

Shu-peng LAI, Meng-lu LAN, Ya-xuan LI, Ben M. CHEN, 2019. Safe navigation of quadrotors with jerk limited trajectory. *Frontiers of Information Technology & Electronic Engineering*, 20(1):107-119. <https://doi.org/10.1631/FITEE.1800719>

Safe navigation of quadrotors with jerk limited trajectory

Key words: Quadrotor; Unmanned aerial vehicle; Motion planning

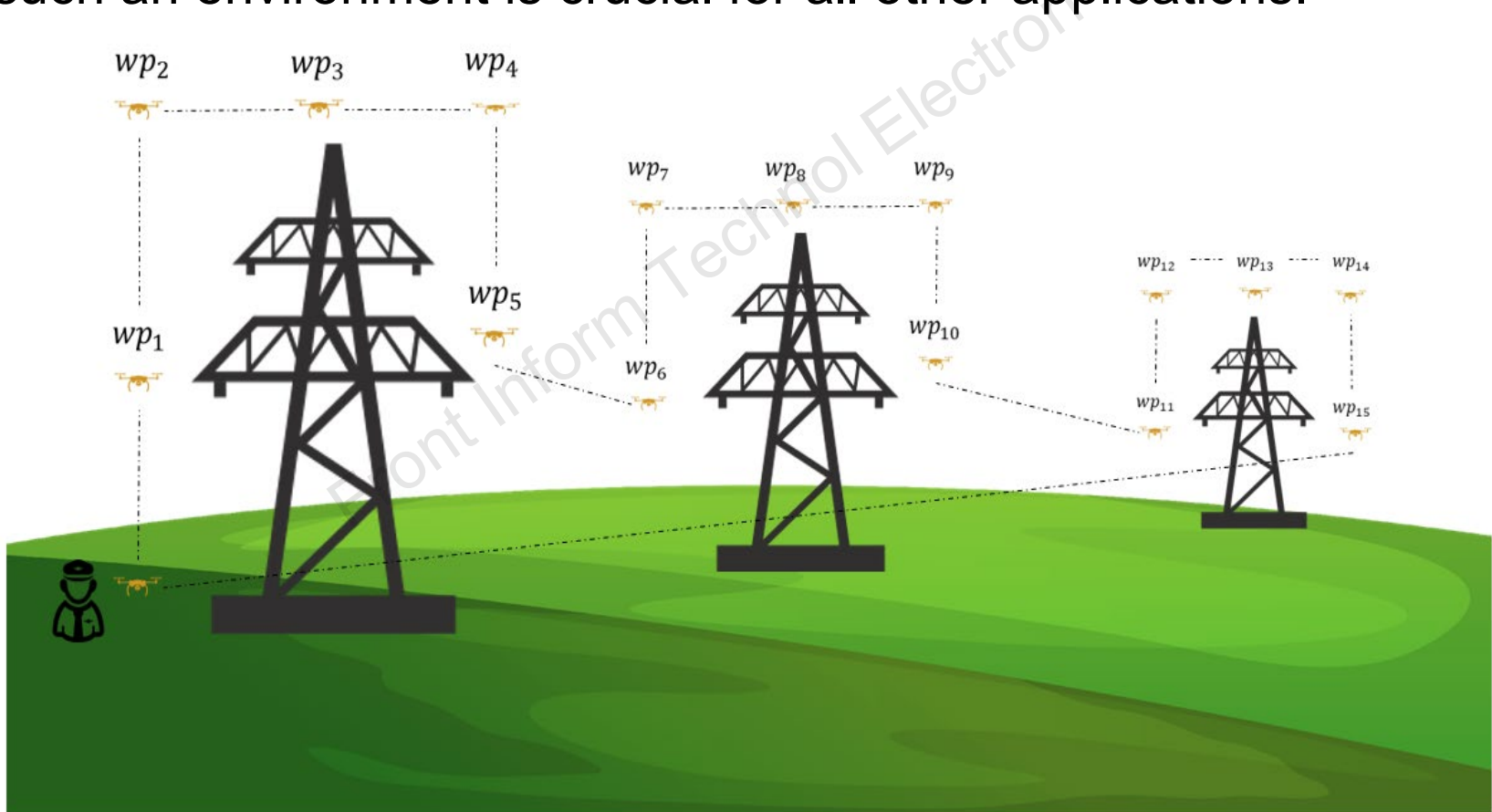
Corresponding author: Shu-peng LAI

E-mail: elelais@nus.edu.sg

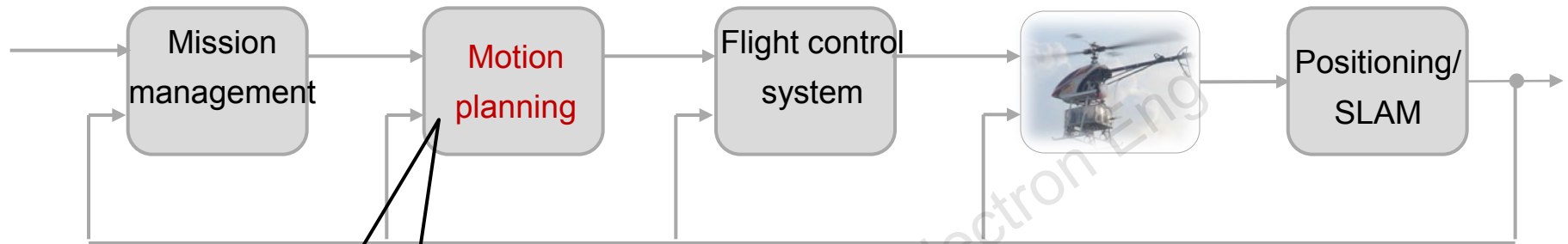
 ORCID: <http://orcid.org/0000-0003-2597-5392>

Motivation

Quadrotors are usually tasked to fly in low-altitudes and obstacle-strewn environments. The ability to navigate safely in such an environment is crucial for all other applications.

