The 8th ACM/EG Expressive Symposium
C. Kaplan, A. Forbes, and S. DiVerdi (Editors)

Single Stroke Aerial Robot Light Painting

Kejia Ren[1] and Paul G. Kry[2]

[1]Tongji University, Shanghai, China
[2]McGill University, Montreal, Canada

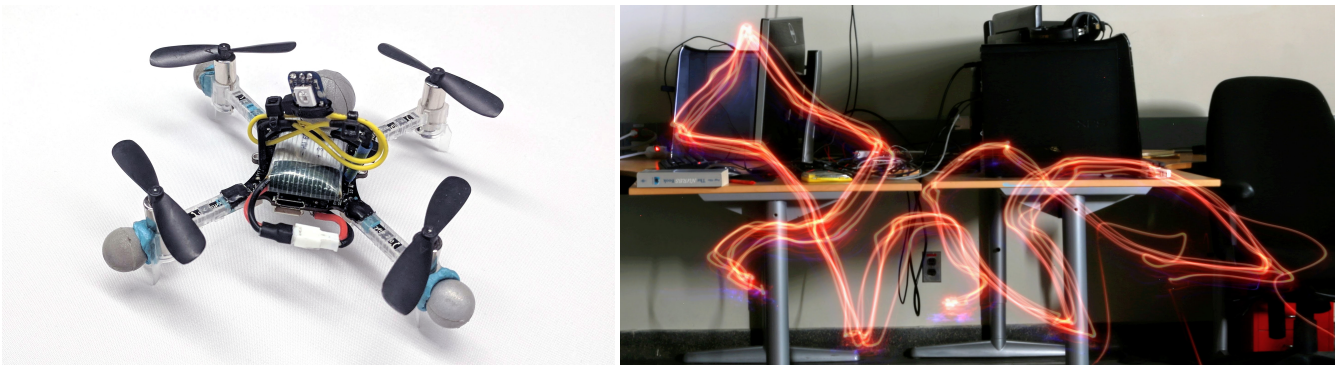

Figure 1: *A NeoPixel mounted on a Crazyflie 2.0 (left) illuminates during flight to create light paintings in a long exposure photograph. A composite photograph (right) shows variability across multiple flights in creating a light painting of a fox.*

Abstract

This paper investigates trajectory generation alternatives for creating single-stroke light paintings with a small quadrotor robot. We propose to reduce the cost of a minimum snap piecewise polynomial quadrotor trajectory passing through a set of waypoints by displacing those waypoints towards or away from the camera while preserving their projected position. It is in regions of high curvature, where waypoints are close together, that we make modifications to reduce snap, and we evaluate two different strategies: one that uses a full range of depths to increase the distance between close waypoints, and another that tries to keep the final set of waypoints as close to the original plane as possible. Using a variety of one-stroke animal illustrations as targets, we evaluate and compare the cost of different optimized trajectories, and discuss the qualitative and quantitative quality of flights captured in long exposure photographs.

CCS Concepts

• ***Computing methodologies*** → ***Motion path planning;*** *Image and video acquisition;*

1. Introduction

Light shows with aerial robots have become a reality, with thousands of robots operating like flying pixels, each with a controllable light, and coordinating to form shapes and messages in the sky to amaze crowds at large sporting events. But rather than exploring the frontiers of robot swarm cooperation, in this work we investigate the minimalist problem of producing light paintings with long exposure photography of a single aerial robot. Robots have become an important part of artistic works [Gol11], and have been a vehicle for exploring ideas in the creation of artifacts, for example, in drawing [TL12], stippling [GK17], and painting [LMPD15]. At the core of many of these endeavors are important technical challenges and computational problems that require a scientific approach to designing and evaluating these robot systems.

In our work, we directly put into practice the seminal work of Mellinger and Kumar [MK11], which proposes minimum snap piecewise polynomials for quadrotor trajectory planning through a given a set of waypoints. Given a set of waypoints specifying the flight path of a single stroke light painting, restricting the path of the robot to a plane is unnecessary because the primary goal is to have the path of the light on the robot project to the desired illustration in the photograph. Because high curvature trajectories involve larger cost (i.e., typically higher snap), we displace waypoints off the plane while preserving their projection in order to generate a

slightly longer optimized trajectory of similar appearance with better overall cost.

