

# Are we ready for Drone Racing?

## Optimal Control and Trajectory Estimation

Antoni Rosinol<sup>1</sup>

**Abstract**—In light of the recent interest in drone racing, both manned and unmanned, we present an approach to calculate the optimal control inputs and state trajectory of a quadcopter to finish a given race track in a minimum time. We assume the race track to consist of a set of arbitrarily positioned gates in 3D that the drone has to go through. We show how to solve this problem by using GPOPS-II, a matlab-based general purpose optimal control software. The hope of this work is to provide useful information for human pilots, and developers of autonomous drones, as would be provided by a tactician in an air race competition.

**Index Terms**—Optimal Control, Optimal Trajectory, Quadcopter, Drone Racing

### SUPPLEMENTARY MATERIAL

Videos of the experiments: **TODO:** Add video url  
Open-source code: **TODO:** Add code url

### I. INTRODUCTION

THE advent of drone racing competitions, both manned and unmanned, requires the estimation of the fastest possible trajectory through a given race track. Finding this optimal trajectory can be useful for human pilots, and their tacticians, to evaluate their performance and improve upon it. For example, the teams in the Red Bull Air Race [1] make use of such information to train and perfect their maneuvers. In this same way, drones can autonomously follow a dynamically feasible trajectory with high accuracy. This is especially interesting for autonomous drone races such as the Alpha Pilot race [2]. Therefore, being able to infer the optimal feasible trajectory could potentially make fully-autonomous drones beat human pilots. We are still not there, but great progress in this direction is being made [3].

#### Contributions.

- Formulation of the minimum-time problem for the specific problem of drone-racing through multiple gates.
- Presentation of a means to solve the problem using readily available software.

**Paper Structure.** Section II presents the mathematical formulation of our approach, and discusses the assumptions made. Section IV reports and discusses the experimental results.

<sup>1</sup>A. Rosinol is with the Laboratory for Information & Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA, USA, arosinol@mit.edu

### II. APPROACH

We formulate our drone race optimization problem as a minimum time optimal control problem with nonlinear dynamics and constraints. Therefore, we must define both the dynamics of a drone and the constraints to pass through each gate. Section II-A, II-B and II-C present the nonlinear dynamics for a quadcopter. Section II-D casts the quadcopter dynamics using a state-space model. The state-space formulation allows us to present in a compact form the optimization problem that we detail in section III.

#### A. Mathematical Model of a Quadcopter

The absolute position  $\xi$  of the drone is given in an Euclidean inertial frame, referred to as *world frame* ( $G$ ), by:  $x, y, z$ . This corresponds to the 3D coordinates of the center of mass of the drone. The attitude  $\eta$  of the drone with respect to the world frame is given using Euler angles: roll  $\phi$ , pitch  $\theta$  and yaw  $\psi$ .

$$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}$$

Figure 1 shows the frame of reference of the drone, referred to as *body frame* ( $B$ ), with respect to the world frame.

Linear velocities (body frame):  $V_B$

Angular velocities (body frame):  $\nu$

$$V_B = \begin{bmatrix} v_{x,B} \\ v_{y,B} \\ v_{z,B} \end{bmatrix}, \quad \nu = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

The rotation matrix from the body frame  $B$  to the world frame  $G$  is given by:  $X^B = R_G^B X^G = R(\phi)R(\theta)R(\psi)X^G$

$$R_B^G = R = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix}$$

where  $S_x = \sin(x)$  and  $C_x = \cos(x)$ .

Time derivative of this rotation matrix provides us with the angular velocities. These are not simply the time derivatives of the independent Euler angles, instead:

$$\dot{\eta} = W_\eta^{-1} \nu, \quad \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi / C_\theta & C_\phi / C_\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Conversely,

$$\nu = W_\eta \dot{\eta}, \quad \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & C_\theta S_\phi \\ 0 & -S_\phi & C_\theta C_\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

where  $T_x = \tan(x)$ . The matrix  $\mathbf{W}_\eta$  is invertible if  $\theta \neq (2k-1)\phi/2$ , ( $k \in \mathbb{Z}$ )

