Mechatronics 30 (2015) 65–75

Contents lists available at ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

Systems design and implementation with jerk-optimized trajectory generation for UAV calligraphy

Swee King Phang [a,*], Shupeng Lai [a], Fei Wang [b], Menglu Lan [c], Ben M. Chen [c]

[a] NUS Graduate School for Integrative Sciences & Engineering, National University of Singapore, 21 Lower Kent Ridge Rd, 119077 Singapore, Singapore
[b] Temasek Laboratories, National University of Singapore, 21 Lower Kent Ridge Rd, 119077 Singapore, Singapore
[c] Department of Electrical & Computer Engineering, National University of Singapore, 21 Lower Kent Ridge Rd, 119077 Singapore, Singapore

ARTICLE INFO

Article history:
Received 16 September 2014
Accepted 10 June 2015
Available online 26 June 2015

Keywords:
Mechatronics systems design
Unmanned aerial vehicles
UAV calligraphy
Trajectory optimization

ABSTRACT

Unmanned aerial vehicle (UAV) and Chinese calligraphy seem like two completely unrelated subjects in today's world. UAV, as one of the most advanced technology to date, has gathered much attention lately due to its potentially unlimited applications. Contrarily, Chinese calligraphy is one of the most beautiful and ancient calligraphy art originally developed from China few thousand years ago. Today in this manuscript, we present to you the art of autonomous calligraphy writing with UAVs. The proposed UAV calligraphy system is able to trace the user handwritten inputs, and then execute the writing by mimicking the user handwriting with four autonomous UAVs. This manuscript details the design considerations and implementation process of such a system. The UAV calligraphy system was performed in Singapore Airshow 2014. Robustness and reliability of the system has been well tested, and high performance can be seen from the resulting calligraphy writing.

1. Introduction

Driven by the advancement in processing power of tiny micro-processor, the unmanned aerial vehicle (UAV) research has reached a new horizon where they are getting smaller in size but smarter. Many researchers have shifted their attention from the usual UAV to small scale or miniature UAV development [20,21,24]. Due to their small size and light weight, these UAVs are capable of maneuvering indoors for the various missions or tasks. In particular, UAVs of quadrotor platforms are most frequently used by the researchers, due to their simplicity in structure and scalability. In general, a strong micro-processor and an inertial measurement unit (IMU) are needed for orientation control of a quadrotor, while GPS localization is used for outdoor position control. In the environment where GPS signal is not available, e.g. indoor environment, a different localization method will be needed. Visual based navigation was introduced in [9,11,12] to estimate the relative distance of the UAV from its original position, while the researchers from the University of Pennsylvania were one of the first few group to utilize the Vicon motion tracking system to measure the UAV in a confined space [23]. Once the localization issue of the indoor UAV is solved, control and navigation problem such as trajectory generation can be handled [6,7,13,14,17,22,39,40].

Chinese calligraphy is one of the finest of all Chinese traditional arts. It is an inseparable part of Chinese history, and its delicate aesthetic appreciation are commonly considered to be unique among all calligraphic arts [34]. In this manuscript, we proposed to combine this traditional Chinese art together with the modern development of UAV, to perform what we called UAV calligraphy (see Fig. 1). Prior to our development, many researchers have investigated the generation of strokes of Chinese characters in simulation [32,35,38]. In the work documented in [33], the authors visually analyzed and then classified the Chinese calligraphy characters. To the best of our knowledge, there are, however, no research or successful example of calligraphy writing with any aircraft or flying machine. The closest example sees the development of an omnidirectional ground vehicle to perform calligraphy writing [10], while most of the development in automated calligraphy writing was realized with robotic arms [5,18,19,31,37].

We have identified a few challenges in realizing autonomous UAV calligraphy writing. In most of the applications documented in the literature, 6 degree-of-freedom (DOF) robotic arm is used to execute the writing. They are able to trace simple B-spline optimized trajectories [36]. In our work, we need to incorporate the UAV's dynamics in path generating and thus it increases computational complexity and loads to the calligraphy writing system. As

* Corresponding author. Tel.: +65 97789556.
E-mail addresses: king@u.nus.edu (S.K. Phang), shupenglai@u.nus.edu (S. Lai), tslwf@nus.edu.sg (F. Wang), lanmenglu@u.nus.edu (M. Lan), bmchen@nus.edu.sg (B.M. Chen).

http://dx.doi.org/10.1016/j.mechatronics.2015.06.006
0957-4158/

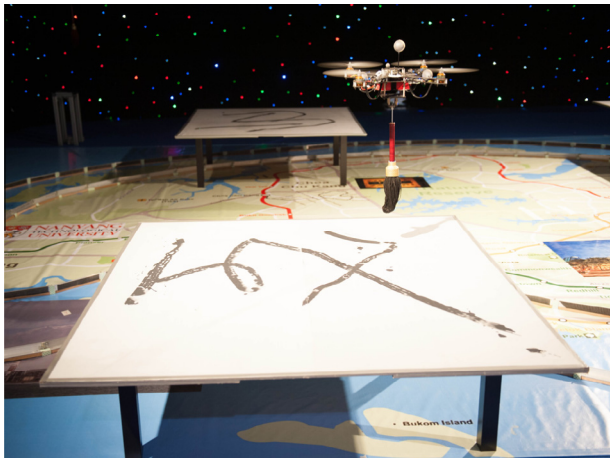


Fig. 1. UAV calligraphy performance in Singapore Airshow.

the airborne UAV must always remain upright, we thus consider only 3 translational DOF in executing the calligraphy writing. As we wish the calligraphy writing system works on the handwriting of user input, the second challenge occurs in decoding of user handwriting to the information recognized by the machine. Besides a simple graphical interface for user handwritten input (see Fig. 2), a sophisticated algorithm is needed to extract important turning points of the handwritten character. Lastly, the mechanical design of the calligraphy brush and its connector to the UAV poses an important challenge to us. Unlike the robotic arm, the force applying to the calligraphy brush during writing will be reflected back to the airborne UAV, and thus affects its stability. A sophisticated UAV control scheme needs to be revised, together with a creative design of the calligraphy brush to make the system work with as little disturbance as possible.

The implemented UAV calligraphy system is first introduced to the public in Singapore Airshow 2014 [27]. Singapore Airshow is one of the world's largest aviation event held biyearly in Singapore. It has attracted exhibitors from more than 50 countries to participate and to showcase their development in aviation sector. Our team from the National University of Singapore, has realized four UAVs writing four different Chinese characters simultaneously. The handwriting tracing system is also proven to work well as the public handwritten input is sketched by the UAVs on the spot.

This manuscript documents the design architecture and the implementation procedure of the UAV calligraphy system. Section 2 details the design and implementation of the hardware needed to realized UAV calligraphy. Controls and implementation on the UAV will be discussed in Section 3. Sections 4 and 5 shows the handwritten character strokes tracing and trajectory generating, which makes the core content of this manuscript. Results of the UAV writing calligraphy will be shown in Section 6, while Section 7 gives concluding remarks of our work.

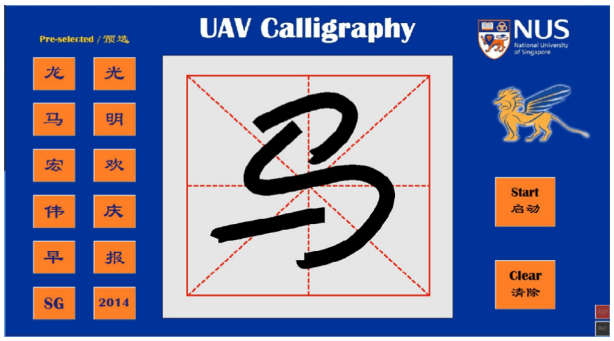


Fig. 2. Graphical interface for user handwriting input.

2. Hardware setup

The UAV calligraphy system is able to work on any miniature self-stabilized UAV in general. In our project realization, quadrotor UAVs with high orientation control bandwidth are utilized. Each quadrotor has four propellers of 10 inches in diameter and has the largest dimension of 40 cm from motor to motor. It is approximately 1 kg and has a high inner-loop bandwidth of 25 rad/s. The high bandwidth enables a fast tracking outer-loop controller to be designed, as will be shown in Section 3 later.

To realize calligraphy writing, a calligraphy brush together with the holding mechanism is customized. The full overview design of the brush can be visualized in Fig. 3. More specifically, we wish to highlight an important design consideration to realize UAV calligraphy – the linear bearing joint (zoomed view in Fig. 3). A linear bearing is included at the base of the brush holder, while a shaft attached to the brush passes through and is locked to the bearing. The installation of such a mechanism enables the following two points.