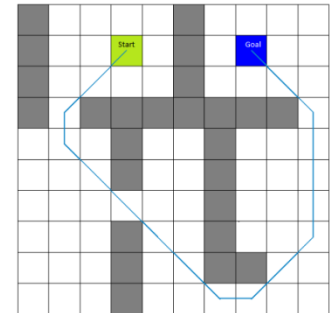
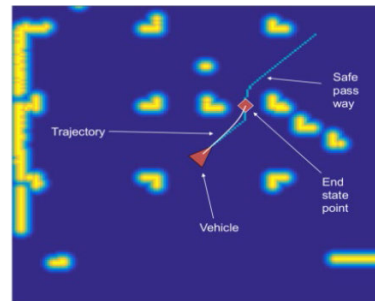


System structure



Nominal path plan consists of **line-segments** from the user or other algorithms.

Trajectory generation: smooth and feasible trajectory.



Main idea

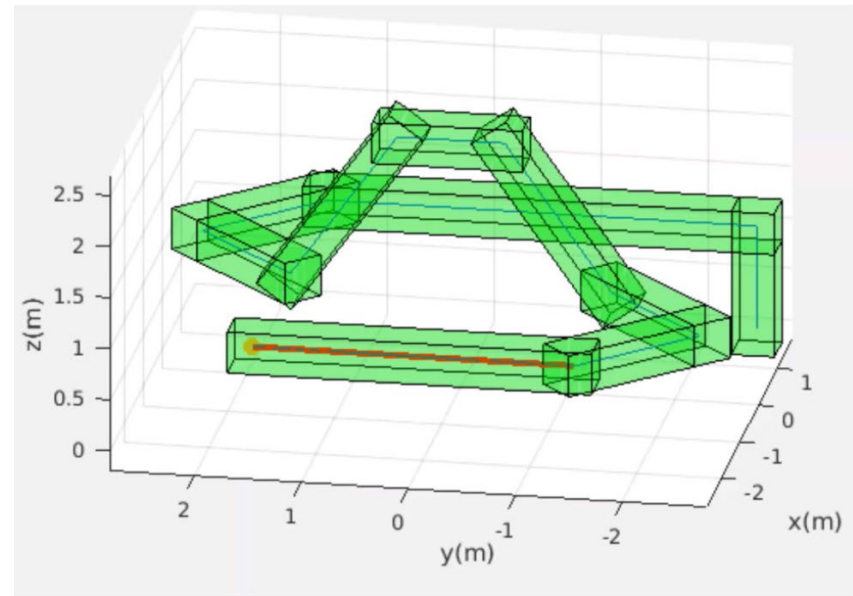
Incremental trajectory generation in safe corridor:

1. An **efficient** algorithm that generates **3D** velocity, acceleration, and jerk-limited trajectory.

2. Instant reaction to change environment or mission.

3. Navigating the nominal path plan:

- (1) safely (safe flying corridor)
- (2) smoothly (C^2 continuity)
- (3) efficiently (computing & flying)

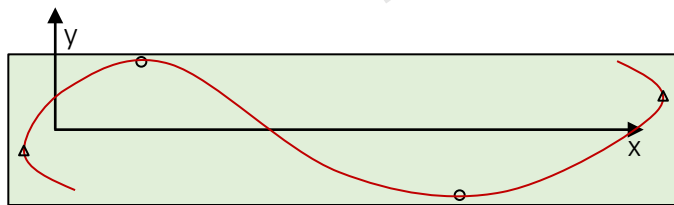




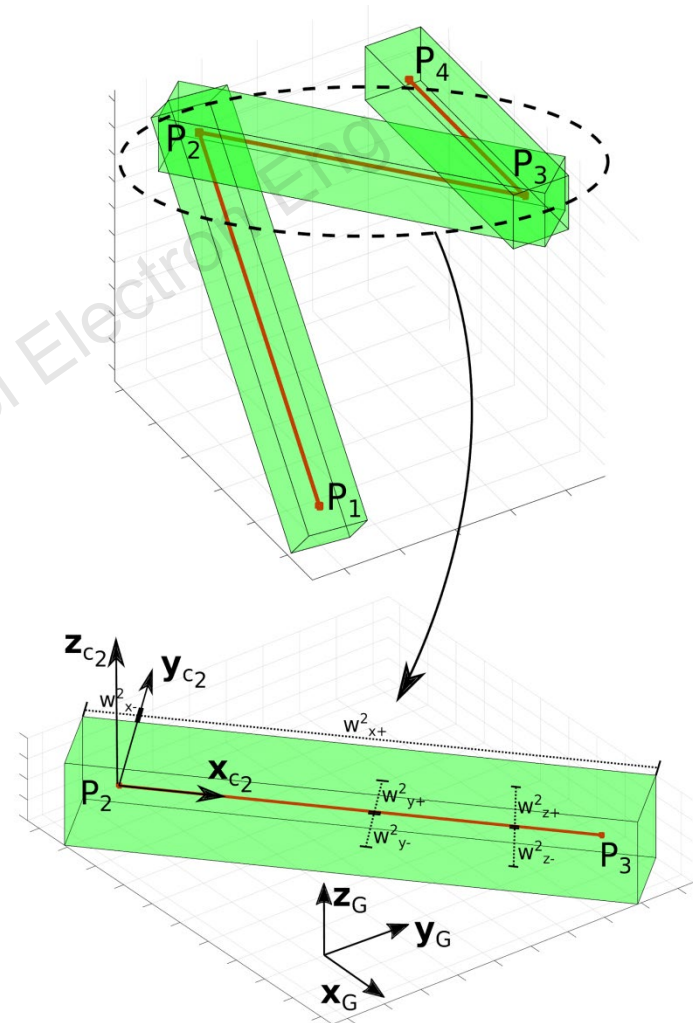
Method overview: safe flying corridor (SFC)

A series of inter-connected **bounding boxes**:

1. Encapsulate the nominal path plan;
2. Aligned with each line-segment;
3. Defines a bounding-box frame;
4. Easy inside checking.