Symbol	Definition
$\xi$	Absolute position (inertial frame)
$\eta$	Attitude (inertial frame)
$\mathbf{V}_B$	Linear velocities (body frame)
$\nu$	Angular velocities (body frame)
$\mathbf{R}$	Rotation matrix from body to inertial frame
$\mathbf{W}_\eta$	Transformation matrix for angular velocities from inertial to body frame
$\mathbf{I}$	Inertia matrix
$\mathbf{G}$	Gravity

TABLE I: Definitions and Notations

#### Assumptions:

- The quadrotor structure is rigid and symmetrical with a center of mass aligned with the center of the body frame of the vehicle. The four arms of the quadcopter are aligned with the body x- and y-axes. Therefore, the inertia matrix  $\mathbf{I}$  is diagonal:

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

We further have that due to symmetry, that the inertial components in the x- and y-axes are equal:  $I_{xx} = I_{yy}$ .

- The thrust and drag of each motor is proportional to the square of the motor velocity.
- The propellers are considered to be rigid and therefore blade flapping is negligible (deformation of propeller blades due to high velocities and flexible material).
- The Earth is flat and non-rotating (difference of gravity by altitude or the spin of the earth is negligible).
- Ground effects that increase lift and decrease aerodynamic drag when flying close to the ground is considered negligible.

These assumptions and basic dynamics lead to the model used in this work.

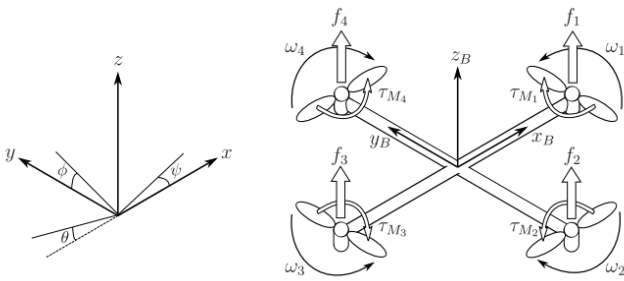


Fig. 1: The inertial and body frames of a quadcopter using the East North Up convention (ENU). Figure from [4].

#### B. Quadcopter Dynamics

The following derivations are a summary of the equations in [4].

- The angular velocity of rotor  $i$ , denoted with  $\omega_i$ , creates force  $f_i$  in the direction of the rotor axis:

$$f_i = k\omega_i^2$$

where the lift constant is  $k$ . The combined forces of rotors create thrust  $T$  in the direction of the body z-axis (represented as  $\mathbf{T}_B$ ).

$$T = \sum_{i=1}^4 f_i = k \sum_{i=1}^4 \omega_i^2, \quad \mathbf{T}_B = \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix}$$

- The angular velocity and acceleration of the rotor also creates torque  $\tau_{M_i}$  around the rotor axis:

$$I_M \dot{\omega}_i = \tau_{M_i} - b\omega_i^2$$

in which the drag constant is  $b$  and the inertia moment of the rotor is  $I_M$ . Usually the acceleration of the rotor ( $\dot{\omega}_i$ ) is considered small and thus it is omitted. This holds when we assume that the quadrotor is operating in stable flight and that the propellers are maintaining a constant thrust and not accelerating. This assumption results in the torque about the global z axis being equal to the torque due to drag.

Torque  $\tau_B$  consists of the torques on the three axis  $\tau_\phi, \tau_\theta, \tau_\psi$ :

$$\begin{aligned} \tau_B = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} &= \begin{bmatrix} lk(-\omega_2^2 + \omega_4^2) \\ lk(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^4 (-1)^{i+1} \tau_{M_i} \end{bmatrix} \\ &= \begin{bmatrix} lk(-\omega_2^2 + \omega_4^2) \\ lk(-\omega_1^2 + \omega_3^2) \\ b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \end{aligned}$$

where  $l$  is the distance between the rotor and the center of mass of the quadcopter.