1. A free low-friction linear movement along z-direction of the brush with reference to the UAV above it. This reduces the disturbance to the airborne UAV resulted from the contact between the calligraphy brush and the writing board. Performance of the UAV will thus not be affected during the contact-writing instances.
2. The calligraphy brush can be rotated freely along z-axis, resulting smooth and natural writing along an arc or circular drawing.

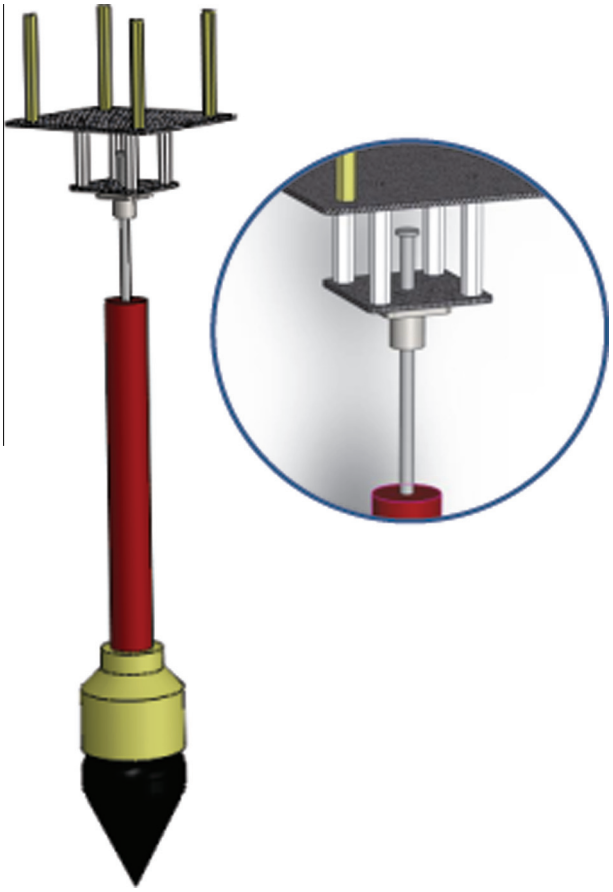


Fig. 3. The designed calligraphy brush and its holder.

Next, an accurate position sensor is needed for each quadrotor to track the generated trajectory precisely. Here in this UAV calligraphy project, the accuracy of the UAV position measurement is the prime factor affecting the writing result. As the UAVs will write in a controlled environment, i.e., an environment we could set up for the purpose of performing the UAV calligraphy, an external source of accurate object position estimation is used. A Vicon motion sensing system is set up for this purpose. Working principle of the system has been previously documented in [28]. In this project, a total of 36 Vicon cameras are installed to the system to provide object position estimation with a resolution up to 0.001 mm. With such accuracy, the performance of the UAV calligraphy boils down to its trajectory tracking and position control accuracy, which will be discussed in Section 3.

3. UAV control system

For UAV calligraphy, flight performance of the controlled platform is the main concern to be addressed. Otherwise, there is no foundation for the higher level trajectory planning algorithm to be built upon. Here, the UAV control problem is decomposed into two layers, namely the attitude stabilization layer and the position tracking layer (see Fig. 4). The former involves the design of an inner-loop controller which makes sure the UAV roll, pitch and yaw dynamics are robustly stable. The latter position tracking layer involves the design of an outer-loop controller which enables the UAV to track any smooth 3-D trajectory in a responsive and precise way. To guarantee the desired flight performance, the outer-loop bandwidth should be designed low enough with respect to the closed inner-loop dynamics.

It can be seen from Fig. 4 that the output from the outer-loop controller is the acceleration commands in the global frame $\mathbf{a}_c$, while the inner-loop controller needs the input from the attitude references ϕ_c, θ_c, ψ_c. To properly connect the two blocks, a global-to-body rotation followed by a command conversion is needed. Moreover, the body-axis acceleration command $\mathbf{a}_b$ does not contribute to the heading direction reference ψ_c. Therefore, unlike the other two attitude angle references, the yaw angle reference ψ_c is not involved in this conversion, thus will be generated independently. In addition, the acceleration reference in the UAV body z-axis is controlled directly with the throttle control input δ_{thr}, which is independent from the inner loop. Based on the above ideas, we assign $\mathbf{G}_c$ as the steady-state gain matrix from the UAV body-frame accelerations to the inputs $(\delta_{thr}, \phi_c, \theta_c)$ as

$$(\delta_{thr} \quad \phi_c \quad \theta_c)^T = \mathbf{G}_c \mathbf{a}_b, \tag{1}$$

where

$$\mathbf{G}_c = \begin{bmatrix} 0 & 0 & 0.0334 \\ 0 & 0.1019 & 0 \\ -0.1019 & 0 & 0 \end{bmatrix}. \tag{2}$$

In this work, the quadrotor inner-loop controller is implemented onboard with a PX4FMU multi-rotor controller developed by PIXHAWK ETH. It is an all-in-one open source electronic board capable of sensing, filtering, processing and servo driving. By following the software framework of this board, a simple inner-loop controller is implemented and tuned towards fast closed-loop dynamics. As large amounts of work about quadrotor stability control have been published in literature, we will not elaborate it again in detail here.

However, the design of the outer-loop controller is critical for the application of UAV calligraphy due to the stringent requirements on position tracking. Unlike many other UAV applications in which only the steady-state position tracking performance is to be ensured while a loose transient tracking is acceptable, UAV calligraphy needs the actual position of the UAV to track the reference trajectory at every instance as close as possible. The robust and perfect tracking (RPT) control concept from [4] perfectly fits this requirement. Theoretically, a system controlled by the RPT method is able to track any given reference with arbitrarily fast settling time subjected to disturbances and initial conditions. The basic design idea of RPT controller is as follows. For a linear time invariant system

$$\Sigma = \begin{cases} \dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u} + E\boldsymbol{w} \\ \boldsymbol{y} = C_1\boldsymbol{x} + D_1\boldsymbol{w} \\ \boldsymbol{h} = C_2\boldsymbol{x} + D_2\boldsymbol{u} + D_{22}\boldsymbol{w} \end{cases} \tag{3}$$

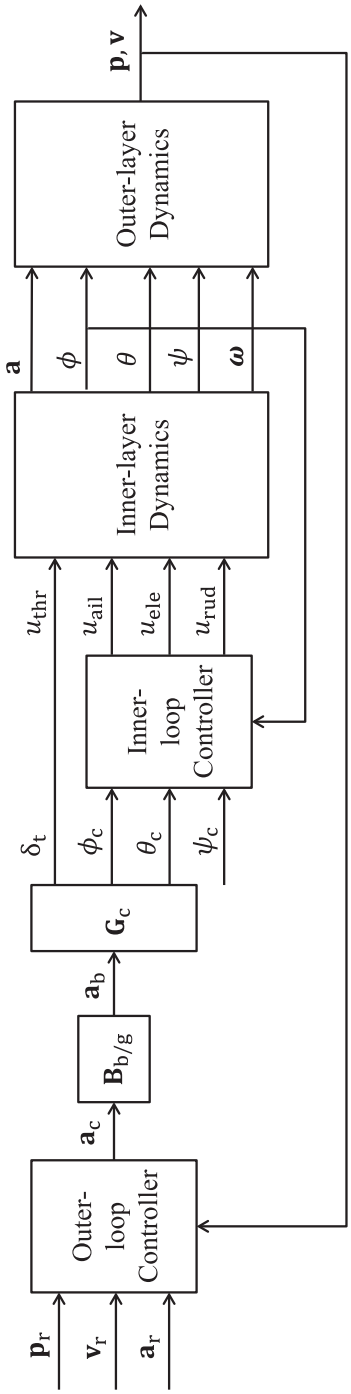


Fig. 4. Dual-loop control structure of the quadrotor.

with $\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w}, \boldsymbol{y}, \boldsymbol{h}$ being the state, control input, disturbance, measurement and controlled output respectively, the task of RPT controller is to formulate a dynamic measurement control law of the form

$$\dot{\boldsymbol{v}} = A_c(\varepsilon)\boldsymbol{v} + B_c(\varepsilon)\boldsymbol{y} + G_0(\varepsilon)r + \cdots + G_{\kappa-1}(\varepsilon)r^{\kappa-1},$$
$$\boldsymbol{u} = C_c(\varepsilon)\boldsymbol{v} + D_c(\varepsilon)\boldsymbol{y} + H_0(\varepsilon)r + \cdots + H_{\kappa-1}(\varepsilon)r^{\kappa-1},$$

where $\boldsymbol{v}$ being the controller state, so that when a proper $\varepsilon > 0$ is chosen,

1. The resulted closed-loop system is asymptotically stable subjected to zero reference.
2. If $e(t, \varepsilon)$ is the tracking error, then for any initial condition $\boldsymbol{x}_0$, there exists:

$$\|e\|_p = \left(\int_0^\infty |e(t)^p| dt\right)^{1/p} \to 0, \quad \text{as} \quad \varepsilon \to 0. \tag{4}$$

For non-zero references, their derivatives are used to generate additional control inputs. Thus, any references of the form $r(t) = p_1 t^k + p_2 t^{k-1} + \cdots + p_{k+1}$ are covered in the RPT formulation. Furthermore, any references that have a Taylor series expansion at $t = 0$ can also be tracked using the RPT controller.

