2014 11th IEEE International
Conference on Control & Automation (ICCA)
June 18-20, 2014. Taichung, Taiwan

UAV Calligraphy

Swee King Phang, Shupeng Lai, Fei Wang, Menglu Lan, Ben M. Chen

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 many 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 perform 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.**

I. 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 [1] [2]. Due to their small size and light weight, these UAVs are capable of maneuvering indoors for the various missions or tasks. In the environment where GPS signal is not available, e.g. indoor environment, visual based navigation is usually utilized [3] [4] to estimate the relative distance of the UAV from its original position. 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 [5]. Once the localization issue of the indoor UAV is solved, control and navigation problem such as trajectory generation can be handled [6] [7] [8] [9] [10].

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 [11]. 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 [12] [13]. 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 [14], while most of the development in automated calligraphy writing was realized with robotic arms [15] [16].

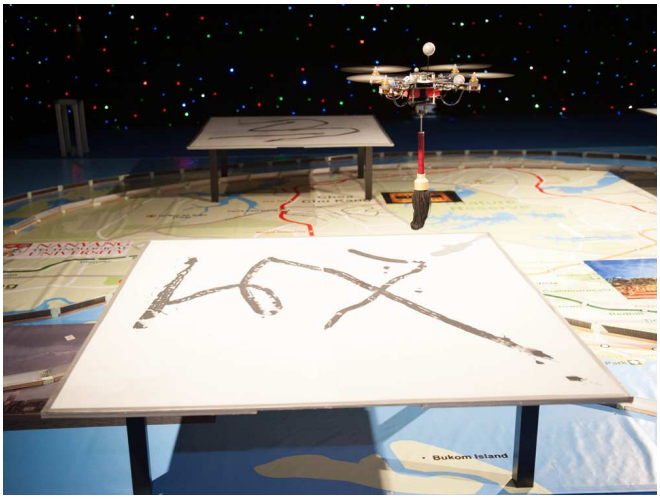


Fig. 1: UAV calligraphy performance in Singapore Airshow

We have identified a few challenges in realizing autonomous UAV calligraphy writing. 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. 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. 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

S. K. Phang and S. Lai are with the NUS Graduate School for Integrative Sciences & Engineering, The National University of Singapore, Singapore. E-mail: {shupenglai,king}@nus.edu.sg

F. Wang is with the Temasek Laboratories, The National University of Singapore, Singapore. E-mail: tslwf@nus.edu.sg

M. Lan and B. M. Chen are with the Department of Electrical & Computer Engineering, The National University of Singapore, Singapore. E-mail: {lanmenglu,bmchen}@nus.edu.sg

978-1-4799-2836-1/14/$31.00 ©2014 IEEE

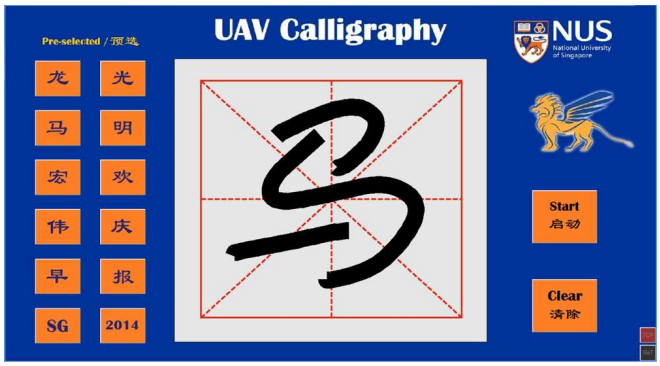


Fig. 2: Graphical interface for user handwriting input

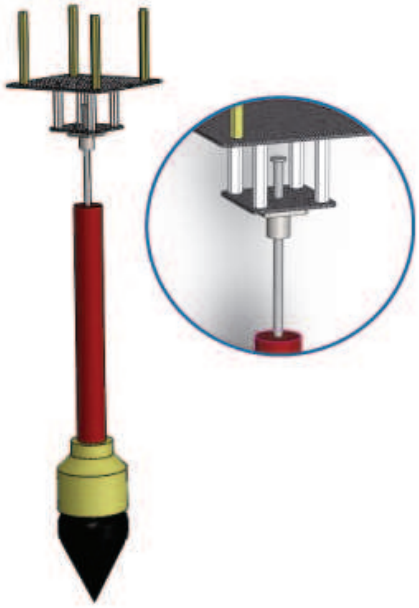


Fig. 3: Custom made calligraphy brush and holder

on the spot.

