used to determine controller gains that minimize deviation from the nominal trajectory for a predefined subset of initial conditions drawn from the space of possible initial conditions. This set of PSO controller gains will later be compared with gains obtained using Ziegler-Nichols tuning. In this section actuator dynamics were remove for simplicity, but they are taken back during the final safety evaluation in section V.

A. Optimal control problem

The theory of optimal control deals with the problem of finding the input function $u(t)$ that minimizes a given user defined cost function $J(t_0, x(t), u(t)$. Where $x(t)$ is the state vector of a dynamic system. The cost is a functional (a function of functions) and depending on its form the overall problem receives a different name, e.g minimum time, minimum energy, minimum fuel, etc. The classic optimal control problem (OCP) has been well studied in [14] [15] among many others. In the present TPF under study there is no need to penalize the vehicle position, hence the first three states (position $\mathbf{x}^{ned}$) are removed from the OCP. The state vector is redefined as $\mathbf{x} = [\ \mathbf{q}_{xyz/ned}\ \mathbf{v}^{ned}\ \boldsymbol{\omega}^{xyz}]^T$. The input vector is redefined as $\mathbf{u} = [T_1\ T_2\ T_3\ T_4]^T$. The OCP then takes the form.

$$\begin{aligned} \min_{\mathbf{u}(t)}\ J(t_0) &= \Phi(t_f, \mathbf{x}_f) + \int_{t_0}^{t_f} L(t, \mathbf{x}, \mathbf{u}) dt \\ \dot{\mathbf{x}} &= \mathbf{f}(t, \mathbf{x}, \mathbf{u}) \\ [0\ 0\ 0\ 0]^T &\leq \mathbf{u} \leq [300\ 300\ 300\ 300]^T \\ |\dot{\mathbf{u}}| &\leq [60\ 60\ 60\ 60]^T \end{aligned} \tag{25}$$

Where $\mathbf{f}(t, \mathbf{x}, \mathbf{u})$ is the dynamic system of the vehicle defined in previous sections. The input constraint are explained in following sections. Finally, $\Phi(t_f, \mathbf{x_f})$ and $L(t, \mathbf{x}, \mathbf{u}) \in \mathbb{R}$ and are defined as:

$$\Phi(t_f, \mathbf{x}_f) = (\mathbf{x} - \mathbf{x}_{ss})^T S(\mathbf{x} - \mathbf{x}_{ss}) \tag{26}$$

$$\mathbf{S} = diag(1, 1, 1, 1, 1, 1, 1, 10, 10, 10) \tag{27}$$

$$L(t, \mathbf{x}, \mathbf{u}) = (\mathbf{x} - \mathbf{x}_{ss})^T \mathbf{Q}(\mathbf{x} - \mathbf{x}_{ss}) + \mathbf{u}^T \mathbf{R}\mathbf{u} \tag{28}$$

$$\mathbf{Q} = \mathbf{S} \tag{29}$$

$$\mathbf{R} = 10^{-5} diag(1, 1, 1, 1) \tag{30}$$

$$\mathbf{x}_{ss} = [1\ 0\ -0.0026\ 0\ 4\ 0\ 4\ 0\ 0\ 0]^T \tag{31}$$

The steady state condition $\mathbf{x}_{ss}$ for the commanded velocity $\mathbf{v}^{ned}_{cmd} = [4\ 0\ 4]^T$ is determined using simulation. A minimum energy approach was selected for the cost function, weighting matrices are shown in equations 26, 27, 28, 29 30 and 31. The penalty for not achieving the final state conditions is increased with respect to the energy penalty (control effort).

The calculus of variations provides the necessary conditions that an optimal solution $\mathbf{u}_{opt}(t)$ must satisfy. In the case of nonlinear systems or nonlinear cost functions, the optimal solutions $\mathbf{x}_{opt}$ and $\mathbf{u}_{opt}$ (and the costate $\boldsymbol{\lambda}_{opt}$) commonly need to satisfy a two-point boundary value problem (2P BVP). This is a differential equation where the known state condition is split between t_0 and t_f. Multiple numerical solutions have been proposed to solve this 2P BVP. Alternatively, a group of approaches known as direct methods solve this OCP by directly discretizing the dynamics and constraints and then using nonlinear programming techniques to find the optimal solution. A review of different numerical methods to solve OCP (direct and indirect) is given in [16].

1. Direct collocation

In this work, a fixed final time direct collocation method is implemented and a perfect knowledge of the state is assumed. In direct collocation methods, the input and system dynamics are discretized into N points (collocation points). The discrete approximation of system dynamics are enforce by the collocation constraints $\mathbf{e}_k$. The cost functions $\Phi(t, \mathbf{x})$ and $L(t, \mathbf{x}, \mathbf{u})$ are replaced by their discrete approximation. $\Phi_d(t_k, \mathbf{x}_k)$ and $L_d(t_k, \mathbf{x}_k, \mathbf{u}_k)$. A piecewise polynomial interpolation of the state and input is used in between segments $(t_k, \mathbf{x}_k, \mathbf{u}_k)$ and $(t_{k+1}, \mathbf{x}_{k+1}, \mathbf{u}_{k+1})$. It is therefore possible to define the vector $\mathbf{y} = [x_1\ ..\ x_N u_1\ ..\ u_N]^T$, and calculate the discretized cost $J_d(\mathbf{y})$, while also enforcing the collocation constraints. Therefore, the OCP can be transcripted (transformed) into a nonlinear programming (NLP) problem. As usual, numerical methods of varying orders can be used in this process, a reference on this topic can be found in Chapter 4 of [17].