We use Crazyflie 2.0 robots in this work. They are a nice platform because they have open software and hardware and are easy to extend. Furthermore, they are small and light, making them very safe in comparison to many other quadrotor robots. Figure 1 shows a photo of the quadrotor robot we use in our light painting work and a preview of the results.

2. Related Work

The related work can be divided into two main categories: research on trajectory planning and control of quadrotor robots, and work related to light painting.

The work by Mellinger and Kumar [MK11] proposes polynomial trajectories and shows that with a differentially flat representation of the quadrotor, there can be a very convenient formulation for generating high quality quadrotor trajectories. The positional coordinates and yaw are decoupled in both the cost function and the constraints, allowing four quadratic programming problems to be solved separately with a specified timing, and then subsequently refined with gradient descent by allowing adjustment to the timing. Richter et al. [RBR16] presents a modification to solve longer piecewise polynomial trajectory problems efficiently. We use an implementation of this trajectory optimization algorithm which was adapted from the work of Burri et al. [BO*15], which also includes velocity and acceleration constraints. These trajectory optimization techniques are a very convenient approach to the problem, especially due to the availability of the Crazyswarm software [PHSA17] that we use in this work.

There are a number application specific problems that have been investigated in the context of aerial robot trajectory generation and optimization. For instance, in optimizing trajectories for scanning [RSD*17], and for generating feasible trajectories for quadrotor cameras through re-timing [RH16]. Other recent examples include optimizing aerial camera trajectories for aesthetics [GSH18], and the general control of drones in cinematography [NMD*17, GLC*18]. In our work, the camera is not on the robot, but instead records the performance of an optimized flight trajectory from a fixed position.

In computer graphics research, Salamon et al. [SLE17] present a computational approach to light painting in a post-process by drawing on the screen. Real world light painting with a robot arm is investigated by Huang et al. [HTWL18]. They create swept volume light paintings by taking long exposure photographs of a display panel swept through a curved path. Also of note is the use of an aerial robot in photography to produce optimal rim illumination [SBD14], though in this case the goal is to control the light rather than to photograph the light directly. Robotic light painting projects also exist within the art and design community, such as work by Crossman and McPhail [CM14] that renders Kinect sensor scans using a bright RGB LED panel mounted on an IRB 6440 industrial robot. Keating and Oxman [KO13], present an exploration of fabrication processes in the context of architecture art and design, including real-time light renders generated by robotic control of light sources. Light painting is also used in education as an example application for teaching the inverse kinematics of robotic manipulators [DWK14]. In contrast to these artistic and pedagogical works, our work focuses on the technical challenges of creating good trajectories for a quadrotor in the creation of single stroke light paintings.

Our previous work [GKAK16] provides details of how we control a robot to produce stippled prints. This involves computing a stipple pattern for an image, greedy path planning, a model for how ink is used up as stipples are placed, and a technique for dynamically adjusting future stipples based on past errors. This previous work also discussed how the same system and similar ideas could support multi-stroke light paintings, and included one example of a cube drawn with 12 strokes. In this paper, in contrast, we focus on single stroke trajectories and explore the possibility of varying the depth.

3. Method

For light painting, the quadrotor should always face the camera because the NeoPixel appears brightest when viewed directly. Thus, we to set the yaw to be constant and only optimize for the 3D position trajectory. Likewise, we describe the desired trajectory with a set of waypoints defined on a plane normal to the camera's viewing direction and at a fixed distance. This is because the images we want to produce are all 2D single-stroke illustrations. We chose a variety of single-stroke animal figures for our light paint trajectories, defining camel, penguin, and flamingo trajectories based on famous continuous line drawings by Picasso, and a variety of other animal trajectories based on a set of one-line animal logos created by the French creative duo known as DFT.

We follow the previous work [MK11, RBR16, BO*15] in using minimum snap piecewise polynomial splines as the representation for our quadrotor trajectories, and exploit the separability of the trajectory optimization problem with the center of mass chosen as flat output. For a trajectory with m segments, and a duration T_i for each segment, the cost function of dimension d has the form

$$J_d = \sum_{i=1}^{m} \int_0^{T_i} \left(\frac{d^4 p_{i,d}(t)}{dt^4} \right)^2 , \tag{1}$$

where $p_{i,d}(t)$ is the polynomial trajectory of dimension d for segment i. The waypoints provide constraints to this optimization problem, and when the cost functions of different dimensions are summed together, the problem can be seen as minimizing the integral of the snap norm squared subject to waypoint constraints.