- 1) *Translational dynamics:* Newton's second law:

$$m\ddot{\xi} = \mathbf{G} + \mathbf{R}\mathbf{T}_B$$

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{T}{m} \begin{bmatrix} C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ C_\theta C_\phi \end{bmatrix}$$

- 2) *Rotational dynamics:* The rotational equations of motion are defined in the body frame so that the rotations can be computed about the quadrotor's center and not the center of the global coordinate frame. Applying Euler's second law:

$$\mathbf{I}\dot{\nu} + \nu \times (\mathbf{I}\nu) + \mathbf{\Gamma} = \tau$$

where we have:

- Angular acceleration of the inertia:  $\mathbf{I}\dot{\nu}$
- Centripetal forces:  $\nu \times (\mathbf{I}\nu)$
- Gyroscopic forces:  $\mathbf{\Gamma}$
- External torque:  $\tau_B$

Replacing terms by their definitions, multiplying both sides by  $\mathbf{I}^{-1}$ , and rearranging:

$$\dot{\nu} = \mathbf{I}^{-1} \left( - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx}p \\ I_{yy}q \\ I_{zz}r \end{bmatrix} - I_r \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega_\Gamma + \tau_B \right)$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} (I_{yy} - I_{zz})qr/I_{xx} \\ (I_{zz} - I_{xx})pr/I_{yy} \\ (I_{xx} - I_{yy})pq/I_{zz} \end{bmatrix} - I_r \begin{bmatrix} q/I_{xx} \\ -p/I_{yy} \\ 0 \end{bmatrix} \omega_\Gamma + \begin{bmatrix} \tau_\phi/I_{xx} \\ \tau_\theta/I_{yy} \\ \tau_\psi/I_{zz} \end{bmatrix}$$

where  $\omega_\Gamma = \omega_1 - \omega_2 + \omega_3 - \omega_4$ , and  $I_r$  is the rotor moment of inertia.

The angular accelerations  $\ddot{\eta}$  in world frame are given by the time derivatives of the angular velocities ( $\dot{\eta} = \mathbf{W}_\eta^{-1}\nu$ ),

$$\ddot{\eta} = \frac{d}{dt}(\mathbf{W}_\eta^{-1}\nu) = \frac{d}{dt}(\mathbf{W}_\eta^{-1})\nu + \mathbf{W}_\eta^{-1}\dot{\nu} =$$

$$\begin{bmatrix} 0 & \dot{\phi}C_\phi T_\theta + \dot{\theta}S_\phi/C_\theta^2 & -\dot{\phi}S_\phi C_\theta + \dot{\theta}C_\phi/C_\theta^2 \\ 0 & -\dot{\phi}S_\phi & -\dot{\phi}C_\phi \\ 0 & \dot{\phi}C_\phi/C_\theta + \dot{\phi}S_\phi T_\theta/C_\theta & -\dot{\phi}S_\phi/C_\theta + \dot{\theta}C_\phi T_\theta/C_\theta \end{bmatrix} \nu + \mathbf{W}_\eta^{-1}\dot{\nu}.$$

At this point, it is common to do a simplification by setting  $[\dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T = [p \ q \ r]^T$ , which holds true for small angles of movement [5]. Nevertheless, in this work we omit such a simplification.

### C. Aerodynamic Drag

We include the drag force generated by the air resistance. For this, we add a diagonal coefficient matrix that associates the linear velocities to the force slowing down the movement

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{T}{m} \begin{bmatrix} C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ C_\theta C_\phi \end{bmatrix} - \frac{1}{m} \begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$$

in which  $A_x, A_y$  and  $A_z$  are the drag force coefficients for velocities in the corresponding directions of the inertial frame. Several other aerodynamical effects could be included in the model. For example, dependence of thrust on angle of attack, blade flapping and airflow disruptions. We refrain from adding these for simplicity of the model. We also ignore rotational drag forces since we assume rotational velocities to be small. Alternatively, we could add the components  $\tau_w = [\tau_{wx} \ \tau_{wy} \ \tau_{wz}]^T$  to the overall torque.

### D. State-space Model

We can write our nonlinear dynamics using the following state vector:

$$\mathbf{X}^T = [x \ y \ z \ \dot{x} \ \dot{y} \ \dot{z} \ \phi \ \theta \ \psi \ p \ q \ r]^T$$

Moreover, we define our inputs as follow:

- $U_1$ : the resulting thrust of the four rotors.
- $U_2$ : the difference of thrust between the motors on the  $x$  axis which results in roll angle changes and subsequent movement in the lateral  $x$  direction.