This manuscript documents the design architecture and the implementation procedure of the UAV calligraphy system. Section II details the design and implementation of the hardware needed to realized UAV calligraphy. Controls and implementation on the UAV will be discussed in Section III. Section IV and Section V shows the handwritten character strokes tracing and trajectory generating. Results of the UAV writing calligraphy will be shown in Section VI, while Section VII gives concluding remarks of our work.

II. 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. The high bandwidth enables a fast tracking outer-loop controller to be designed, as will be shown in Section III 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. 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

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.
2) The calligraphy brush can be rotated freely along z-axis, resulting smooth and natural writing along an arc or circular drawing.

Next, an accurate position sensor is needed for each quadrotor to track the generated trajectory precisely. A Vicon motion sensing system is set up for this purpose. Working principle of the system has been previously documented in [17]. 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 next.

III. UAV Control System

For UAV calligraphy, flight performance of the controlled platform is the main concern to be addressed. 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.

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 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 [18] perfectly fits this requirement.

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

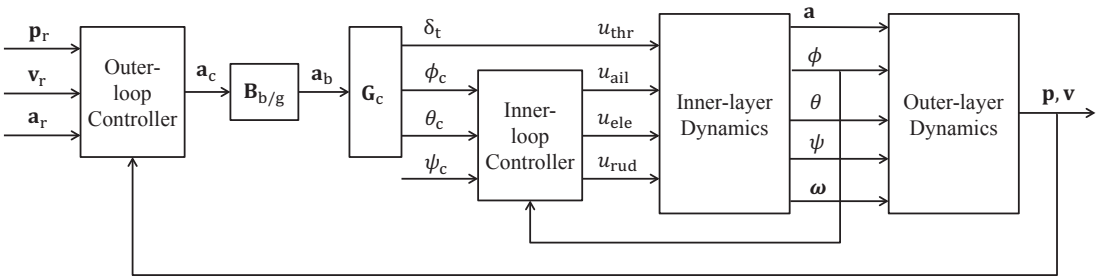


Fig. 4: Dual-loop control structure of the quadrotor

proper choice of flat outputs is

$$\sigma = [x, y, z, \psi]^\mathrm{T}. \tag{1}$$

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. 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

$$\begin{cases} \dot{\boldsymbol{x}}_\mathrm{aug} = \begin{bmatrix} 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}_\mathrm{aug} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_\mathrm{aug} \\ \boldsymbol{y}_\mathrm{aug} = \boldsymbol{x}_\mathrm{aug} \\ h_\mathrm{aug} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}_\mathrm{aug} \end{cases}, \tag{2}$$

where $\boldsymbol{x}_\mathrm{aug} = \begin{bmatrix} \int(\mathbf{p}_\mathrm{e}) & \mathbf{p}_\mathrm{r} & \mathbf{v}_\mathrm{r} & \mathbf{a}_\mathrm{r} & \mathbf{p} & \mathbf{v} \end{bmatrix}^\mathrm{T}$ 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 [19], a linear feedback control law of the following form can be acquired:

$$u_\mathrm{aug} = F_\mathrm{aug} \boldsymbol{x}_\mathrm{aug}, \tag{3}$$

where

$$F_\mathrm{aug} = \begin{bmatrix} \dfrac{k_i \omega_n^2}{\varepsilon^3} & \dfrac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} & \dfrac{2\zeta\omega_n + k_i}{\varepsilon} \\ & 1 & -\dfrac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} & -\dfrac{2\zeta\omega_n + k_i}{\varepsilon} \end{bmatrix}.$$

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 (2) through

$$p_i(s) = (s + k_i)(s^2 + 2\zeta\omega_n s + \omega_n^2). \tag{4}$$

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:

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

IV. 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 the 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 [20]. The original sequenced 2-D points set is acquired by recording user's handwritten input through a tablet. 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.