In this work, a trapezoidal approximation on the cost and a Hermite–Simpson approximation on the states was used. For each initial condition of the vehicle, an initial guess $\mathbf{x}_{guess}(t)$ and $\mathbf{u}_{guess}(t)$ is taken from the simulations of the near-hover controller described in following sections. Thirty seconds are studied, and time limits of the OCP

Table 2 Input constraints used in the direct collocation method

Variable T_i	Value
Minimum Thrust	0 N
Maximum Thrust	300 N
Maximum Thrust rate of change	60 N/s

are $t_0 = 0$ s, $t_f = 30$ s and $dt = (t_f - t_0)/N = 0.7895$ s. Thirty eight collocation points where used ($N = 38$), ranging from $(t_0, \mathbf{x}(t_0), \mathbf{u}(t_0))$ to $(t_f, \mathbf{x}(t_f), \mathbf{u}(t_f))$. The referred number provides a good balance between accuracy and execution time (about eight hours per case on a GNU / Linux system running on an Intel Core i7-7600U CPU @ 2.80GHz × 4). MATLAB provided fmincon algorithm was used to find the optimal solution of the NLP problem defined by

$$\begin{aligned} \min_{\mathbf{y}=[x_1\ ..\ x_N, u_1\ ..\ u_N]^T} \quad & J_d(\mathbf{y}) = \Phi_d(t_N, \mathbf{x}_N) + \sum_{k=1}^{N-1} L_d(t_k, \mathbf{x}_k, \mathbf{u}_k) \\ & \mathbf{e}_k = 0 \ \ \forall\ k \in \{1,\ ..\ N-1\} \\ & [0\ 0\ 0\ 0]^T \leq \mathbf{u}_k \leq [300\ 300\ 300\ 300]^T \ \ \forall\ k \in \{1,\ ..\ N\} \\ & \frac{1}{dt}|\mathbf{u}_{k+1} - \mathbf{u}_k| \leq [60\ 60\ 60\ 60]^T \ \ \forall\ k \in \{1,\ ..\ N-1\} \end{aligned} \tag{32}$$

where the last constraint is a first order numerical approximation of the input derivative $\dot{\mathbf{u}}$. The constraint on the input $\mathbf{u}_k$ are the same as in the original OCP but enforced at each collocation point. The collocation constraint e_k represents the error in the discretized system dynamics in the segment between t_k and t_{k+1}, and is described by:

$$\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}_{k+1} + \frac{dt}{6}(\mathbf{f}_k + 4\mathbf{f}_c + \mathbf{f}_{k+1}) \tag{33}$$

$$\mathbf{f}_c = \mathbf{f}(t_c, \mathbf{x}_c, \mathbf{u}_c) \tag{34}$$

$$\mathbf{u}_c = \frac{1}{2}(\mathbf{u}_k + \mathbf{u}_{k+1}) \tag{35}$$

$$\mathbf{x}_c = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{dt}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) \tag{36}$$

$$t_c = \frac{1}{2}(t_k + t_{k+1}) \tag{37}$$

The cost functions are described by:

$$\Phi_d(t_N, \mathbf{x}_N) = \Phi(t_N, \mathbf{x}_N) \tag{38}$$

$$L_d(t_k, \mathbf{x}_k, \mathbf{u}_k) = \frac{1}{2}(L(t_k, \mathbf{x}_k, \mathbf{u}_k) + L_d(t_{k+1}, \mathbf{x}_{k+1}, \mathbf{u}_{k+1}))\ dt \tag{39}$$

As previously mentioned, limits on motor current result in a maximum motor torque. Propeller dynamics and aerodynamic torque will then impose a limit on the rate at which thrust can change. These limits are enforced by setting a constraint on the maximum value of thrust and the maximum rate of change in thrust (equations 32 and 25). The values given in table 2 are more restrictive than what the actuator can actually achieve under certain conditions. However, constraints are applied uniformly along the trajectory and a conservative case is chosen.

The initial guess given to the NLP solver was purely longitudinal and so was the optimal trajectory found. But its important to notice that beside the collocation constraints, there were no further restrictions imposed on the states, this also includes the lateral velocity, roll and yaw. Finally, figure 7 shows the optimal trajectories for the vehicle when a steady state velocity $v_{cmd}^{ned} = [4\ 0\ 4]$ is desired.

Table 3 Subset of initial conditions under study

Name	velocity m/s	pitch deg	pitch rate deg/s
case-0 (nominal)	[+0.00 0 3.67] ^{T}	0.0	0
case-1	[+0.32 0 3.67] ^{T}	0.0	18.8
case-2	[+0.26 0 3.64] ^{T}	5.5	15.2
case-3	[+0.10 0 3.65] ^{T}	8.9	5.8
case-4	[-0.10 0 3.69] ^{T}	8.9	-5.8
case-5	[-0.26 0 3.70] ^{T}	5.5	-15.2
case-6	[-0.32 0 3.67] ^{T}	0.0	-18.8

B. Defining the space of possible initial conditions

Variation in initial conditions are induced by "wrist motion" of the multirotor under the parachute (i.e. pendulum oscillations around the parachute attachment point) causing non-zero pitch and pitch rate. The model for initial conditions considers a simple pendulum oscillation as seen in figure 6. The angle $\theta(t)$ is that between the nadir of the pivot and the vehicle center of mass. Because the nose of the vehicle is perpendicular to the pivot line, the initial pitch angle is also equal to θ. The parameters are the length of the pendulum $r = 0.34$ m, amplitude $\theta_0 \sim \mathcal{N}(\mu = 0°,\ \sigma = 3°)$, frequency $\omega = 2$ rad/s and phase $\phi \sim U[-\pi,\ +\pi]$.