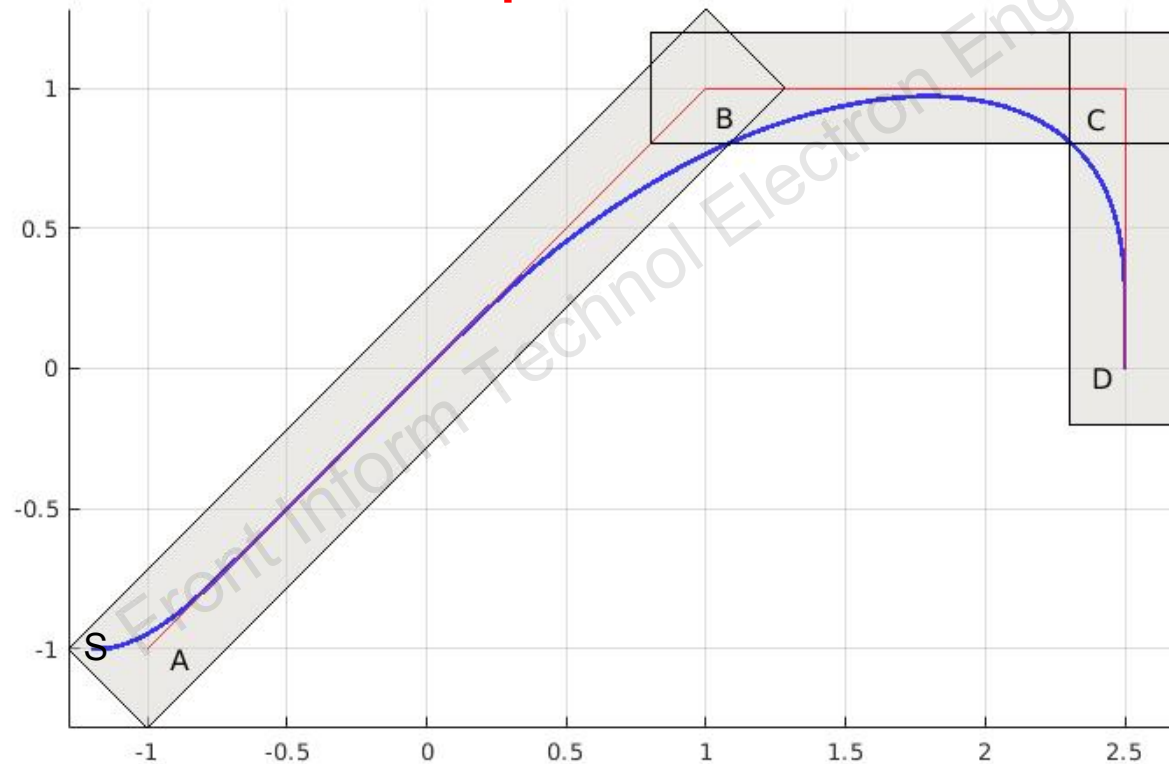
- y axis min/max
- △ x axis min/max





Method overview: SFC navigation

Online and incremental,
sample and throw

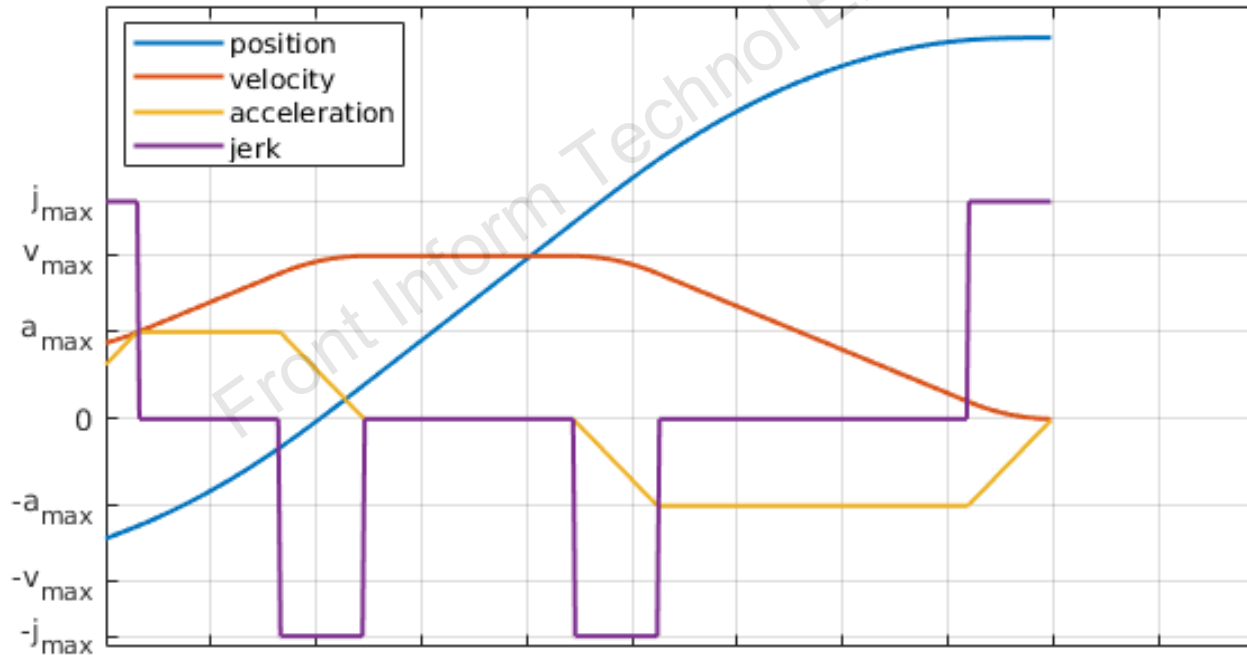


1. How to generate the **feasible** trajectory **fast enough**?
2. How to test its safety **efficiently**?



The jerk limited trajectory

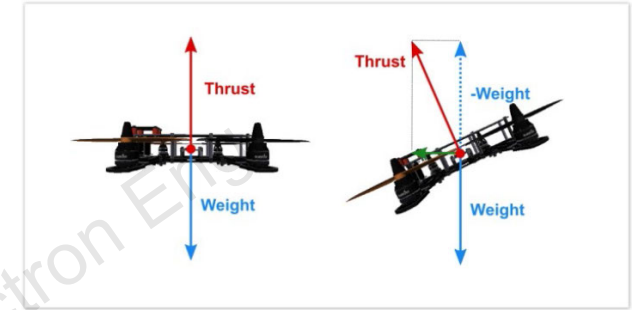
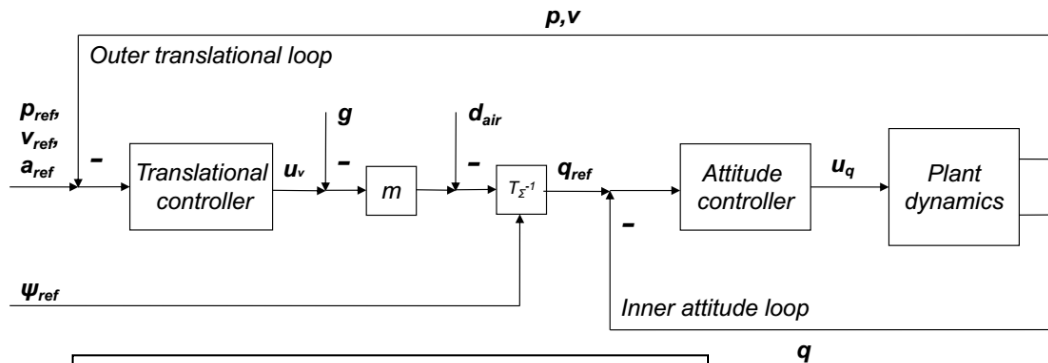
1. Not only limited jerk, but also limited acceleration and velocity.
2. Effectiveness for quadrotors is proved.



The jerk limited trajectory for a single axis



The jerk limited trajectory: why effective?



Inner loop constraints:

Limited body rate $\omega_{\max}$
and total thrust $f_{\max}$

Sufficient condition

Outer loop constraints:

The limited acceleration and jerk

$$\sqrt{\ddot{x}^2 + \ddot{y}^2 + \ddot{z}^2} \leq (\ddot{z}_{\min} + g)\omega_{\max}.$$

$$\|f\| = \sqrt{\ddot{x}^2 + \ddot{y}^2 + (\ddot{z} + g)^2} \leq \frac{f_{\max}}{m},$$

$$\ddot{z} \geq \ddot{z}_{\min} \geq \frac{f_{\min}}{m} - g,$$

Decoupling

Sufficient condition

Final constraints:

The limited acceleration and jerk

$$-v_{\max} \leq v(t) \leq v_{\max},$$

$$-a_{\max} \leq a(t) \leq a_{\max},$$

$$-u_{j_{\max}} \leq u_j(t) \leq u_{j_{\max}},$$

Limit the velocity for safety

Single axis constraints:

The limited acceleration and jerk

$$-a_{\max} \leq a(t) \leq a_{\max},$$

$$-u_{j_{\max}} \leq u_j(t) \leq u_{j_{\max}},$$