Similar to the case introduced in [22], the outer dynamics of our quadrotor UAV is differentially flat, meaning all its state variables and inputs can be expressed in terms of algebraic functions of flat outputs and their derivatives. A proper choice of flat outputs is

$$\sigma = [x, y, z, \psi]^{\mathrm{T}}. \tag{5}$$

It can be observed that the first three outputs, x, y, z, are totally independent. In other words, we can consider the UAV as a mass point with constrained velocity, acceleration and its higher derivatives in the individual axis of the 3-D global frame when designing its outer-loop control law and generating the position references. Hence, a stand-alone RPT controller based on multiple-layer integrator model in each axis can be designed to track the corresponding reference in that axis. For each axis, the nominal system can be written as

$$\dot{\boldsymbol{x}}_{\mathrm{n}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{x}_{\mathrm{n}} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_{\mathrm{n}}, \quad \boldsymbol{y}_{\mathrm{n}} = \boldsymbol{x}_{\mathrm{n}} \tag{6}$$

where $\boldsymbol{x}_{\mathrm{n}}$ contains the position and velocity state variables and u_{n} is the desired acceleration.

To achieve a good tracking performance, it is common to include an error integral to ensure zero steady-state error. This requires an augmented system to be formulated as

$$\begin{cases} \dot{\boldsymbol{x}}_{\mathrm{aug}} = \begin{bmatrix} 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}_{\mathrm{aug}} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_{\mathrm{aug}} \\ \boldsymbol{y}_{\mathrm{aug}} = \boldsymbol{x}_{\mathrm{aug}} \\ h_{\mathrm{aug}} = [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]\boldsymbol{x}_{\mathrm{aug}} \end{cases} \tag{7}$$

where $\boldsymbol{x}_{\mathrm{aug}} = \left[\int(\mathbf{p}_{\mathrm{e}}) \quad \mathbf{p}_{\mathrm{r}} \quad \mathbf{v}_{\mathrm{r}} \quad \mathbf{a}_{\mathrm{r}} \quad \mathbf{p} \quad \mathbf{v}\right]^{\mathrm{T}}$ with $\mathbf{p}_{\mathrm{r}}, \mathbf{v}_{\mathrm{r}}, \mathbf{a}_{\mathrm{r}}$ as the position, velocity and acceleration references in the controlled axis, $\mathbf{p}, \mathbf{v}$ as the actual position and velocity and $\mathbf{p}_{\mathrm{e}} = \mathbf{p}_{\mathrm{r}} - \mathbf{p}$ as the tracking error of the position. By following the procedures in [3], a linear feedback control law of the following form can be acquired:

$$u_{\mathrm{aug}} = F_{\mathrm{aug}} \boldsymbol{x}_{\mathrm{aug}}, \tag{8}$$

where

$$F_{\mathrm{aug}} = \left[\frac{k_i\omega_n^2}{\varepsilon^3} \quad \frac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} \quad \frac{2\zeta\omega_n + k_i}{\varepsilon} \quad 1 \quad -\frac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} \quad -\frac{2\zeta\omega_n + k_i}{\varepsilon}\right].$$

Here, ε is a design parameter to adjust the settling time of the closed-loop system. ω_n, ζ, k_i are the parameters that determine the desired pole locations of the infinite zero structure of (7) through

$$p_i(s) = (s + k_i)(s^2 + 2\zeta\omega_n s + \omega_n^2). \tag{9}$$

Theoretically, when the design parameter ε is small enough, the RPT controller can give arbitrarily fast responses. However, in real life, due to the constraints of the UAV physical dynamics and its inner-loop bandwidth, it is safer to limit the bandwidth of the outer loop to be much smaller than that of the inner-loop dynamics. For our case, the following design parameters are used:

$$x, y: \begin{cases} \varepsilon = 1 \\ \omega_n = 1.772 \\ \zeta = 0.5 \\ k_{\mathrm{i}} = 0.2 \end{cases}, \quad z: \begin{cases} \varepsilon = 1 \\ \omega_n = 2.089 \\ \zeta = 0.66 \\ k_{\mathrm{i}} = 0.23 \end{cases}$$

4. Handwriting extractions

In order to perform regression and rebuild a flyable reference trajectory, the original user input is sampled by finding the more *meaningful* points. Here, the *meaningful* points refer to the starting, ending and turning points where they usually form the skeleton of the handwriting. To determine the turning points in a sequenced 2-D points set, a split-and-merge algorithm is applied to divide these 2-D points into individual line segments. The turning points will then be assigned to the endpoints of these line segments [2]. The original sequenced 2-D points set is acquired by recording user's handwritten input through a tablet via an interface application we have created. They are sorted in the chronological order and sent to the ground station for turning point extractions.

The algorithm is illustrated in Fig. 5. The sequence of the split-and-merge algorithm is as follows:

1. Connect the first point A and the last point B.
2. Find point C among all data points that has the longest perpendicular distance to line A–B.
3. If this longest distance is within a threshold, then a cluster is created containing points between A and B.
4. Else, the input points will be split into two sub-groups, A–C and C–B. For each sub-group, the split-and-merge algorithm will be called recursively.
5. The algorithm stops when all longest distance fall inside the preset threshold.

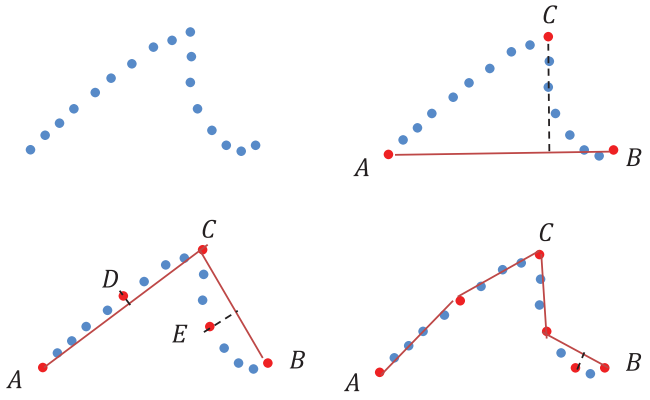


Fig. 5. Split-and-merge sequence on continuous line segments.

Finally, the algorithm will return all the endpoints of the resulted point-sub-groups in their original chronological order. All these endpoints are used to run a B-spline based regression to generate a reference trajectory under the constraints of UAV dynamic.

5. Minimum jerk trajectory planning

In order to formulate a trajectory representing the user's input that could be tracked precisely, it is necessary to consider the vehicle's dynamic. As shown in the previous section, the quadrotor is a differential flat system with the flat outputs:

$$\sigma = [x, y, z, \psi]^{\mathrm{T}}, \tag{10}$$

where $p = [x, y, z]$ denote the position of the center of mass and the ψ is the heading angle of the vehicle. Specifically, for our application, there is no need to alternate the heading angle at all time. Therefore, the smooth trajectory of the UAV could be represented by the sub-space of the flat outputs with a constant heading angle given by

$$p(t) : [t_{\mathrm{ini}}, t_{\mathrm{end}}] \longrightarrow \mathbb{R}^3, \tag{11}$$

where t_{ini} and t_{end} denote the initial and final time of the desired trajectory. The trajectory requires at least C_2 smoothness as its feasibility as input to the vehicle is determined by its derivatives of different orders. As a differential flat system, it is sufficient to limit the maximum acceleration or thrust of the vehicle by considering the dynamic of the UAV in generating a reference path. According to the work in [25], the feasibility condition of the vehicle's thrust is constrained by

$$\max\left\{\sum_{i=1}^{3}(\ddot{p}_i(t) - g_i)^2\right\} \leqslant f_{\max}^2, \quad \forall t \in [t_{\mathrm{ini}}, t_{\mathrm{end}}], \tag{12}$$

where $p_1 = x, p_2 = y, p_3 = z, g$ is gravity vector and $f_{\max}$ is the maximum allowable thrust.