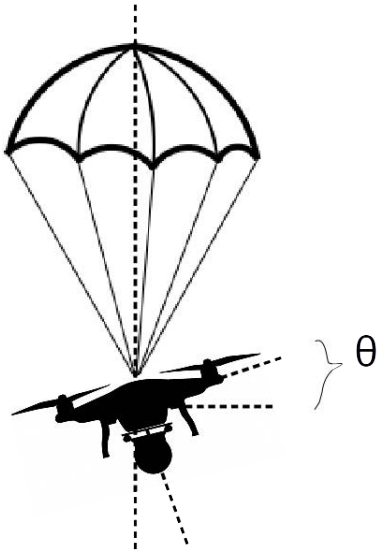

Fig. 6 Schematics of possible parachute oscillation

$$\theta(t) = \theta_0\ \sin(\omega t\ +\ \phi) \tag{40}$$

$$\dot{\theta}(t) = \omega\ \theta_0\ \cos(\omega t\ +\ \phi) \tag{41}$$

By taking into consideration the parachute descent velocity, safety delay and angular velocity the initial velocity becomes

$$v_n = +\dot{\theta}\ r\ \cos(\theta)\ |_{t=0} \tag{42}$$

$$v_d = v_0\ -\dot{\theta}\ r\ \sin(\theta)\ |_{t=0} \tag{43}$$

$$v_0 = 3.0\ +\ 1.35\ (t_d = 0.5s) = 3.6750\ m/s \tag{44}$$

Setting $t = 0$, $\theta_0 = 9.4°$ and varying ϕ, six initial conditions are obtained. Including the nominal case there is a total of seven initial conditions as shown in table 3.

Table 4 Longitudinal gains of vehicle PID controllers

	Pitch rate		Pitch	Vertical vel		Horizontal vel	
	K_P	K_I	K_P	K_P	K_I	K_P	K_I
Near-hover controller	2.0000	0.0010	0.6210	0.6750	0.0675	1.1000	0.0016
PSO controller	2.0794	0.0001	0.8463	5.0000	0.4338	1.2171	0.0188

C. Particle swarm optimization

Particle swarm optimization (PSO) is a powerful technique that is used for optimization problems where the dynamics or the objective function are discontinuous, non-differentiable, or highly nonlinear. PSO tries to find the optimum point in a given vector subspace. In this technique, there is a population of particles that are randomly distributed at the beginning. In each iteration the particles will move through the solution subspace in a direction that is a combination of its local gradient and the global one (that of the particle with lower cost). Gain tuning of PID controllers using PSO and genetic algorithms has been studied extensively (see for example [18], [19], and [20] for a multi-rotor specific example). A review on the topic can be found in [21].

The gains to be optimized by PSO are those involved in the longitudinal axis of the vehicle. This set is defined as the PSO state vector y_{PID}. A function mapping y_{PID} to the RMS error between the nominal optimal state trajectory $x_{opt}(t)$ and the simulated state trajectory $x_{sim}(t, y_{PID})$ is defined as

$$J^i_{RMS}(\mathbf{y}_{PID}) = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\|\mathbf{x}_{opt}(t_k) - \mathbf{x}^i_{sim}(t_k, \mathbf{y}_{PID})\|^2} \tag{45}$$

Unlike the in near-hover controller's case, there is now an optimal trajectory to follow. And $v^{ned}_{opt}(t)$ is used as the tracking reference $v^{ned}_{cmd}(t)$ in figure 3. This RMS value of the trajectory is calculated for each initial conditions and the maximum one is selected as the associated cost of a given $\mathbf{y}_{PID}$ according to

$$\min_{\mathbf{y}_{PID} \in Y} J(\mathbf{y}_{PID}) = \max_{J_{RMS}} \{ J^i_{RMS}(\mathbf{y}_{PID}) \mid i \in I \} \tag{46}$$

Where Y is the vector subspace of gains $[0.0001\ ,\ 5.0]^7 \subset \mathbb{R}^7$ and I is the set of initial conditions given in table 3.

This Min-Max strategy for the cost function increase the likelihood that even under different initial conditions, the PID controller can bring the vehicle back to the nominal optimal trajectory (or as close as possible). In other words, to make the controller more robust under varying initial conditions. The MATLAB provided algorithm particleswarm was used to optimize the gains. The execution time for this particular PSO problem was 6 days (GNU / Linux system running on an Intel Core i7-7600U CPU @ 2.80GHz × 4).

D. Near-hover controller

A common starting point for control design begins with a known steady state condition $(\mathbf{x}_{ss}, \mathbf{u}_{ss})$ around which the systems is expected to operate. For multicopters this condition is hover, and is the equilibrium condition where the vehicle has zero translational velocity and zero rotational velocity. The system is typically linearized around this point so that any of the multiple control design for linear systems can be applied. An extensive reference on PID techniques can be found on [22] and a summarized review is given in [21]. Additionally, a well-known reference on steady state conditions and linearization as applied to aerial vehicle can be found on Chapter 2 of [10].

Instead of linearization, a Ziegler-Nichols method is used to derive a starting point for PID gains. Since this is an heuristic method, the values were manually tuned in order to provide better performance for a step velocity command in forward flight. The need for tuning gains in order to improve vehicle response is recognized in [10]. Table 4 shows the final results of this process. The proportional and integral gains of the near-hover and PSO controller are summarized. Proportional gains for pitch rate, pitch and horizontal velocity are very similar. Nevertheless, all integral gains differ in one order of magnitude, the same difference is seen in the case of the proportional gain for vertical velocity.

V. Safety evaluation

In order to evaluate the performance of the two synthesized controllers, a state cost function $J_{state}(\mathbf{x}(t))$ is defined as

$$J_{state}(\mathbf{x}(t)) = \max\{J_v\ ,\ J_\theta\ ,\ J_Q\} \tag{47}$$

$$J_v = \frac{\|\mathbf{v}^{ned}\|}{V_{max}} \tag{48}$$

$$J_\theta = \frac{\theta}{\theta_{max}} \tag{49}$$

$$J_Q = \frac{Q}{Q_{max}} \tag{50}$$

Where θ is the vehicle pitch angle and Q is the body angular rate along the y-axis (pitch rate). Limits V_{max}, θ_{max} and Q_{max} are defined in table 1.