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.

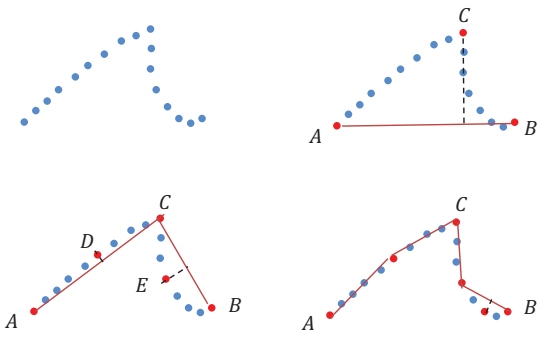


Fig. 5: Split-and-merge sequence on continuous line segments

V. Minimum Jerk Trajectory Planning

In order to achieve a smooth response from the vehicle, a minimum jerk trajectory is naturally considered. We have divided the minimum jerk trajectory planning problem into three different parts. Starting with the normalization of the uniform B-spline, followed an approach to solve the minimum jerk trajectory problem with numerical quadratic programming. 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.

A. Normalized Uniform B-Spline

The base of the B-splines has been defined in [21] as a recursive function

$$N_{i,0}(u) = \begin{cases} 1, & \text{if } u_i \le u < u_{i+1}, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

$$N_{i,p}(u) = \frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u), \tag{6}$$

where $N_{i,p}(u)$ is the basis function of the generally defined B-splines and $[u_0,u_1,u_2,...]$ forms the knot vector of B-splines. Normalization of the uniform B-spline will thus produce the knot vector $[0,1,2,3,...]$.

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 $[0,0,0,0,1,2,3,...,m-1,m,m,m,m]$, 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 Equation (5), we could easily deduce the different basis accordingly.

The general cubic clamped spline function can be written in the form of

$$S_3(t) = v(t) = \sum_{i=0}^{M+5} \tau_i F_{i,3}(t), \quad \tau_i \in \mathbb{R}, \tag{7}$$

where τ_i denote the trajectory points to be optimized, $F_{i,3}$ are the customized spline bases derived from the normalized spline.

B. Minimum Jerk Trajectory: Quadratic Programming

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] and [23]. If a set of sampled data points is given as

$$\begin{aligned} D &= \{d_i \in \mathbb{R} : i = 1,...,N\}, \\ T &= \{t_i \in \mathbb{R} : i = 1,...,N\}, \end{aligned} \tag{8}$$

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

$$J = w_g \int_{-\infty}^{\infty} v^2(t)dt + \sum_{i=1}^{N}(S_3(t_i) - d_i)^2, \tag{9}$$

where w_g is a weighting factor, and

$$v(t) = \sum_{i=0}^{M+5} \frac{d^3}{dt^3}\tau_i F_{i,3}(t), \quad \tau_i \in \mathbb{R}, \tag{10}$$

is the third derivative of the spline function.

It is further shown in [24] that $\int_{-\infty}^{\infty} v^2(t)dt$ can be expressed as the form of $\tau^T G\tau$, where $\tau = [\tau_0,\tau_1,...,\tau_{M+5}]^{\mathrm{T}}$, and G can be calculated explicitly. More specifically, the elements in G are

$$g_{i,j} = \alpha^5 \int_{-\infty}^{\infty} F_{i,3}^{(3)}(t)F_{j,3}^{(3)}(t)dt, \tag{11}$$

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

$$H = \begin{bmatrix} F_{0,3}(\alpha t_1) & F_{1,3}(\alpha t_1) & \dots & F_{M+5,3}(\alpha t_1) \\ F_{0,3}(\alpha t_2) & F_{1,3}(\alpha t_2) & \dots & F_{M+5,3}(\alpha t_2) \\ \vdots & \vdots & \ddots & \vdots \\ F_{0,3}(\alpha t_N) & F_{1,3}(\alpha t_N) & \dots & F_{M+5,3}(\alpha t_N) \end{bmatrix}, \tag{12}$$

we can express the second term of J as

$$\sum_{i=1}^{N}(S_3(t_i) - d_i)^2 = (H\tau - d)^{\mathrm{T}}(H\tau - d), \tag{13}$$