For the application on UAV calligraphy, aggressive maneuvering is not necessary. The constraint on vehicle's thrust is thus split into three individual axis to enhance the computation speed. The constraint can be simplified to

$$\max\left\{(\ddot{p}_i(t) - g_i)^2\right\} \leqslant \alpha_i^2, \quad \forall t \in [t_{\mathrm{ini}}, t_{\mathrm{end}}], \tag{13}$$

where $i \in \{1, 2, 3\}$ and α_i satisfies

$$\sum_{i=1}^{3}\alpha_i^2 \leqslant f_{\max}^2. \tag{14}$$

It has been proven in [22] that the upper bound of the norm of the vehicle's body angular rates is proportional to the norm of trajectory's jerk (derivative of the acceleration) as

$$\bar{\omega}^2 = \frac{\sum_{i=1}^{3}\max_{t\in[t_{\mathrm{ini}},t_{\mathrm{end}}]} j_i(t)^2}{\sum_{i=1}^{3}\min_{t\in[t_{\mathrm{ini}},t_{\mathrm{end}}]}(\ddot{p}_i(t) - g_i)^2}, \tag{15}$$

where j_1, j_2, and j_3 are the corresponding jerk of the trajectory in x-, y- and z-axis. In order to achieve a smooth response from the vehicle, a minimum jerk trajectory is naturally considered. It is further supported by the research work in [8], in which it is proven that human reaching trajectory is indeed a minimum jerk trajectory. It is thus reasonable to allow the vehicle to repeat this behavior during its writing process.

The trajectory planning of the UAV calligraphy system is strongly based on the basis mentioned above. In order to realize the trajectory generation for such system, we have divided the problem into three different parts. Starting with the normalization of the uniform B-spline, followed by two approaches to solve the minimum jerk trajectory problem – a closed solution and the more practical quadratic programming solution. Then, an optimal time segmentation algorithm is introduced to further optimize the flight time of the UAV. Both the minimum jerk trajectory and the flight time of the UAV will be optimized iteratively, until a converged local minimal solution is reached. The final generated flight path will then be the converged result of the iterations.

5.1. Normalized uniform B-spline

The normalized uniform B-splines is named after its normalized and hence equally segmented knots vector. The base of the B-splines has been defined in [1] as a recursive function

$$N_{i,0}(u) = \begin{cases} 1, & \text{if } u_i \leqslant u < u_{i+1}, \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i}N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}}N_{i+1,p-1}(u), \tag{17}$$

where $N_{i,p}(u)$ is the basis function of the generally defined B-splines and $[u_0, u_1, u_2, \ldots]$ forms the knot vector of B-splines. Normalization of the uniform B-spline will thus produce the knot vector $[0, 1, 2, 3, \ldots]$. A basis element function of a normalized uniform B-splines has been given in [30] as

$$T_{j,i}(s) = \begin{cases} 1, & \text{if } i = j = 0, \\ \frac{1-s}{i}T_{0,i-1}(s), & \text{if } j = 0, i \neq 0, \\ \frac{s}{i}T_{i-1,i-1}(s), & \text{if } j = i > 0, \\ \frac{i-j+s}{i}T_{j-1,i-1}(s) + \frac{1+j-s}{i}T_{j,i-1}(s), & \text{if } j = 1, \ldots, i-1, \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

Thus, a spline function can then be expressed as

$$S_k(t) = \sum_{i=1}^{M} c_i B_k(t - i + 1), \quad c_i \in \mathbb{R}^3, \tag{19}$$

where

$$B_k(s) = \begin{cases} T_{k-j,k}(s - j), & j \leqslant s < j + 1, \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

Here, $j = 0, \ldots, k$ where k denotes the order of the spline, and M denotes the number of user specified control points. Notice that in general path optimization, c_i are the trajectory points in 3-D space to be optimized later. As we have broken down the trajectory to its components along x-, y- and z-directions, c_i becomes a scalar in our problem formulation, i.e., $c_i \in \mathbb{R}$. Also, as the trajectory references fed to the outer-loop RPT controller mentioned in the previous section is up to the second derivatives of the positions (the accelerations), we can specifically formulate this optimization problem in third order, i.e. $k = 3$.

In order to arbitrarily specify the boundary conditions in our system, a cubic clamped normalized uniform B-spline is proposed. This customized B-spline has a knot vector in the form of $[0, 0, 0, 0, 1, 2, 3, \ldots, m-1, m, m, m, m]$, where the first three and the last three bases corresponding to the vector elements of 0 and m are for the initial and the final conditions of the desired trajectory. Note that in our case, $m = M - 1$, and thus the knot vector has $M + 6$ terms. From Eq. (16), we could easily deduce the different basis accordingly. The expression of common basis element $B_3(u)$ is given by

$$B_3(u) = \begin{cases} \frac{1}{6}u^3, & \text{if } u \in [0,1), \\ \frac{1}{6} - \frac{1}{2}(u-1)^3 + \frac{1}{2}(u-1)^2 + \frac{1}{2}(u-1), & \text{if } u \in [1,2), \\ \frac{1}{6} + \frac{1}{2}(u-3)^3 + \frac{1}{2}(u-3)^2 - \frac{1}{2}(u-3), & \text{if } u \in [2,3), \\ -\frac{1}{6}(u-4)^3, & \text{if } u \in [3,4), \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

whereas the initial three and the final three bases can be uniquely expressed as

$$N_{0,3}(u)=\begin{cases}(1-u)^3, & \text{if } u\in[0,1),\\ 0, & \text{otherwise.}\end{cases} \tag{22}$$

$$N_{1,3}(u)=\begin{cases}u(1-u)^2+\frac{u(4-3u)(2-u)}{8}, & \text{if } u\in[0,1),\\ -\frac{(u-2)^3}{4}, & \text{if } u\in[1,2),\\ 0, & \text{otherwise.}\end{cases} \tag{23}$$

$$N_{2,3}(u)=\begin{cases}\frac{u^2(-3u+4)}{4}+\frac{u^2(3-u)}{6}, & \text{if } u\in[0,1),\\ \frac{u(u-2)^2}{4}+\frac{(3-u)(-2u^2+6u-3)}{6}, & \text{if } u\in[1,2),\\ \frac{u^2(-3u+4)}{4}+\frac{u^2(3-u)}{6}, & \text{if } u\in[2,3),\\ 0, & \text{otherwise.}\end{cases} \tag{24}$$

and

$$N_{M+3,3}(u)=N_{2,3}(-u+M-1), \tag{25}$$
$$N_{M+4,3}(u)=N_{1,3}(-u+M-1), \tag{26}$$
$$N_{M+5,3}(u)=N_{0,3}(-u+M-1). \tag{27}$$

Finally, we can express the overall cubic spline in

$$S_3(t)=\sum_{i=-2}^{0}c_iN_{i+2,3}(t)+\sum_{i=1}^{M}c_iB_3(t-i+1)+\sum_{i=M+1}^{M+3}c_iN_{i+2,3}(t),\quad c_i\in\mathbb{R}, \tag{28}$$

for each individual channel in x-, y- and z-direction.

Now, there are $M+6$ control points where the first three and last three are used to determine the boundary conditions. Thus in general, if a new basis for the clamped spline is defined, the cubic spline can be expressed as

$$S_3(t)=\sum_{i=0}^{M+5}\tau_iF_{i,3}(t),\quad \tau_i\in\mathbb{R}, \tag{29}$$

where

$$\tau_i=c_{i-2},$$
$$F_{i,3}(t)=\begin{cases}N_{i,3}(t), & \text{if } i\in\{0,1,2,M+3,M+4,M+5\},\\ B_3(t-i+3), & \text{otherwise.}\end{cases} \tag{30}$$

5.2. Minimum jerk trajectory: closed solution

We can now formulate the problem of the minimum jerk trajectory based on the clamped normalized uniform B-spline. The formulation of the problem is similar to the documented works in [22,29] but with 6 additional clamped bases as the initial and the final conditions of the trajectory. If a set of sampled data points is given as

$$D=\{d_i\in\mathbb{R}:i=1,\dots,N\},$$
$$T=\{t_i\in\mathbb{R}:i=1,\dots,N\}, \tag{31}$$

where D is the set of 1-D trajectory data points, T is the set of time indicating at what time each of the above data points are reached, and N is the total number of trajectory points to be optimized, then, the minimum jerk trajectory will be achieved by minimizing the criterion function

$$J=w_g\int_{-\infty}^{\infty}v^2(t)dt+\sum_{i=1}^{N}(S_3(t_i)-d_i)^2, \tag{32}$$

where w_g is a weighting factor, and

$$v(t)=\sum_{i=0}^{M+5}\frac{d^3}{dt^3}\tau_iF_{i,3}(t),\quad \tau_i\in\mathbb{R}, \tag{33}$$

is the third derivative of Eq. (29).