This cost represents the margins between vehicle's state and their their respective maximum allowable values. The higher the value of this function, the riskier the current state of the vehicle is. This time-dependant function is used to quantify the safety of the TPF maneuver and also evaluate each controller. A value of 1 means that at least one of the safety limits has been reached, and any value above 1 would put the vehicle under considerable risk. The vehicle needs to satisfy condition $J_{state}(\mathbf{x}(t)) < 1$ during the entire TPF maneuver.

Figure 7 shows the flight path and safety performance of both controllers under the initial conditions in table 3. Figure 8 shows the same, but only during the first few seconds. In both figures the point of release from the parachute is at (0,0). It is possible to identify the evolution of J_{state} and detect that the near-hover controller brings the vehicle to a safety value of 1 for some of the initial conditions under study, which is undesirable.

Table 5 Cost comparison of J_{TPF} for the near-hover controller and PSO controller

case	$J_1 = J_{TPF}$ near-hover controller	$J_2 = J_{TPF}$ PSO controller	Percentage difference $\frac{J_1 - J_2}{J_1}$
case-0	0.77	0.61	20 %
case-1	0.77	0.78	-1 %
case-2	0.90	0.63	29 %
case-3	1.01	0.55	45 %
case-4	0.98	0.60	38 %
case-5	0.88	0.67	23 %
case-6	0.80	0.70	13 %

By selecting the highest instantaneous cost along each trajectory it is possible to obtain a single value for the whole TPF maneuver and then compare between each case and controller as seen in table 5. As a reference, the maximum state cost of the optimal nominal trajectory is $J_{TPF} = 0.5475$.

$$J_{TPF} = \max_{t\ \in\ [0\ 30]} J_{state}(\mathbf{x}(t)) \tag{51}$$

The table and the two figure mentioned here show that the new controller has, in general, lower state cost than the near-hover controller, at least under the tested initial conditions. The worst case is a slight increase of 1 % in cost, while the best improvement is a decrease of 44 % in maneuver risk. Mean improvement is 24 % and the standard deviation is 15 %.

A. Monte Carlo simulation

In order to better analyze the performance of both controllers under initial conditions other than the ones used so far (table 3) a Monte Carlo simulation of 100 trials is performed. The initial conditions are realizations of the stochastic process detailed in equations 40 through 44.

Three Monte Carlo simulations were run. The first ones makes use of the near-hover controller and assumes that no trajectory is known. Therefore the controller uses the target steady state velocity as a step input command. The second

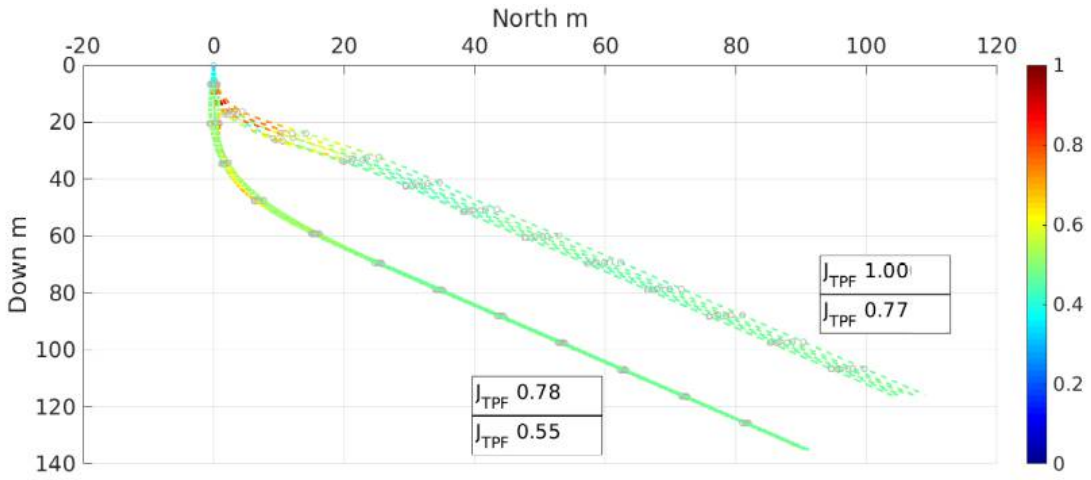

Fig. 7 Flight paths for near-hover controller trajectories (up and right, dashed lines), and PSO controller (down and left, dot and dashed lines). Simulation of initial conditions under study. The highest and lowest J_{TPF} are indicated at the bottom of each trajectory group. Marks are plotted at 1.56 s intervals (every 3 collocation points)