The jerk limited trajectory: compared with TOC

The jerk limited trajectory problem considers a system characterized by

$$\begin{pmatrix} \dot{p} \\ \dot{v} \\ \dot{a} \end{pmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} p \\ v \\ a \end{pmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_j$$

with $-v_{\max} \leq v(t) \leq v_{\max}$, $-a_{\max} \leq a(t) \leq a_{\max}$, $-u_{j\max} \leq u_j(t) \leq u_{j\max}$

The traditional time optimal control (TOC) or bang-bang control problem considers

$$\dot{x}(t) = Ax(t) + Bu(t) \quad (4.5)$$

with

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ a \end{bmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y \\ v \end{pmatrix} = \begin{pmatrix} y \\ \dot{y} \end{pmatrix} \quad (4.6)$$

Note that $v = \dot{y}$ is the velocity of the system. Let the control input be constrained as follows:

$$|u(t)| \leq u_{\max} \quad (4.7)$$



The jerk limited trajectory: velocity set-point

Problem:

From an arbitrary state to a desired velocity such that:

$$v(0) = v_0, \quad v(T) = v_{\text{ref}}$$

$$a(0) = a_0, \quad \mathbf{a}(T) = 0$$

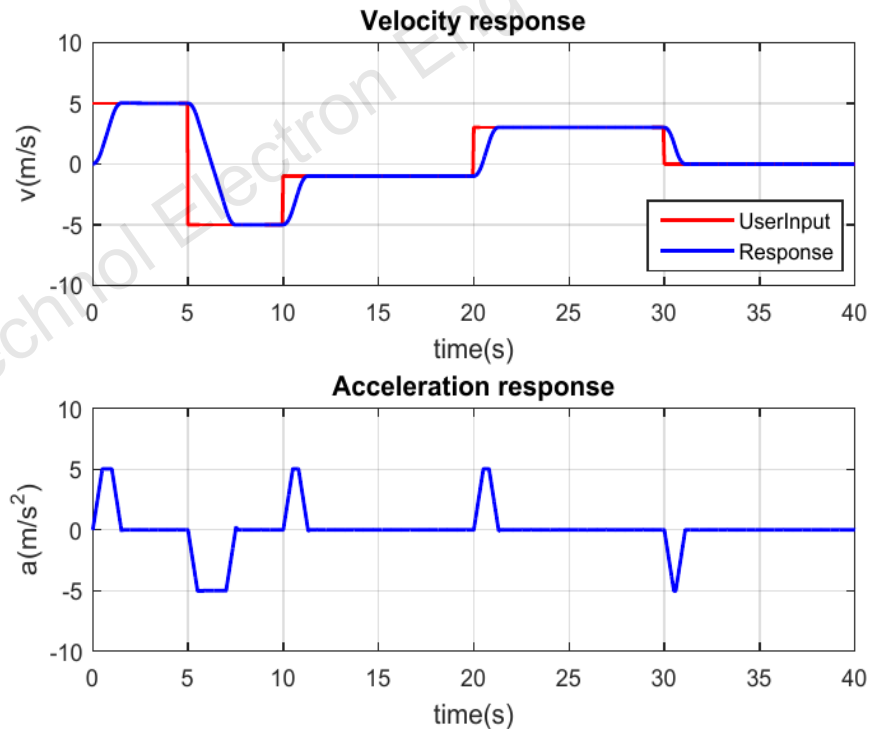
$$\dot{v}(t) = a(t)$$

$$\dot{a}(t) = u_j(t)$$

$$-a_{\text{max}} \leq a(t) \leq a_{\text{max}}, \quad \forall t \in [0, T]$$

$$-u_{j_{\text{max}}} \leq u_j(t) \leq u_{j_{\text{max}}}, \quad \forall t \in [0, T]$$

An example:



It is shown that the trajectory consists of at most three segments with $u = \pm u_{j_{\text{max}}}$ and 0.