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 Equation (9) can be reformulated as

$$J_{\min} = \min_{\tau}\left\{w_g\tau^{\mathrm{T}}G\tau + (H\tau - d)^{\mathrm{T}}(H\tau - d)\right\}. \tag{14}$$

This 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. Reformulating the equation, we get

$$J_{\min} = \min_{\tau} \left\{ \tau^{\mathrm{T}} (w_g G + H^{\mathrm{T}} H) \tau - 2 d^{\mathrm{T}} H \tau + d^{\mathrm{T}} d \right\}. \quad (15)$$

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, we have

$$\frac{dS_3(t)}{dt} = \sum_{i=0}^{M+4} \eta_i F_{i+1,2}(t), \quad (16)$$

where

$$\eta_i = \frac{3}{t_{i+4} - t_{i+1}} (\tau_{i+1} - \tau_i). \quad (17)$$

According to [25], to limit maximum and minimum value of the derivative trajectory, a sufficient condition is to let

$$\dot{S}_{3,\min} \leq \eta_i \leq \dot{S}_{3,\max}, \quad \forall i = \{0, 1, ..., M+4\}. \quad (18)$$

Since here we are focusing on the constraint of acceleration, the second order derivative of the original trajectory is examined. We can then project the acceleration constraints into the constraints of control points vector τ as

$$\ddot{S}_{3,\min} \leq L_i \tau \leq \ddot{S}_{3,\max}, \quad \forall i = \{0, 1, ..., M+5\}, \quad (19)$$

where L_i denotes the i-th row of L. Here L can be calculated easily from Equation (16). 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

$$A_{\mathrm{eq}} \tau = b_{\mathrm{eq}}, \quad (20)$$

where

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

Equations (15), (19), and (20) form a typical convex quadratic programming problem which can be solved numerically in an efficient manner [26].

C. 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

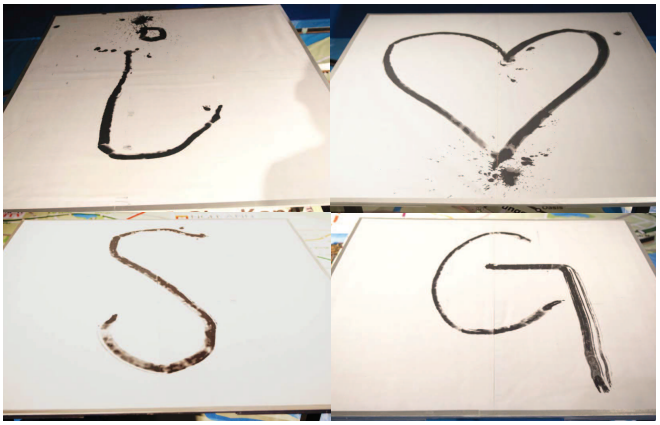
$$\min f(\mathbf{T}), \quad \text{s.t. } T_i > 0, \quad (21)$$

where $f(\mathbf{T})$ is the optimal solution to Equation (15) 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

$$\nabla f = \frac{f(\mathbf{T} + h\mathbf{g_i}) - f(\mathbf{T})}{h}, \quad (22)$$

where h is an arbitrary small number, $\mathbf{g_i}$ 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 $\frac{-1}{N-2}$. 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 resegmentation, the trajectory is further smoothen. It is now more suitable for UAV to perform precision tracking. Time resegmentation has also been proven in our results that it can improve the interpolation accuracy of the generated trajectory.