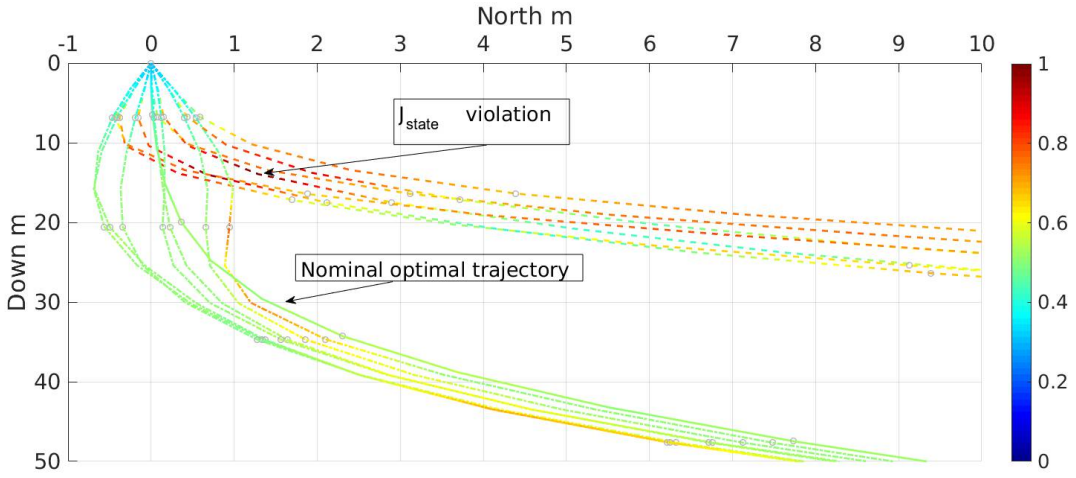

Fig. 8 First few seconds after parachute release. Flight paths for near-hover controller (up and right, dashed lines), and PSO controller (down and left, dot and dashed lines). Simulation of initial conditions under study. Marks are plotted at 1.56 s intervals (every 3 collocation points)

run uses the near-hover controller once again, but this time it tries to track the optimal trajectory. The third run uses the PSO controller combined with the optimal trajectory as well.

1. Near-hover controller and step input

Figure 9 shows the results of the first set of Monte Carlo simulation. Both plots depict the velocity trajectories of each run, with the left plot showing velocity expressed in inertial coordinates and the right plot expressed in body coordinates. The trajectories are color coded according to the state cost $J_{state}(t)$ in equation 47. The desired steady state velocity is $\mathbf{v}^{ned} = [4\ 0\ 4]^T$. The velocity variance at the moment of the release is about the same order of magnitude than at the end of the trajectory. This means that the controller is not consistently bringing the vehicle to the required final velocity.

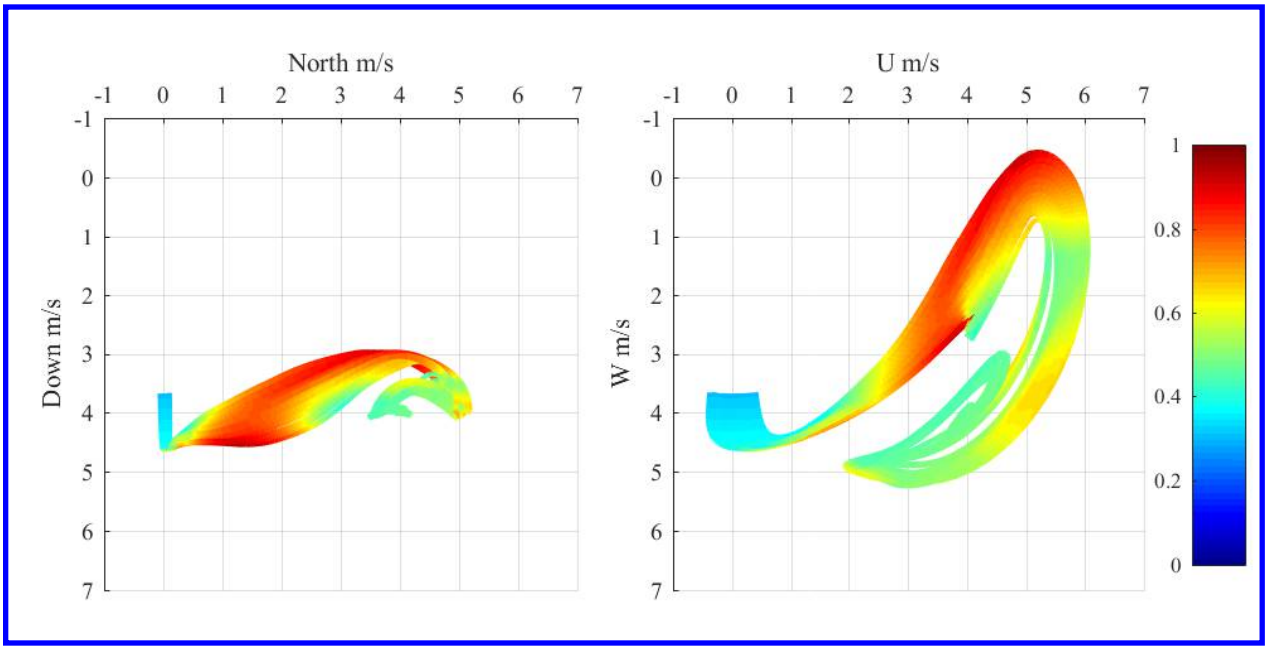

Fig. 9 Velocity trajectories of the Monte Carlo simulation: near-hover controller and step input velocity command $\mathbf{v}_{cmd} = [4\ 0\ 4]^T$. Trajectories are color coded according to $J_{state}(t)$

The lowest J_{TPF} cost is 0.7744 and the highest is 0.9110, mean is 0.81 and standard deviation is 0.04. Additionally, the dependency of J_{TPF} on initial pitch angle and initial pitch rate is depicted in figure 10. This cost function is more sensitive to large pitch angles (beyond ± 4 deg) than to pitch rate. The origin (0, 0) represents a nominal condition. The points are also color coded according to J_{TPF}.