It is further shown in [16] that $\int_{-\infty}^{\infty}v^2(t)dt$ can be expressed as the form of $\tau^TG\tau$, where $\tau=[\tau_0,\tau_1,\dots,\tau_{M+5}]^{\mathrm{T}}$, and G can be calculated explicitly. More specifically, the elements in G are

$$g_{i,j}=\alpha^5\int_{-\infty}^{\infty}F_{i,3}^{(3)}(t)F_{j,3}^{(3)}(t)dt, \tag{34}$$

where $\alpha=(M+6)/T_{\text{true}}$ and $T_{\text{true}}=t_{\text{end}}-t_{\text{ini}}$ is the total time of the trajectory. On the other hand, by letting

$$H=\begin{bmatrix}F_{0,3}(\alpha t_1) & F_{1,3}(\alpha t_1) & \cdots & F_{M+5,3}(\alpha t_1)\\ F_{0,3}(\alpha t_2) & F_{1,3}(\alpha t_2) & \cdots & F_{M+5,3}(\alpha t_2)\\ \vdots & \vdots & \ddots & \vdots\\ F_{0,3}(\alpha t_N) & F_{1,3}(\alpha t_N) & \cdots & F_{M+5,3}(\alpha t_N)\end{bmatrix}, \tag{35}$$

we can express the second term of J as

$$\sum_{i=1}^{N}(S_3(t_i)-d_i)^2=(H\tau-d)^{\mathrm{T}}(H\tau-d), \tag{36}$$

with $d=[d_1,d_2,\dots,d_N]^{\mathrm{T}}$. Note that the dimension of vector τ and d are $M+6$ and N respectively.

Now, the minimization of the criterion function in Eq. (32) can be reformulated as

$$J_{\min}=\min_{\tau}\left\{w_g\tau^{\mathrm{T}}G\tau+(H\tau-d)^{\mathrm{T}}(H\tau-d)\right\}. \tag{37}$$

However, in the clamped cubic spline, the first three and last three elements in τ are used to determine the boundary condition. Therefore, they are fixed and should not be treated as optimization variable. Here, a transformation matrix U is used to separate the fixed part τ_F and the programmable part τ_P of τ as

$$U\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}=\tau. \tag{38}$$

Then, Eq. (37) can be rewrite as

$$J_{\min}=\min_{\tau_P}\left\{\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}^{\mathrm{T}}w_gU^{\mathrm{T}}GU\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}+\left(HU\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}-d\right)^{\mathrm{T}}\left(HU\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}-d\right)\right\}, \tag{39}$$

which can be further simplified to

$$J_{\min}=\min_{\tau_P}\left\{\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}^{\mathrm{T}}(w_gU^{\mathrm{T}}GU+U^{\mathrm{T}}H^{\mathrm{T}}HU)\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}-2d^{\mathrm{T}}HU\begin{bmatrix}\tau_F\\ \tau_P\end{bmatrix}+d^{\mathrm{T}}d\right\}. \tag{40}$$

By letting $R=w_gU^{\mathrm{T}}GU+U^{\mathrm{T}}H^{\mathrm{T}}HU, S=d^{\mathrm{T}}HU$ then partition R and S according to the dimensions of τ_F and τ_P as

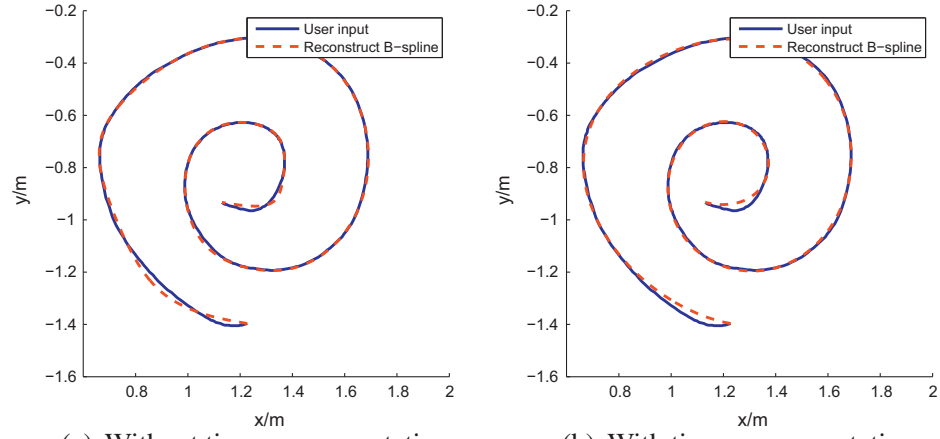
$$R=\begin{bmatrix}R_{FF} & R_{FP}\\ R_{PF} & R_{PP}\end{bmatrix}\text{ and }S=[S_F,S_P], \tag{41}$$

Eq. (40) can be expressed as

$$J_{\min}=\min_{\tau_P}\left\{\tau_F^{\mathrm{T}}R_{FF}\tau_F+\tau_F^{\mathrm{T}}R_{FP}\tau_P+\tau_P^{\mathrm{T}}R_{PF}\tau_F+\tau_P^{\mathrm{T}}R_{PP}\tau_P-2S_F\tau_F-2S_P\tau_P+d^{\mathrm{T}}d\right\}. \tag{42}$$

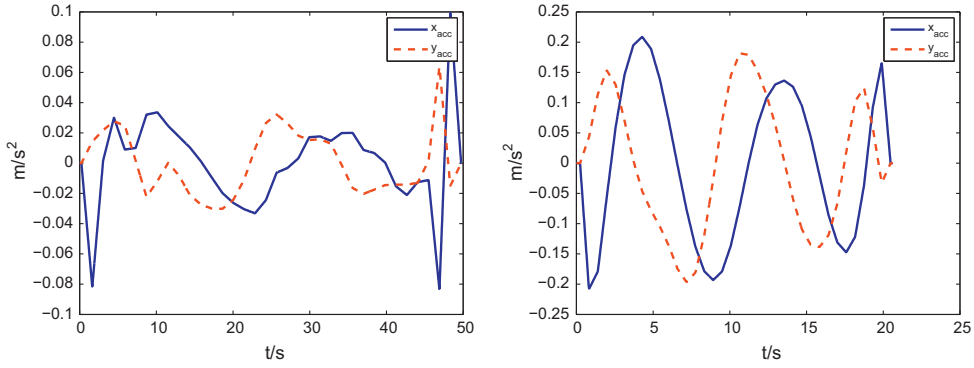
By taking the first derivative with respect τ_P, we will obtain the optimal values for the programmable part in a closed form as

$$\tau_P^*=R_{PP}^{-1}\left(S_P^{\mathrm{T}}-R_{FP}^{\mathrm{T}}\tau_F\right). \tag{43}$$



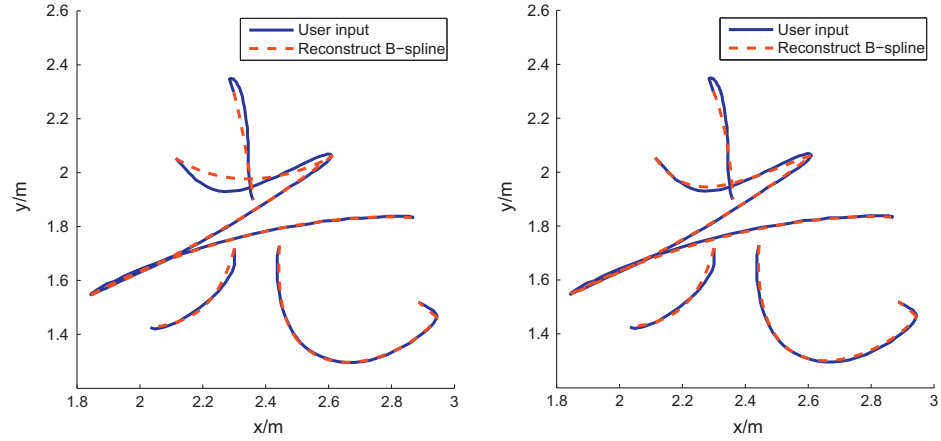
(a) Without time re-segmentation (b) With time re-segmentation

Fig. 6. User input and generated spline of vortex drawing.