- $U_3$ : the difference of thrust between the motors on the  $y$  axis which results in pitch angle changes and subsequent movement in the lateral  $y$  direction.
- $U_4$ : the difference of torque between the clockwise and counterclockwise rotors which results in a moment that rotates the quadrotor around the vertical  $z$  axis.

This results in the control vector  $U$  defined as:

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} T \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} k \sum_{i=1}^4 \omega_i^2 \\ lk(-\omega_2^2 + \omega_4^2) \\ lk(-\omega_1^2 + \omega_3^2) \\ b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix}.$$

Our nonlinear dynamics can be written as a nonlinear differential equation of the states  $\mathbf{X}$  and control inputs  $U$ :

$$\dot{\mathbf{X}} = f(\mathbf{X}, U). \quad (1)$$

Or, more precisely:

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \frac{T}{m}[S_\phi S_\psi + C_\phi C_\psi S_\theta] - \frac{A_x}{m}\dot{x} \\ \frac{T}{m}[C_\phi S_\psi S_\theta - C_\psi S_\phi] - \frac{A_y}{m}\dot{y} \\ -g + \frac{T}{m}[C_\phi C_\theta] - \frac{A_z}{m}\dot{z} \\ p + r[C_\phi T_\theta] + q[S_\phi T_\theta] \\ q[C_\phi] - r[S_\phi] \\ r\frac{C_\phi}{C_\theta} + q\frac{S_\phi}{C_\theta} \\ \frac{I_y - I_z}{I_x}rq - \frac{I_r}{I_x}q\omega_\Gamma + \frac{\tau_\phi}{I_x} \\ \frac{I_z - I_x}{I_y}pr + \frac{I_r}{I_y}p\omega_\Gamma + \frac{\tau_\theta}{I_y} \\ \frac{I_x - I_y}{I_z}pq + \frac{\tau_\psi}{I_z} \end{bmatrix} = f(\mathbf{X}, U)$$

### III. MINIMUM TIME OPTIMAL CONTROL PROBLEM

A P-phase optimal control problem can be stated in the following general form. Determine the state  $\mathbf{x}^{(p)}(t) \in \mathbb{R}^{n(p)}$ , control,  $\mathbf{u}^{(p)}(t) \in \mathbb{R}^{n(t)}$ , initial time,  $t_0^{(p)} \in \mathbb{R}$ , final time,  $t_f^{(p)} \in \mathbb{R}$ , in each phase  $p \in [1, \dots, P]$ , that minimize a given cost functional, subject to dynamic constraints:

$$\dot{\mathbf{x}}^{(p)} = \mathbf{a}^{(p)}[\mathbf{x}^{(p)}, \mathbf{u}^{(p)}, t^{(p)}], \quad (p = 1, \dots, P)$$

**Cost:** our objective is to minimize the time taken for the drone to pass through all gates in a particular order. Therefore, the cost we try to minimize is simply the final time:

$$J = \sum_p \int_{t_0^{(p)}}^{t_f^{(p)}} dt = \int_{t_0}^{t_f} dt = t_f$$

where  $t_0 = 0$  is fixed, but  $t_f$  is free. We consider that the drone has finished the race once it has reached the last gate.

**Nonlinear Dynamics:** are formulated using the generic form:  $\dot{x}(t) = f(x(t), u(t), t)$ . As detailed in eq. (1), we make use of a nonlinear function of the state  $\mathbf{X}(t)$  and the input  $\mathbf{U}(t)$ . Although the nonlinear function  $f$  may depend explicitly on time  $t$ , the drone dynamics do not depend directly on  $t$ , therefore the time dependency can be dropped.

**Phases:** the racetrack is constructed in a piecewise fashion using phases, where each phase spans from one gate to the next one. Therefore, we consider as many phases as there are gates in the racetrack.

#### A. Bounds

In order to simulate a realistic drone we must constraint the control inputs to a range of possible values.