The jerk limited trajectory: velocity set-point

Intuition on the solution:

Covered area (acc)= Δ Velocity

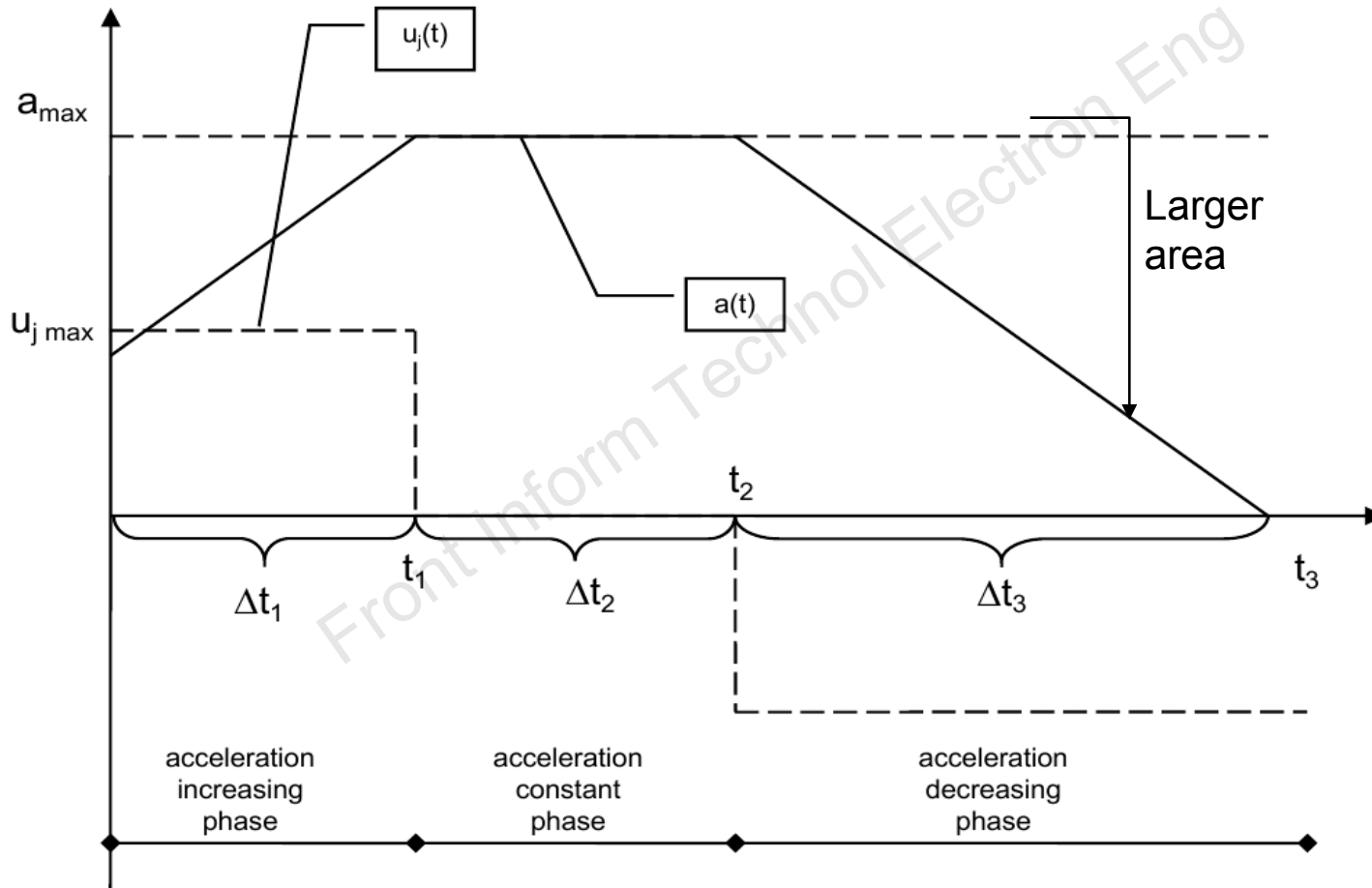


Figure 4.12: Trapezoidal acceleration profile



The jerk limited trajectory: position set-point

Problem:

From an arbitrary state to a desired point such that

$$p(0) = p_0, \quad p(T) = p_{\text{ref}}$$

$$v(0) = v_0, \quad v(T) = 0$$

$$a(0) = a_0, \quad a(T) = 0$$

$$\dot{p}(t) = v(t)$$

$$\dot{v}(t) = a(t)$$

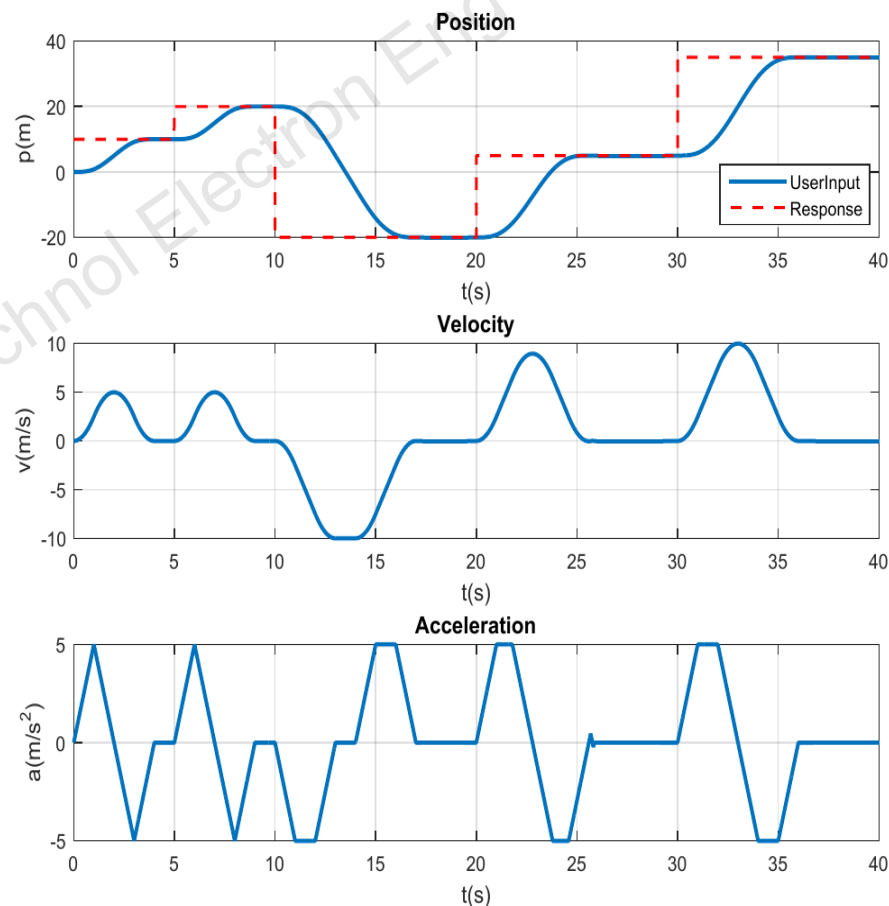
$$\dot{a}(t) = u_j(t)$$

$$-v_{\text{max}} \leq v(t) \leq v_{\text{max}}, \quad \forall t \in [0, T]$$

$$-a_{\text{max}} \leq a(t) \leq a_{\text{max}}, \quad \forall t \in [0, T]$$

$$-u_{j_{\text{max}}} \leq u_j(t) \leq u_{j_{\text{max}}}, \quad \forall t \in [0, T]$$

An example:



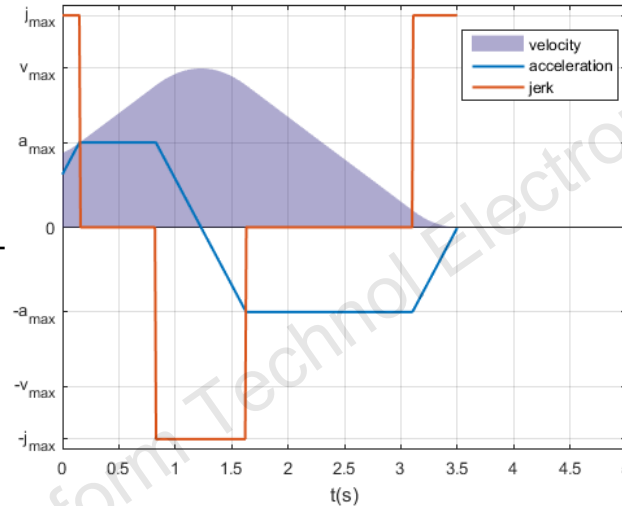


The jerk limited trajectory: position setpoint

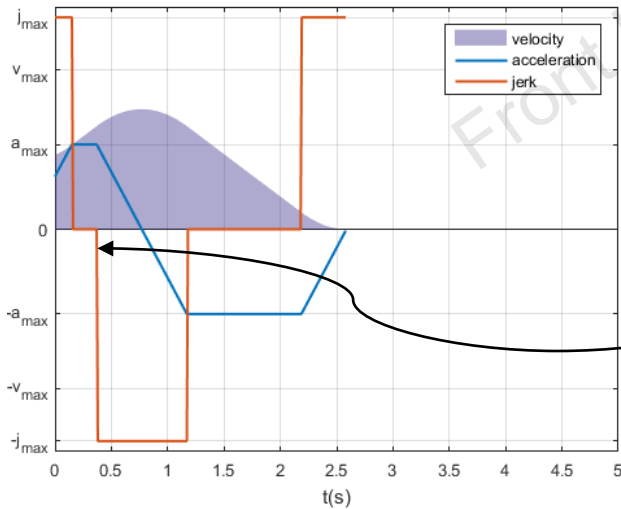
Intuition on the solution:

Covered area (vel) = Δ Position

Needing
smaller
area

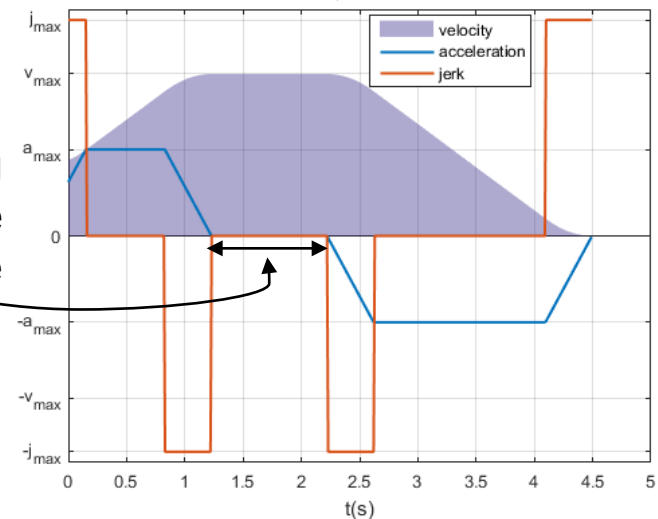


Needing
larger
area



Determining
switching
moment

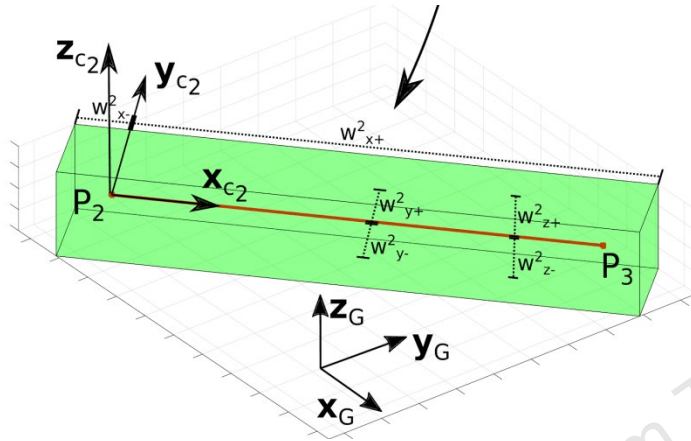
Determining
cruise
time





The jerk limited trajectory: trajectory along lines

For the trajectory to "converge" to a given line segment:

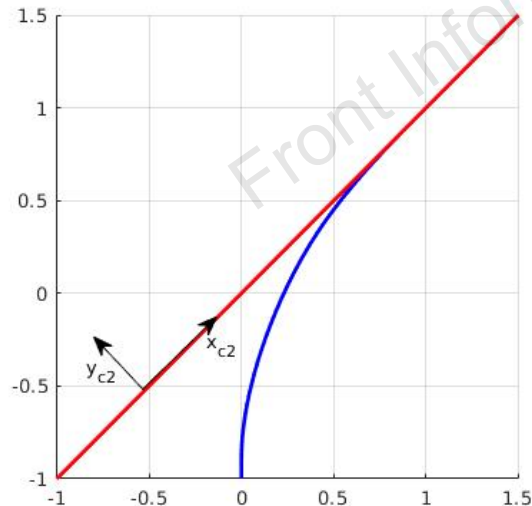


On the y and z axes:

Set the position set-point to zero.

On the x axis:

Set the position set-point to $|P_3 - P_2|$.



Result:

The trajectory converges to the desired line-segment path.



The jerk limited trajectory: how to control?

The robust and perfect tracking (RPT) controller:

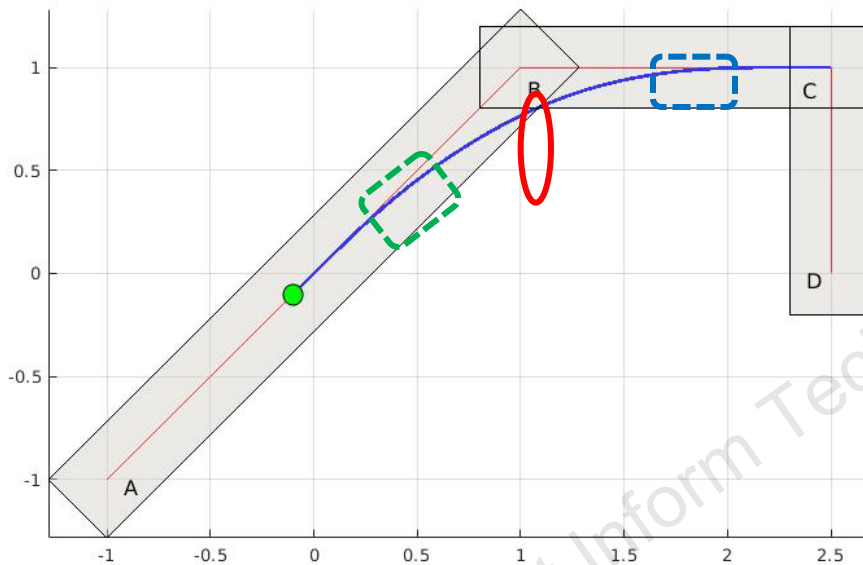
$$a_{x,n} = - \left[\frac{\omega_{n,x}^2}{\varepsilon_x^2} \quad \frac{2\zeta_x \omega_{n,x}}{\varepsilon_x} \right] \begin{pmatrix} x_n \\ u_n \end{pmatrix} + \left(\frac{\omega_{n,x}^2}{\varepsilon_x^2} \right) x_{n,r} + \left(\frac{2\zeta_x \omega_{n,x}}{\varepsilon_x} \right) u_{n,r} + a_{x,n,r}$$

$$a_{y,n} = - \left[\frac{\omega_{n,y}^2}{\varepsilon^2} \quad \frac{2\zeta_y \omega_{n,y}}{\varepsilon} \right] \begin{pmatrix} y_n \\ v_n \end{pmatrix} + \left(\frac{\omega_{n,y}^2}{\varepsilon^2} \right) y_{n,r} + \left(\frac{2\zeta_y \omega_{n,y}}{\varepsilon} \right) v_{n,r} + a_{y,n,r}$$

$$a_{z,n} = - \left[\frac{\omega_{n,z}^2}{\varepsilon^2} \quad \frac{2\zeta_z \omega_{n,z}}{\varepsilon} \right] \begin{pmatrix} z_n \\ w_n \end{pmatrix} + \left(\frac{\omega_{n,z}^2}{\varepsilon^2} \right) z_{n,r} + \left(\frac{2\zeta_z \omega_{n,z}}{\varepsilon} \right) w_{n,r} + a_{z,n,r}$$

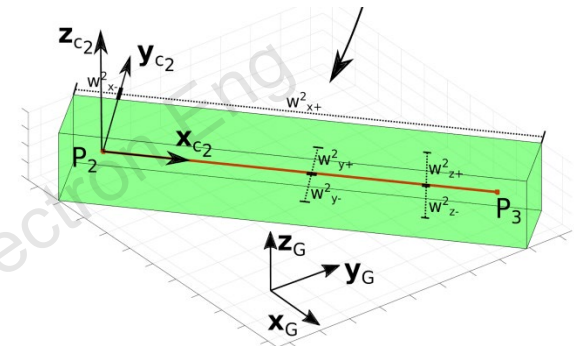


Safety check



Advantages:

1. Finite checking time regardless of the length of the corridor;
2. Guaranteed soundness.



Inside a single bounding box?

1. Project into the bounding-box frame;
2. Check the extreme value in individual axes.

Inside multiple bounding boxes?