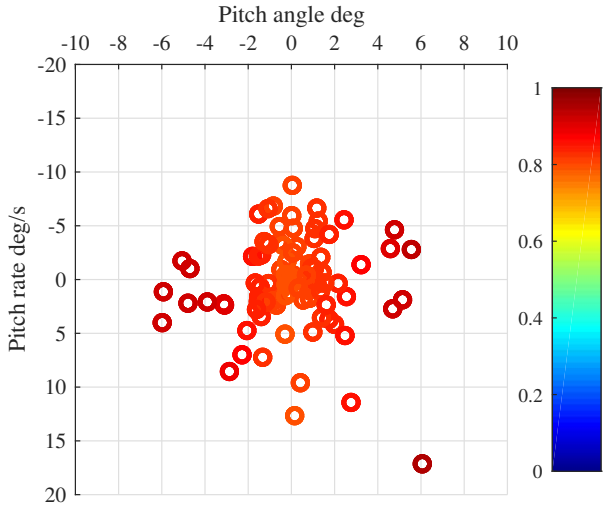

Fig. 10 Near-hover gains and step input; J_{TPF} cost for different initial values of θ and $\dot{\theta}$

2. Near-hover controller and optimal trajectory

Figure 11 shows the results of the second set of Monte Carlo simulations. Again, plots depict the velocity trajectories of 100 runs, with the left plot showing velocity expressed in inertial coordinates and the right plot expressed in body

coordinates. The trajectories are color coded according to the state cost $J_{state}(t)$ in equation 47. The desired steady state velocity is $\mathbf{v}^{ned} = [4\ 0\ 4]^T$. The velocity variance at the moment of the release is more than one order of magnitude bigger than at the end of the trajectory, indicating that this controller is consistently bringing the vehicle to the required final velocity.

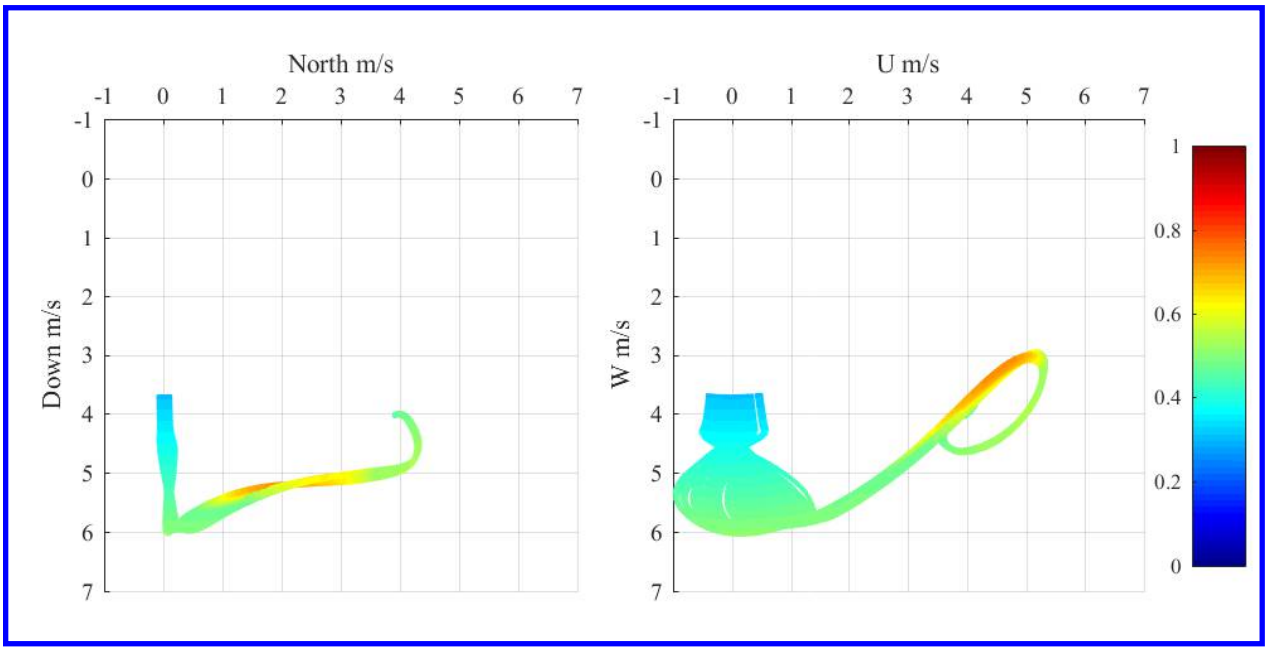

Fig. 11 Velocity trajectories of the Monte Carlo simulation: near-hover controller and reference tracking of optimal velocity $\mathbf{v}_{opt}(t)$. Trajectories are color coded according to $J_{state}(t)$

The lowest J_{TPF} cost is 0.6164 and the highest is 0.7222, mean is 0.65 and standard deviation is 0.02. The dependency of J_{TPF} on initial pitch angle and initial pitch rate is depicted in figure 12. This cost function is higher than average on the quadrant of negative pitch and positive pitch rates and lower than average in the quadrant of positive pitch angles and negative pitch rates. The origin (0, 0) represents a nominal condition. The points are also color coded according to J_{TPF}.

3. PSO controller and optimal trajectory

Figure 13 shows the results of the third Monte Carlo simulation. Again, plots depict the velocity trajectories of 100 runs, with the left plot showing velocity expressed in inertial coordinates and the right plot expressed in body coordinates. The trajectories are color coded according to the state cost $J_{state}(t)$ in equation 47. The desired steady state velocity is $\mathbf{v}^{ned} = [4\ 0\ 4]^T$. Again the velocity variance at the moment of the release is more than one order of magnitude bigger than at the end of the trajectory. This means that the controller is consistently bringing the vehicle to the required final velocity. The trajectory is also well tracked.

The lowest J_{TPF} cost is 0.6069 and the highest is 0.6358, mean is 0.62 and standard deviation is 0.01. This mean cost represents a reduction of 23 % when compared to the near-hover controller and step input, and 4.6 % when compared to near-hover controller and optimal trajectory.