- **Input Bounds:** generic input constraints can be formulated as:

$$M_i^- \leq U_i(t) \leq M_i^+,$$

where  $M_i^-$  is the lower bound of control input  $U_i(t)$ , while  $M_i^+$  is its upper bound. In our case, we use the bounds in table II, which were calculated using the following heuristics. For bounds on  $U_1$  input (thrust) the minimum bound is 0 since we consider that the quadcopter has fixed-pitch propellers. With a variable pitch propeller, one may expect the thrust to be negative as well. The upper bound is calculated as  $4kl\omega_{\max}^2$ , where  $k$  is the lift constant and  $\omega_{\max}$  is the maximum motor speed. For  $U_2, U_3$  upper and lower bounds we use  $\pm 4kl\omega_{\max}^2$ , where  $l$  is the quadrotor's arm length. Similarly, for the yaw component  $U_4$ , we use  $\pm 2b\omega_{\max}^2$ , where  $b$  is the drag constant.

Input	Definition	$M_i^-$	$M_i^+$
$U_1$	Thrust	0.0	43.5
$U_2$	Roll rate	-6.25	6.25
$U_3$	Pitch rate	-6.25	6.25
$U_4$	Yaw rate	-2.25	2.25

TABLE II: Constraints on control inputs.

- **State Bounds:** we also bound the state space as detailed in table III.

State	Definition	$M_i^-$	$M_i^+$
$[x, y, z]$	Absolute Position	-10	10
$[\dot{x}, \dot{y}, \dot{z}]$	Linear Speed	-10	10
$[\phi, \theta]$	Roll and Pitch	$-\pi/2$	$\pi/2$
$\psi$	Yaw	$-\pi$	$\pi$
$[p, q, r]$	Angular rate	$-50(2\pi/360)$	$50(2\pi/360)$

TABLE III: Constraints on control inputs.

- **Time Bounds:** it is also necessary to specify the initial and final time bounds for each phase. In our case, we only fix the initial time to 0 for the first phase by setting its lower and upper bound to 0 as well. Conversely, final

times are left free and given conservative lower and upper bounds.

#### Boundary Conditions:

Initial conditions for the quadcopter at the start of the first phase are such that the quadcopter is at rest (no linear or angular velocities), and positioned at the first gate. For the rest of the phases, the initial conditions are set such that the drone is centered at the gate, with a small tolerance of 0.01m, and with an arbitrary orientation. Similarly, both linear velocities and angular rates are left virtually free by setting loose bounds.

#### B. Initial Guess

Our initial guess for the states at the initial and final times for each phase are straightforward. We assume that the drone takes about a second to go from gate to gate, as all gates are approximately at the same distance. Moreover, we assume that the state of the quadcopter is at the center of each gate, and that the control input is such that the drone is hovering.

### IV. EXPERIMENTAL RESULTS

#### A. State-Control Results

#### B. Modelling Limits and Extensions

### V. CONCLUSION

### VI. SUPPLEMENTARY MATERIAL

Euler angle rotation matrices:

$$\begin{aligned} \mathbf{R}(\psi) &= \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ \mathbf{R}(\theta) &= \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \\ \mathbf{R}(\phi) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \end{aligned}$$

### REFERENCES

- [1] "Home page — Red Bull Air Race." [Online]. Available: <https://airrace.redbull.com/en>
- [2] "AlphaPilot AI Drone Innovation Challenge — Lockheed Martin." [Online]. Available: <https://www.lockheedmartin.com/en-us/news/events/ai-innovation-challenge.html>
- [3] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler, and D. Scaramuzza, "Are we ready for autonomous drone racing? the uzhfpv drone racing dataset," in *IEEE Int. Conf. Robot. Autom.(ICRA)*, 2019.
- [4] "Modelling and control of quadcopter," Tech. Rep. [Online]. Available: [http://sal.aalto.fi/publications/pdf-files/eluu11{ }\\_public.pdf](http://sal.aalto.fi/publications/pdf-files/eluu11{ }_public.pdf)
- [5] F. Sabatino, "Quadrotor control: modeling, nonlinear control design, and simulation," *Medical Journal of Malaysia*, vol. 41, no. 3, pp. 278–280, 2015.