(a) Without time re-segmentation (b) With time re-segmentation

Fig. 7. Generated spline's acceleration of vortex drawing.



(a) Without time re-segmentation (b) With time re-segmentation

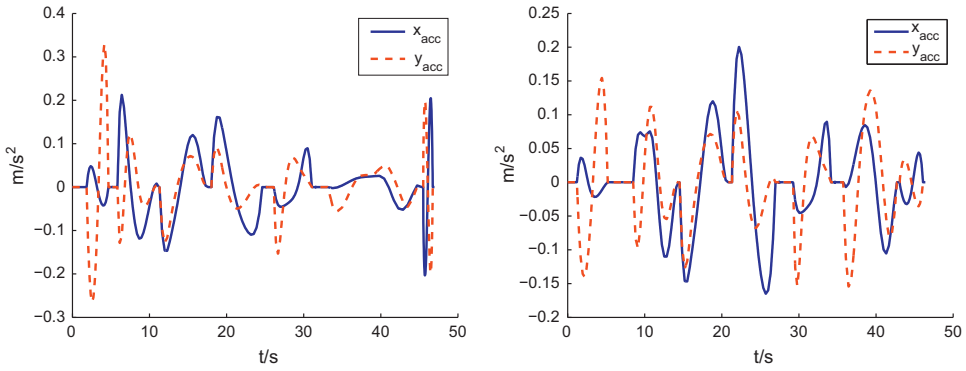
Fig. 8. User input and generated spline of Chinese character *Guang*.

By using the optimal τ_P, one could easily reconstruct the control points vector and the B-spline using the de Boor's algorithm [1].

5.3. Minimum jerk trajectory: quadratic programming

Although there is a closed solution for the minimum jerk trajectory, it is usually not advisable in real application as the dimension of the solution matrices are too large to be computed efficiently. On the other hand, taking a closer look at Eq. (37), it is a typical quadratic optimization problem which could be solved efficiently using off-the-shelf optimization solvers such as *quadprog* from Matlab or *CPLEX* from IBM. Also by considering the problem as a general optimization problem, it allows us to add more constraints on the trajectory, such as the maximum velocity or acceleration.

For the vehicle to write smoothly, a small attitude angle is preferred to minimize the pendulum effect caused by the calligraphy



(a) Without time re-segmentation (b) With time re-segmentation

Fig. 9. Generated spline's acceleration of Chinese character *Guang*.

Fig. 10. Four UAVs writing calligraphy to the public.

brush. As the flyable area is limited, the velocity of the vehicle will naturally be restricted, it is reasonable to just limit the acceleration of the UAV. For generality, the numerical method presented in this manuscript is also suitable to explicitly limit derivatives of any degrees that is smaller than the order of the B-spline basis.

Reformulating Eq. (37), we get

$$J_{\min} = \min_{\tau}\left\{\tau^{\mathrm{T}}(w_g G + H^{\mathrm{T}}H)\tau - 2d^{\mathrm{T}}H\tau + d^{\mathrm{T}}d\right\}. \tag{44}$$

Note that $w_g G + H^{\mathrm{T}}H$ is always positive semi definite which guarantees a unique minimum of the quadratic programming problem. Now, in order to constraint the derivatives, we first express the derivatives of the trajectory. Taking the first derivative of the B-spline trajectory in Eq. (29), we have

$$\frac{dS_3(t)}{dt} = \sum_{i=0}^{M+4} \eta_i F_{i+1,2}(t), \tag{45}$$

where

$$\eta_i = \alpha \frac{3}{u_{i+4} - u_{i+1}}(\tau_{i+1} - \tau_i). \tag{46}$$

By substituting u according to the uniform knot vector $[0,0,0,0,1,2,3,\ldots]$, one could obtain a matrix K as

$$K = \alpha \begin{bmatrix} -3 & 3 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & -\frac{3}{2} & \frac{3}{2} & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & \ldots & 0 \\ \vdots & & & & \ddots & & & & \vdots \\ 0 & 0 & \ldots & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 0 & -\frac{3}{2} & \frac{3}{2} & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & -3 & 3 \end{bmatrix},$$

where

(a) *i love SG (Singapore)* (b) *yi ma dang xian*

Fig. 11. Samples of the UAV calligraphy results.

$$\eta = [\eta_1, \eta_2, \eta_3, \ldots]^{\mathrm{T}} = K\tau. \tag{47}$$

According to [15], to limit maximum and minimum value of the derivative trajectory, a sufficient condition is to let

$$\dot{S}_{3,\min} \leqslant K_i \tau \leqslant \dot{S}_{3,\max}, \quad \forall i = \{0, 1, \ldots, M+4\}, \tag{48}$$

where K_i denotes the i-th row of K. Since here we are focusing on the constraint of acceleration, the second order derivative of the original trajectory is examined. By taking the another derivative on Eq. (45), we arrived at

$$\frac{d^2 S_3(t)}{dt^2} = \sum_{i=0}^{M+3} \gamma_i F_{i+2,1}(t). \tag{49}$$

Here

$$\gamma_i = \alpha \frac{2}{u'_{i+3} - u'_{i+1}} (\eta_{i+1} - \eta_i), \tag{50}$$

where u' is the uniform time vector in the form of $[0, 0, 0, 1, 2, 3, \ldots]$. Substituting them, we have another matrix

$$K' = \alpha \begin{bmatrix} -2 & 2 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & \ldots & 0 \\ \vdots & & & & \ddots & & & & \vdots \\ 0 & 0 & \ldots & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & -2 & 2 \end{bmatrix},$$

where

$$\gamma = [\gamma_1, \gamma_2, \gamma_3, \ldots]^{\mathrm{T}} = K'\eta. \tag{51}$$

Combining both the Eqs. (51) and (47), we have

$$\gamma = K'K\tau \equiv L\tau \tag{52}$$

where matrix L can be computed easily as

$$L = \alpha^2 \begin{bmatrix} 6 & -9 & 3 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & \frac{3}{2} & \frac{-5}{2} & 1 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 & 0 & \ldots & 0 \\ \vdots & & & & \ddots & & & & \vdots \\ 0 & 0 & \ldots & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & \frac{3}{2} & \frac{-5}{2} & 1 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 6 & -9 & 3 \end{bmatrix}.$$

We can then project the acceleration constraints into the constraints of control points vector τ as

$$\ddot{S}_{3,\min} \leqslant L_i \tau \leqslant \ddot{S}_{3,\max}, \quad \forall i = \{0, 1, \ldots, M+3\}, \tag{53}$$

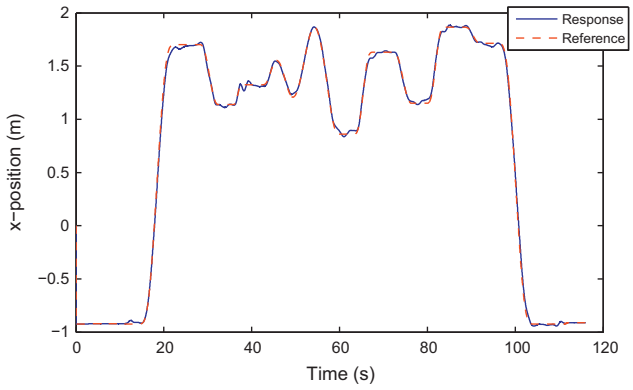
where L_i denotes the i's row of L. To satisfy the boundary condition, one should fix the first three and last three elements of τ. This could be easily formed as an equality constraints of the form

$$A_{\mathrm{eq}} \tau = b_{\mathrm{eq}}, \tag{54}$$

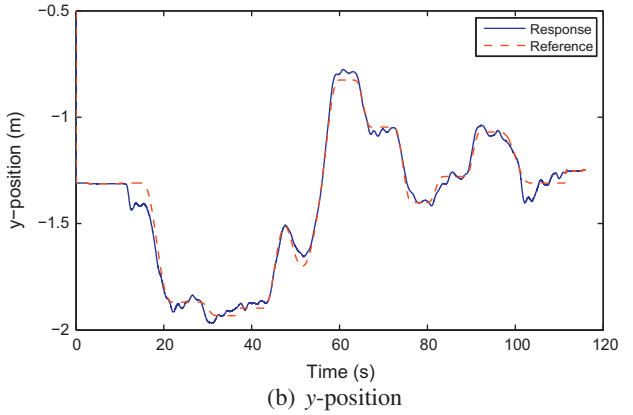
where

$$A_{\mathrm{eq}} = \begin{bmatrix} I_3 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & I_3 \end{bmatrix}.$$