The dependency of J_{TPF} on initial pitch angle and initial pitch rate is depicted in figure 14. This cost function is slightly higher than average for negative pitch rates, slightly lower than average for positive pitch rates and not sensitive to pitch angles. The origin (0, 0) represents a nominal condition. The points are color coded according to J_{TPF}.

VI. Conclusion

This paper addressed the problem of safely attaining powered multirotor flight after deployment from a parachute. The definition of transition to powered flight (TPF) was given together with its requirements. A dynamic model to simulate the vehicle behaviour was also derived. Then, a two step process was used to determine a safe trajectory from

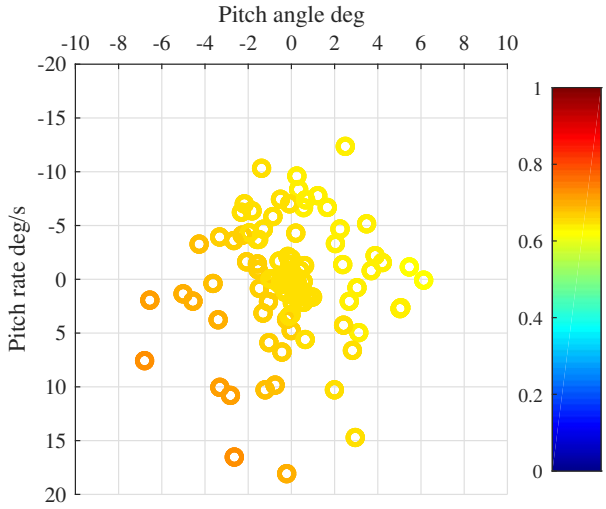

Fig. 12 Near-hover gains and trajectory tracking; J_{TPF} cost for different initial values of θ and $\dot{\theta}$

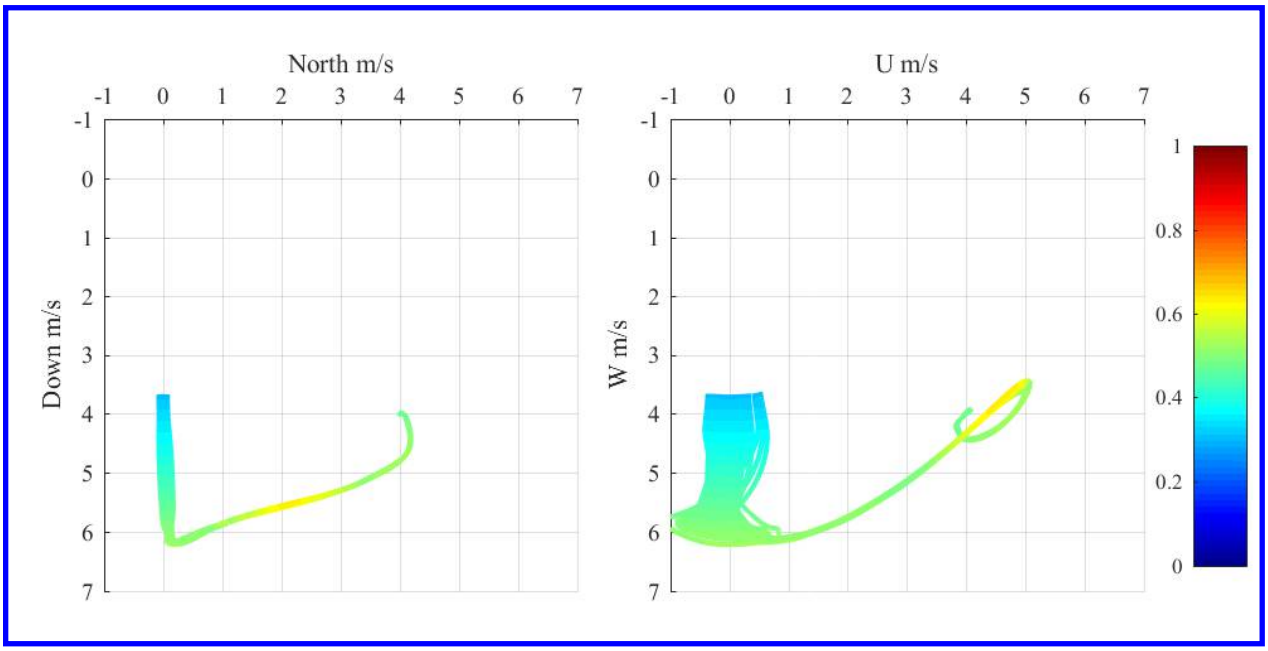

Fig. 13 Velocity trajectories of the Monte Carlo simulation: PSO controller and reference tracking of optimal velocity $\mathbf{v}_{opt}(t)$. Trajectories are color coded according to $J_{state}(t)$

the release condition to a steady state flight condition: trajectory optimization was used to plan a flight path from a nominal initial condition to a steady state powered flight condition; second, particle swarm optimization (PSO) was used to determine a set of controller gains that minimize deviation from this path even under off-nominal initial conditions. To test the flight path and the corresponding controller gains, a safety function based on maximum recommended vehicle's state was defined (velocity, angles and angular rates). Three Monte Carlo simulations were used to evaluate the state cost over the space of possible initial conditions.