1. Find split points;
2. Split the trajectory;
3. Check the split parts individually.

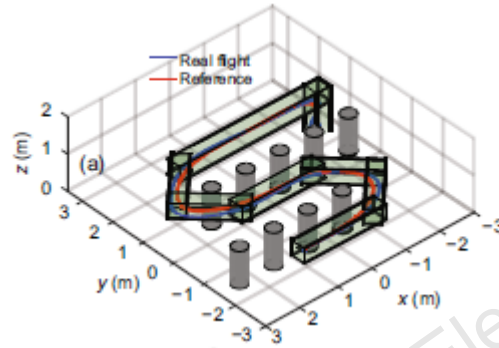


Major results

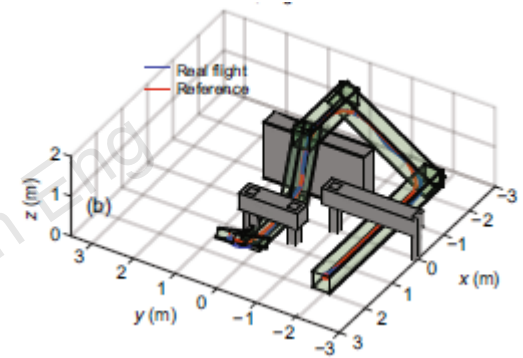
Experimental results



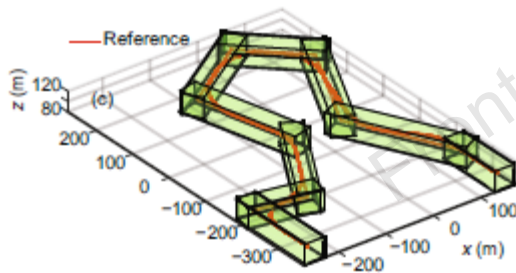
Fig. 14 Platform used for real flight experiments



Experiment A:
horizontal maneuver



Experiment B: vertical
maneuver



Experiment C: long distance fly
(simulation)

Table 2 State constraints

Experiment	$v_{h \max}$ (m/s ²)	$v_{v \max}$ (m/s)	$v_{v \min}$ (m/s)	$a_{h \max}$ (m/s)	$a_{v \max}$ (m/s ²)
A	4	0.8	-0.8	2.2	0.8
B	3	0.8	-0.8	1.2	0.8
C	15	0.8	-0.8	2.2	0.8

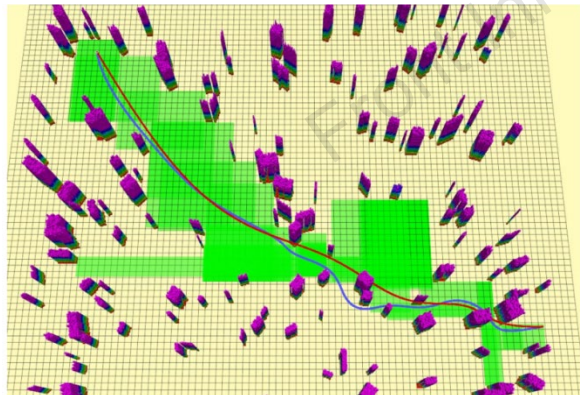
Experiment	$a_{v \min}$ (m/s ²)	$j_{h \max}$ (m/s ³)	$j_{v \max}$ (m/s ³)	$j_{v \min}$ (m/s ³)
A	-0.8	3	3	-3
B	-0.8	3	3	-3
C	-0.8	3	3	-3



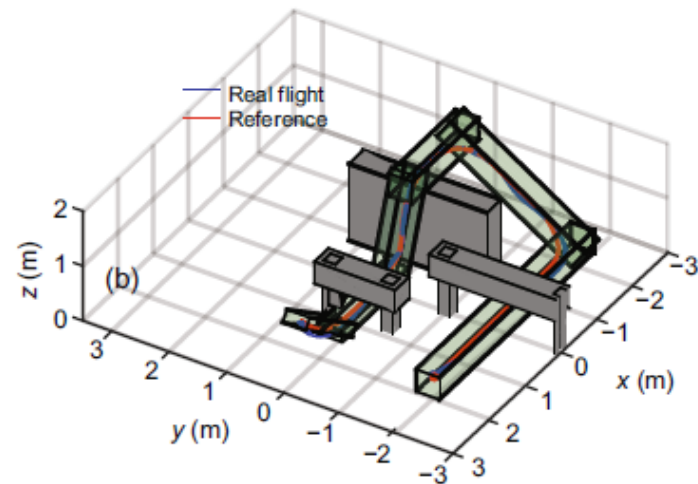
Experimental results: handling of constraints

Reference	Tilted corridor	State constraints	Soundness
Chen et al. (2016)	?	✓	✓
Liu et al. (2017)	✓	✓	?
Tang et al. (2016)	✓	?	✓
Shen et al. (2018)	?	✓	✓
Proposed method	✓	✓	✓

* All other methods, except the proposed one, use the spline approach, and require the solving of one or multiple non-linear programming problems.



Axes aligned corridor



Tilted corridor



Experimental results: computational efficiency

Our method

1. Incrementally generating the trajectory during execution;
2. For vehicle with low-power onboard PC;
3. Faster reaction to changes.

Table 2 Time consumption: smooth navigation

Total time	Trajectory generation	Checking
3.91 us	2.68 us	0.9 us

Spline-based SFC method

Generating the entire trajectory before execution.



Concluding remarks

1. We have proposed a rather down-to-earth solution for navigating a quadrotor along the nominal path plan
 - (1) safely, (2) smoothly, and (3) efficiently,with its physical constraints on velocity, acceleration, and jerk.
2. The proposed approach
 - (1) saves flying time, and (2) is computationally efficient.



References

Chen J, Liu T, Shen S, 2016. Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments. IEEE Int Conf on Robotics and Automation, p.1476-1483.

Liu S, Watterson M, Mohta K, 2017. Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments. *IEEE Rob Autom Lett*, 2(3):1688-1695.

Tang S, Kumar V, 2016. Safe and complete trajectory generation for robot teams with higher-order dynamics. IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.1894-1901.

Gao F, Wu W, Lin Y, 2018. Online safe trajectory generation for quadrotors using fast marching method and Bernstein basis polynomial. IEEE Int Conf on Robotics and Automation.