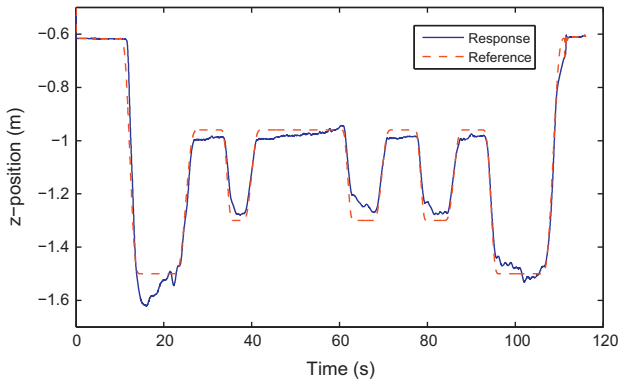
Eqs. (44), (53), and (54) form a typical convex quadratic programming problem which can be solved numerically in an efficient manner [26].



(a) x-position



(b) y-position



(c) z-position

Fig. 12. Position tracking of the UAV.

5.4. Optimal time segmentation

For many cases, especially in our application, the vector T is either not explicitly given or the sampled data does not fit the dynamic of the vehicle. Therefore, it is necessary to re-segment the time vector T while iteratively minimizing the jerk trajectory. Let $T_i = t_{i+1} - t_i$ be the new programmable variables, then a new optimization problem is formulated as

$$\min f(\mathbf{T}), \quad \text{s.t. } T_i > 0. \tag{55}$$

where $f(\mathbf{T})$ is the optimal solution to Eq. (44) for time segment $\mathbf{T} = [T_1, T_2, \ldots, T_{N-1}]^{\mathrm{T}}$. This problem can be solved via a gradient descent method [22] by calculating the gradient vector of f numerically as

$$\nabla f = \frac{f(\mathbf{T} + h\mathbf{g_i}) - f(\mathbf{T})}{h}, \tag{56}$$

where h is a small number at the level of 10^{-6}, $\mathbf{g_i}$ is a perturbation vector with two choices: If the total time to finish the trajectory is allowed to change then $\mathbf{g_i}$ is designed such that its i-th element is 1 and all the others are 0; If the total time is fixed, then $\mathbf{g_i}$ is designed such that its i-th element is 1 and all the others are $-1/N-2$. This is to make sure the $\sum g_i = 0$ so that the total trajectory time remains the same. With the numerically obtained gradient, the gradient descent method is performed using backtrack line search.

By automatic time re-segmentation, the trajectory is further smoothen. It is now more suitable for UAV to perform precision tracking. Benefits of time re-segmentation can be viewed in Figs. 6–9. In the first case study, a simple vortex was hand-drawn to the system. Fig. 6 shows the reconstructed B-spline trajectories both without and with time re-segmentation respectively. Although the trajectory improvement of spline with time re-segmentation seems insignificant in this case, the resulted acceleration reference (Fig. 7) becomes smoother. Besides, the time of the trajectory has improved significantly. In another case study, a Chinese character *Guang* was written. It can be seen that the time re-segmentation improves the interpolation accuracy of the generated trajectory (Fig. 8). According to Fig. 9, the generated acceleration reference to the UAV is relatively more gentle compared to the result without time re-segmentation.

6. Flight test results

In our implementation, the UAV trajectories are generated pre-flight by running a set of MATLAB codes implementing the proposed trajectory planning algorithm. Then, a MATLAB Simulink program is constructed to acquire position and velocity measurements from the Vicon system and execute the outer-loop control law in real time. ZigBee telemetry system is used to transmit and receive data between the UAV on-board avionics and the ground station.

In the testing environment, a graphical interface which allows user handwritten input is presented using a Windows Surface Pro tablet (see Fig. 2). The interface was developed with MATLAB and it is able to transmit user input wirelessly from the Surface to the ground station. The whole UAV calligraphy system has been tested with various arbitrary handwritten inputs. Calligraphy flight performances of public inputs to the system via the Surface were demonstrated live to the audience during the Singapore Airshow 2014 held in Changi Exhibition Center, Singapore. In the system setup during the Airshow, four UAVs are commanded to write four different characters at the same time (see Fig. 10). Reliability and robustness of the whole system has been well tested throughout the entire Airshow duration of 6 days. Several examples of the written words are shown in Fig. 11.

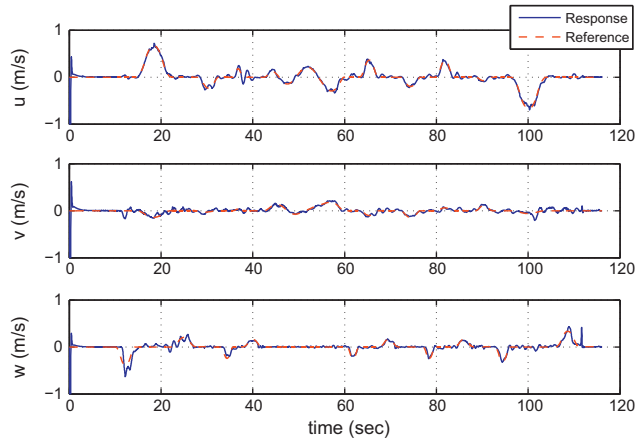
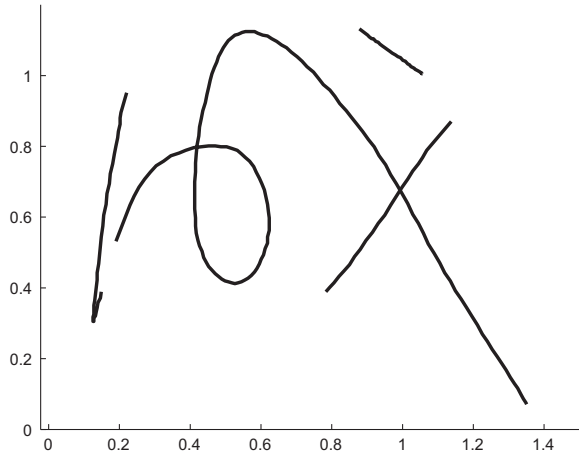
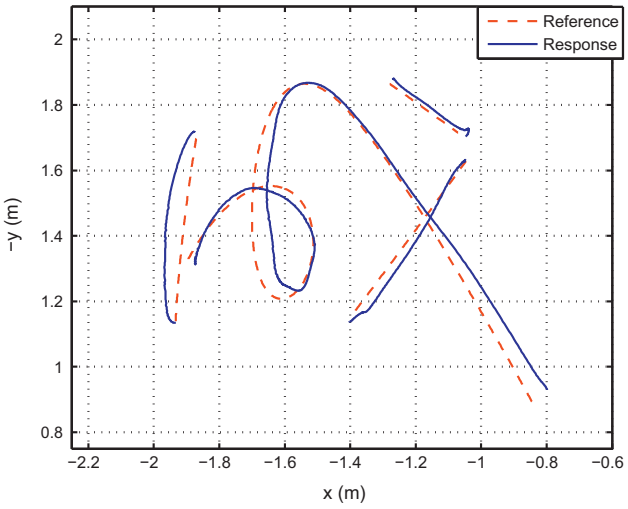


Fig. 13. Velocity tracking of the UAV.



(a) User handwritten input



(b) Generated reference and its response

(c) Writing result

Fig. 14. Sequence of processing.

Besides the visual results, flight tests data were also logged for observation and possible improvement. Fig. 12 shows the *x*-, *y*- and *z*-axis position tracking results during one of the flight test. In general, the UAV is able to perform trajectory tracking almost perfectly, as a result of the RPT controller used in the outer-loop control. However, off-sets were observable on *z*-direction when the UAV descended to certain height due to the ground effect of the UAV from the writing pad.