(a) *i love SG (Singapore)*

(b) *yi ma dang xian*

Fig. 6: Samples of the UAV calligraphy results

VI. 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

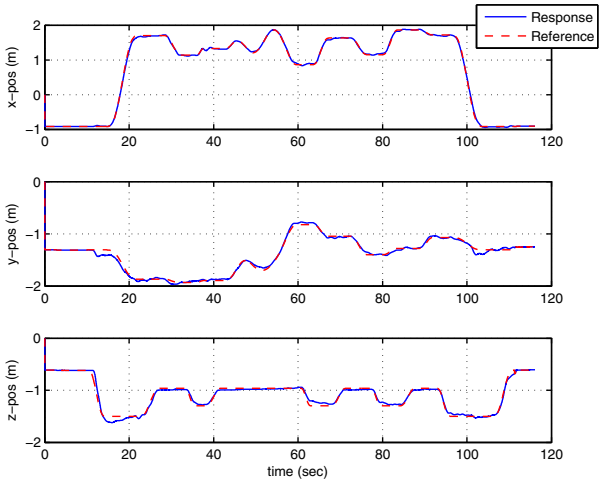


Fig. 7: Position tracking of the UAV

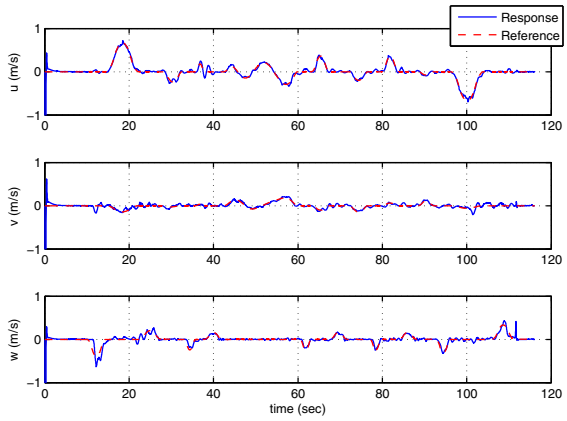
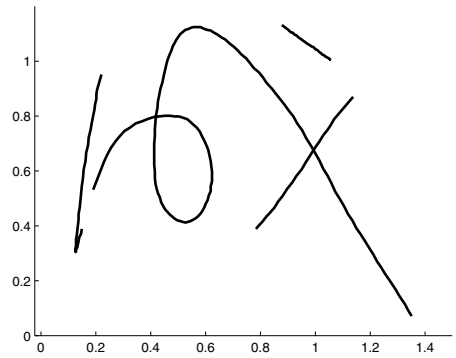
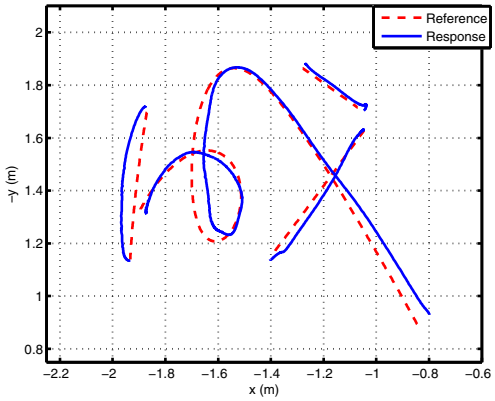


Fig. 8: Velocity tracking of the UAV



(a) User handwritten input



(b) Generated reference and its response

(c) Writing result

Fig. 9: Sequence of processing

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 in Windows Surface Pro (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 Singapore Airshow 2014 held in Changi Exhibition Centre, Singapore. In the system setup during the Airshow, four UAVs are commanded to write four different characters at the same time. 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. 6.

Besides the visual results, flight tests data were also logged for observation and possible improvement. Fig. 7 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. 8 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. 9. The first diagram shows the user handwritten input via the Surface software interface 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.

VII. 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.

References

[1] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision" *Autonomous Robots*, vol. 5(1-2), pp. 21-39, 2012.

[2] L. Meier, P. Tanskanen, F. Fraundorfer and M. Pollefeys, "PIXHAWK: A system for autonomous flight using onboard computer vision", *Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, pp. 2992-2997, 2011.

[3] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor MAV", *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, pp. 4557-4564, 2012.

[4] V. Ghadiok, J. Goldin and W. Ren, "Autonomous indoor aerial gripping using a quadrotor", *Proceedings of 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, CA, pp. 4645-4651, 2011.

[5] D. Mellinger, N. Michael, M. Shomin and V. Kumar, "Recent advances in quadrotor capabilities", *Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, pp. 2964-2965, 2011.