The first one made use of the near-hover controller (derived using Ziegler-Nichols method) and assumed that no trajectory was known. The mean cost was 0.81 and standard deviation was 0.04. The second run used the same near-hover controller, but this time the optimal trajectory was tracked. The mean cost was mean 0.65 and standard

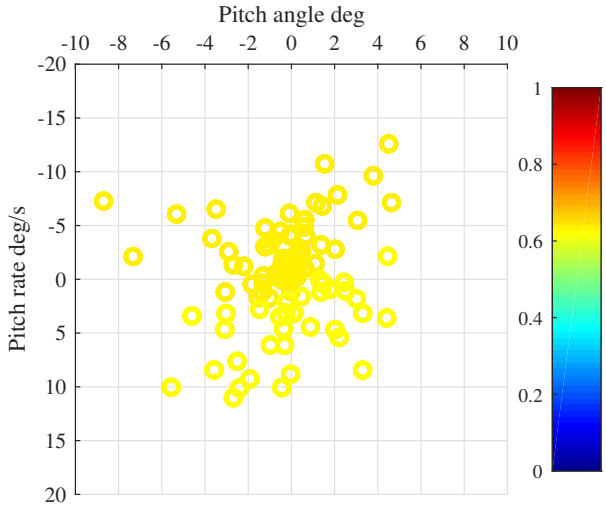

Fig. 14 PSO controller and optimal trajectory tracking; J_{TPF} cost for different initial values of θ and $\dot{\theta}$

deviation was 0.02. Finally, the third run used the PSO controller and the optimal trajectory was tracked. The mean cost was 0.62 and standard deviation was 0.01. Results showed that the combination of the optimal trajectory and the particle swarm derived controller successfully brought the vehicle into the desired steady state. And reduced the state cost of the maneuver by up to a 23% on average, when compared to the near-hover controller.

Acknowledgments

The authors thank Sven Schmitz and Jason Cornelius (Penn State University) for the rotor aerodynamic model and Michael Kinzel (University of Central Florida) for the body aerodynamic model.

References

[1] Schuyler, T. J., Gohari, S., Pundsack, G., Berchoff, D., and Guzman, M. I., "Using a Balloon-Launched Unmanned Glider to Validate Real-Time WRF Modeling," *Sensors*, Vol. 19, No. 8, 2019, p. 1914. doi:10.3390/s19081914.

[2] Parry, D., "Autonomous Deployment Demonstration Program Completes Flight Testing," online `https://www.nrl.navy.mil/news/releases/autonomous-deployment-demonstration-program-completes-flight-testing`, Accessed June 5 2019.

[3] Lorenz, R. D., Turtle, E. P., Barnes, J. W., Trainer, M. G., Adams, D. S., Hibbard, K. E., Sheldon, C. Z., Zacny, K., Peplowski, P. N., Lawrence, D. J., Ravine, M. A., McGee, T. G., Sotzen, K. S., MacKenzie, S. M., Langelaan, J. W., Schmitz, S., Wolfarth, L. S., and Bedin, P. D., "Dragonfly: A rotorcraft lander concept for scientific exploration at titan," *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)*, Vol. 34, No. 3, 2018, pp. 374–387.

[4] Langelaan, J. W., Schmitz, S., Palacios, J., and Lorenz, R. D., "Energetics of rotary-wing exploration of Titan," *2017 IEEE Aerospace Conference*, IEEE, 2017, pp. 1–11.

[5] McGee, T. G., Adams, D., Hibbard, K., Turtle, E., Lorenz, R., Amzajerdian, F., and Langelaan, J., "Guidance, Navigation, and Control for Exploration of Titan with the Dragonfly Rotorcraft Lander," *2018 AIAA Guidance, Navigation, and Control Conference*, 2018, p. 1330.

[6] AZUMA, A., "Induced flow variation of the helicopter rotor operating in the vortex ring state." *Journal of Aircraft*, Vol. 5, No. 4, 1968, pp. 381–386.

[7] Azuma, A., Koo, J., Oka, T., and Washizu, K., "Experiments on a model helicopter rotor operating in the vortex ringstate." *Journal of Aircraft*, Vol. 3, No. 3, 1966, pp. 225–230.

[8] Johnson, W., "Model for vortex ring state influence on rotorcraft flight dynamics," 2005.

[9] Johnson, W., *Helicopter theory*, Courier Corporation, 2012.

[10] Stevens, B. L., Lewis, F. L., and Johnson, E. N., *Aircraft control and simulation: dynamics, controls design, and autonomous systems*, John Wiley & Sons, 2015.

[11] Miller, T., *Permanent Magnet and Reluctance Motor Drives*, Oxford, UK: Oxford Science Publications, 1989.

[12] Lawrence, D., and Mohseni, K., "Efficiency analysis for long duration electric MAVs," *Infotech@ Aerospace*, 2005, p. 7090.

[13] Curtis, H. D., *Orbital mechanics for engineering students*, Butterworth-Heinemann, 2013.

[14] Bryson, A. E., *Applied optimal control: optimization, estimation and control*, Routledge, 2018.

[15] Lewis, F. L., Vrabie, D., and Syrmos, V. L., *Optimal control*, John Wiley & Sons, 2012.

[16] Rao, A. V., "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, Vol. 135, No. 1, 2009, pp. 497–528.

[17] Betts, J. T., *Practical methods for optimal control and estimation using nonlinear programming*, Vol. 19, Siam, 2010.

[18] Gaing, Z.-L., "A particle swarm optimization approach for optimum design of PID controller in AVR system," *IEEE transactions on energy conversion*, Vol. 19, No. 2, 2004, pp. 384–391.

[19] Chen, J., Omidvar, M. N., Azad, M., and Yao, X., "Knowledge-based particle swarm optimization for PID controller tuning," *2017 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2017, pp. 1819–1826.

[20] Mac, T. T., Copot, C., Duc, T. T., and De Keyser, R., "AR. Drone UAV control parameters tuning based on particle swarm optimization algorithm," *2016 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, IEEE, 2016, pp. 1–6.

[21] Bansal, H. O., Sharma, R., and Shreeraman, P., "PID controller tuning techniques: a review," *Journal of control engineering and technology*, Vol. 2, No. 4, 2012, pp. 168–176.

[22] Åström, K. J., and Hägglund, T., *PID controllers: theory, design, and tuning*, Vol. 2, Instrument society of America Research Triangle Park, NC, 1995.