Fig. 13 shows the velocity references and the corresponding responses during calligraphy writing. The benefit of the RPT controller is clearly shown here as the UAV not only tracks its trajectory well, but also has good velocity control performance. Finally, three diagrams of the Chinese character *Cheng* are shown in Fig. 14. The first diagram shows the user handwritten input via the Surface application we have created. The second diagram shows the generated path (dotted line) of the UAV based on B-spline optimization incorporated with UAV dynamics, while the solid line shows the UAV trajectory response. The final diagram shows the result of the UAV calligraphy on the writing board. The results are satisfying.

7. Conclusions

In this manuscript, we have proposed a UAV calligraphy system which can apply to any miniature UAV aircraft. The system will trace user handwritten input, and then command the UAV to write it out autonomously. Several important aspects and design consideration of such a system are discussed in the previous sections. Hardware modification is first introduced such that the UAV has the required tools to perform UAV calligraphy. Then, a sophisticated trajectory tracing and planning algorithm is discussed in detail, which forms the core of this manuscript. Lastly, controller implementation and flight test results were shown in the later sections. The reliability and robustness of the UAV calligraphy system are well tested in Singapore Airshow 2014, which can be viewed online at http://www.youtube.com/watch?v=zlNlt91N8Kc.

References

[1] de Boor C. A practical guide to splines. New York: Springer-Verlag; 1978.
[2] Borges GA, Aldon MJ. A split-and-merge segmentation algorithm for line extraction in 2d range images. In: Proceedings of the 15th international conference on pattern recognition, Barcelona, Spain; 2000. p. 441–4.
[3] Chen BM. Robust and H_∞ control. Communications and control engineering series. New York: Springer; 2000.
[4] Chen BM, Lee TH, Venkataramanan V. Hard disk drive servo systems. Advances in industrial control series. New York: Springer; 2002.
[5] Chu NSH, Tai CL. Real-time painting with an expressive virtual Chinese brush. IEEE Comput Graph Appl 2004;24(5):76–85.
[6] Cons MS, Shima T, Domshlak C. Integrating task and motion planning for unmanned aerial vehicles. Unmanned Syst 2014;2(1):19–38.
[7] Cui Y, Inanc T. Multiple air robotics indoor testbed. In: Proceedings of the 24th Chinese control and decision conference (CCDC), Taiyuan, China; 2012. p. 3487–92.
[8] Flash T, Hogan N. The coordination of arm movements: an experimentally confirmed mathematical model. J Neurosci 1985;5:1688–703.
[9] Fraundorfer F, Heng L, Honegger D, Lee GH, Meier L, Tanskanen P, et al. Vision-based autonomous mapping and exploration using a quadrotor MAV. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS), Vilamoura; 2012. p. 4557–64.
[10] Fujisawa S, Yoshida T, Satonaka N, Shidama Y, Yamaura H. Improved moving properties of an omnidirectional vehicle using stepping motor. In: Proceedings of the 36th IEEE conference on decision and control, San Diego, CA, vol. 4; 1997. p. 3654–6.
[11] Ghadiok V, Goldin J, Ren W. Autonomous indoor aerial gripping using a quadrotor. In: Proceedings of the 2011 IEEE/RSJ international conference on intelligent robots and systems (IROS), San Francisco, CA; 2011. p. 4645–51.
[12] Heng L, Meier L, Tanskanen P, Fraundorfer F, Pollefeys M. Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using on-board processing. In: Proceedings of the 2011 IEEE international conference on robotics and automation (ICRA), Shanghai, China; 2011. p. 2472–7.
[13] Hoffmann G, Huang H, Waslander S, Tomlin C. Quadrotor helicopter flight dynamics and control: theory and experiment. In: Proceedings of the AIAA guidance, navigation, and control conference, vol. 2; 2007. p. 1–20.
[14] Hoffmann G, Waslander S, Tomlin C. Quadrotor helicopter trajectory tracking control. In: Proceedings of the AIAA guidance, navigation and control conference and exhibit, Honolulu, Hawaii; 2008. p. 1–14.
[15] Kano H, Fujioka H, Martin CF. Optimal smoothing and interpolating splines with constraints. Appl Math Comput 2011;218(5):1831–44.
[16] Kano H, Nakata H, Martin CF. Optimal curve fitting and smoothing using normalized uniform B-splines: a tool for studying complex systems. Appl Math Comput 2005;159(1):96–128.
[17] Keller J, Thakur D, Dobrokhodov V, Jones K, Pivtoraiko M, Gallier J, et al. A computationally efficient approach to trajectory management for coordinated aerial surveillance. Unmanned Syst 2013;1(1):59–74.
[18] Lam JHM, Yam Y. Stroke trajectory generation experiment for a robotic Chinese calligrapher using a geometric brush footprint model. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, St. Louis, MO; 2009. p. 2315–20.
[19] Lo KW, Kwok KW, Wong SM, Yam Y. Brush footprint acquisition and preliminary analysis for Chinese calligraphy using a robot drawing platform. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, Beijing, China; 2006. p. 5183–8.
[20] Meier L, Tanskanen P, Fraundorfer F, Pollefeys M. PIXHAWK: a system for autonomous flight using onboard computer vision. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), Shanghai, China; 2011. p. 2992–7.
[21] Meier L, Tanskanen P, Heng L, Lee GH, Fraundorfer F, Pollefeys M. PIXHAWK: a micro aerial vehicle design for autonomous flight using onboard computer vision. Auton Robot 2012;5(1–2):21–39.
[22] Mellinger D, Kumar V. Minimum snap trajectory generation and control for quadrotors. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), Shanghai, China; 2011. p. 2520–5.
[23] Mellinger D, Michael N, Shomin M, Kumar V. Recent advances in quadrotor capabilities. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), Shanghai, China; 2011. p. 2964–5.
[24] Mellinger D, Michael N, Kumar V. Trajectory generation and control for precise aggressive maneuvers with quadrotors. Int J Robot Res 2012;31(5):664–74.
[25] Mueller MW, Hehn M, D'Andrea R. A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, 2013. p. 3480–6.
[26] Nocedal J, Wright SJ. Numerical optimization. Springer series in operations research and financial engineering. Springer; 2006.
[27] Phang SK, Lai S, Wang F, Lan M, Chen BM. UAV calligraphy. In: Proceedings of the 11th IEEE international conference on control & automation, Taichung, Taiwan; 2014. p. 422–8.
[28] Phang SK, Li K, Yu KH, Chen BM, Lee TH. Systematic design and implementation of a micro unmanned quadrotor system. Unmanned Syst 2014;2(2):121–41.
[29] Richter C, Bry A, Roy N. Polynomial trajectory planning for quadrotor flight. <http://www.michigancmes.org/papers/roy7.pdf>.
[30] Takayama K, Kano H. A new approach to synthesizing free motions of robotic manipulators based on the concept of unit motion. IEEE Trans Syst Man Cybern 1995;25(3):453–63.
[31] Teo CL, Burdet E, Lim HP. A robotic teacher of Chinese handwriting. In: Proceedings of the 10th symposium on haptic interfaces for virtual environment and teleoperator systems, Orlando, FL; 2002. p. 335–41.
[32] Wong HT, Ip HH. Virtual brush: a model-based synthesis of Chinese calligraphy. Comput Graph 2000;24(1):99–113.
[33] Wong TS, Leung H, Ip HS. Model-based analysis of Chinese calligraphy images. Comput Vis Image Understan 2008;109(1):69–85.
[34] Xu S, Lau F, Cheung WK, Pan Y. Automatic generation of artistic Chinese calligraphy. IEEE Intell Syst 2005;20(1):99–113.
[35] Yang H, Lu J, Lee H. A Bezier curve-based approach to shape description for Chinese calligraphy characters. In: Proceedings of the 6th international conference on document analysis and recognition, Seattle, WA; 2001. p. 276–80.
[36] Yao F, Shao G. Modeling of ancient-style Chinese character and its application to CCC robot. In: Proceedings of the IEEE international conference on networking, sensing and control, Ft. Lauderdale, FL; 2006. p. 72–77.
[37] Yao F, Shao G, Yi J. Extracting the trajectory of writing brush in Chinese character calligraphy. Eng Appl Artif Intell 2004;17(6):631–44.
[38] Yu J, Peng Q. Realistic synthesis of cao shu of Chinese calligraphy. Comput Graph 2005;29(1):145–53.
[39] Zhang R, Wang X, Cai K. Quadrotor aircraft control without velocity measurements. In: Proceedings of the 48th IEEE conference on decision and control, Shanghai, China; 2009. p. 5213–8.
[40] Zhou M, Prasad JVR. 3D minimum fuel route planning and path generation for a fuel cell powered UAV. Unmanned Syst 2014;2(1):53–72.