[6] R. Zhang, X. Wang and K. Cai, "Quadrotor aircraft control without velocity measurements", *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, pp. 5213-5218, 2009.

[7] G. Hoffmann, H. Huang, S. Waslander and C. Tomlin, "Quadrotor helicopter flight dynamics and control: theory and experiment", *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, vol. 2, 2007.

[8] J. Keller, D. Thakur, V. Dobrokhodov, K. Jones, M. Pivtoraiko, J. Gallier, I. Kaminer and V. Kumar, "A computationally efficient approach to trajectory management for coordinated aerial surveillance", *Unmanned Systems*, vol. 1(1), pp. 59-74, 2013.

[9] M. Zhou and J. V. R. Prasad, "3D minimum fuel route planning and path generation for a fuel cell powered UAV", *Unmanned Systems*, vol. 2(1), pp. 53-72, 2014.

[10] M. S. Cons, T. Shima and C. Domshlak, "Integrating task and motion planning for unmanned aerial vehicles", *Unmanned Systems*, vol. 2(1), pp. 19-38, 2014.

[11] S. Xu, F. Lau, W. K. Cheung and Y. Pan, "Automatic generation of artistic Chinese calligraphy", *IEEE Intelligent Systems*, vol. 20(1), pp. 99-113, 2005.

[12] H. T. Wong and H. H. Ip, "Virtual brush: a model-based synthesis of Chinese calligraphy", *Computers and Graphics*, vol. 24(1), pp. 99-113, 2000.

[13] H. Yang, J. Lu, H. Lee, "A Bezier curve-based approach to shape description for Chinese calligraphy characters", *Proceedings of 6th International Conference on Document Analysis and Recognition*, Seattle, WA, pp. 276-280, 2001.

[14] S. Fujisawa, T. Yoshida, N. Satonaka, Y. Shidama and H. Yamaura, "Improved moving properties of an omnidirectional vehicle using stepping motor", *Proceedings of the 36th IEEE Conference on Decision and Control*, San Diego, CA, vol. 4, pp. 3654-3656, 1997.

[15] C. L. Teo, E. Burdet and H. P. Lim, "A robotic teacher of Chinese handwriting", *Proceedings of 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Orlando, FL, pp. 335-341, 2002.

[16] F. Yao, G. Shao and J. Yi, "Extracting the trajectory of writing brush in Chinese character calligraphy", *Engineering Applications of Artificial Intelligence*, vol. 17(6), pp. 631-644, 2004.

[17] S. K. Phang, K. Li, K. H. Yu, B. M. Chen and T. H. Lee, "Systematic design and implementation of a micro unmanned quadrotor system", *Unmanned Systems*, vol. 2(2), 2014.

[18] B. M. Chen, T. H. Lee and V. Venkataramanan, *Hard Disk Drive Servo Systems*. Advances in Industrial Control Series, New York: Springer, 2002.

[19] B. M. Chen, *Robust and H_∞ Control*. Communications and Control Engineering Series, Springer, 2000.

[20] G. A. Borges and M. J. Aldon, "A split-and-merge segmentation algorithm for line extraction in 2d range images", *Proceedings of the 15th International Conference on Pattern Recognition*, Barcelona, Spain, pp. 441-444, 2000.

[21] C. de Boor, *A Practical Guide to Splines*. Springer-Verlag, New York, 1978.

[22] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors", *Proceedings of 2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, pp. 2520-2525, 2011.

[23] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for quadrotor flight", http://www.michigancmes.org/papers/roy7.pdf.

[24] H. Kano, H. Nakata and C. F. Martin, "Optimal curve fitting and smoothing using normalized uniform B-splines: a tool for studying complex systems", *Applied Mathematics and Computation*, vol. 159(1), pp. 96-128, 2005.

[25] H. Kano, H. Fujioka and C. F. Martin, "Optimal smoothing and interpolating splines with constraints", *Applied Mathematics and Computation*, vol 218(5), pp. 1831-1844, 2011.

[26] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer Series in Operations Research and Finicial Engineering, Second Edition, Springer, 2006.