In our experiments we optimize for 1 m/s and 1 m/s^2 maximum velocity and acceleration, respectively, and note that trajectory waypoint sequences which contain spans lower than 20 cm tend to have much larger cost due to high curvature. This suggests that we should include depth as a parameter in the optimization of the minimum snap polynomials and their timing.

With the main optimization solving decoupled quadratic programs, it would seem at first that there is little benefit to exploiting the depth, but the image taken by the camera is not an orthographic projection (i.e., not a telephoto lens), and ultimately the different

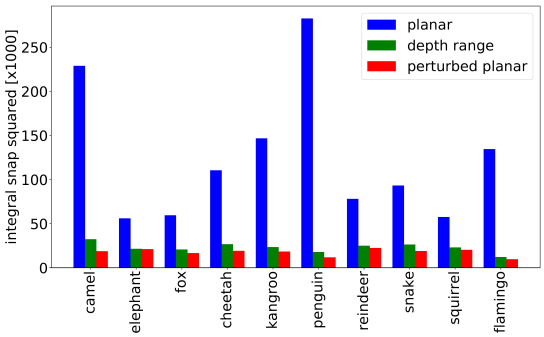

Figure 2: *Integral of snap squared of optimized trajectories is greatly reduced with both depth range and perturbed planar waypoint modification techniques.*

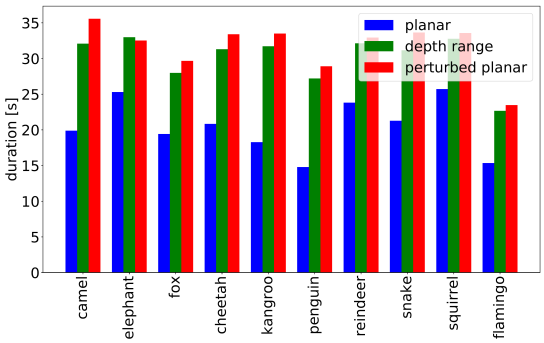

Figure 3: *Duration of flights for different animal figures for different trajectory conditions. The duration of trajectories that exploit depth are always slightly longer than the planar case.*

dimensions are coupled when the solution is refined by adjusting the segment durations.

Instead of formulating a new optimization problem, we set a 20 cm distance threshold between waypoints and define two straightforward procedures for altering the waypoints to reduce the optimized trajectory cost. We loop over the waypoints until we find a waypoint that is within 20 cm of the previous. We then compute two positions on the line between the center of projection of the camera and the problematic waypoint which are 25 cm from the previous waypoint (an additional 5 cm is added to the span for extra effect). At this moment, we either choose to move the robot closer or farther to the camera, and update the problematic waypoint as well as all subsequent waypoints to the new depth value. This scales all the remaining points to be closer together when choosing to move toward the camera, which would seem to be counterproductive except that we cannot simply let the trajectory get pushed arbitrarily far from the camera because it must stay within the motion capture volume. Therefore, we explore two different strategies: keeping the robot in a fixed *depth range*, and keeping the robot as close to the original plane as possible, which we call the *perturbed depth* method.

For the depth range method we fix a minimum and maximum distance. We choose to push the trajectory away from the camera until it would exceed the maximum, at which point we switch to approaching the camera, bouncing between the minimum and maximum. In our experiments we start the robot at the origin of the motion capture coordinate frame (on the floor in the middle of the room). With the camera at a distance of about 2.7 m in the x direction, we set the range to be 0.3 to -0.7 meters in this coordinate frame.

For the perturbed depth method, we remain close to the plane in which the waypoints were originally defined by stepping either toward or away from the camera, depending on which would bring us closer to the original plane.

Figure 2 shows the cost reduction we obtain for the depth range and perturbed depth approaches in comparison to the original planar trajectory. Trajectories which have many short spans, such as the penguin, show the largest reduction. The cost of the perturbed depth optimized trajectories tends to be lower than the depth range

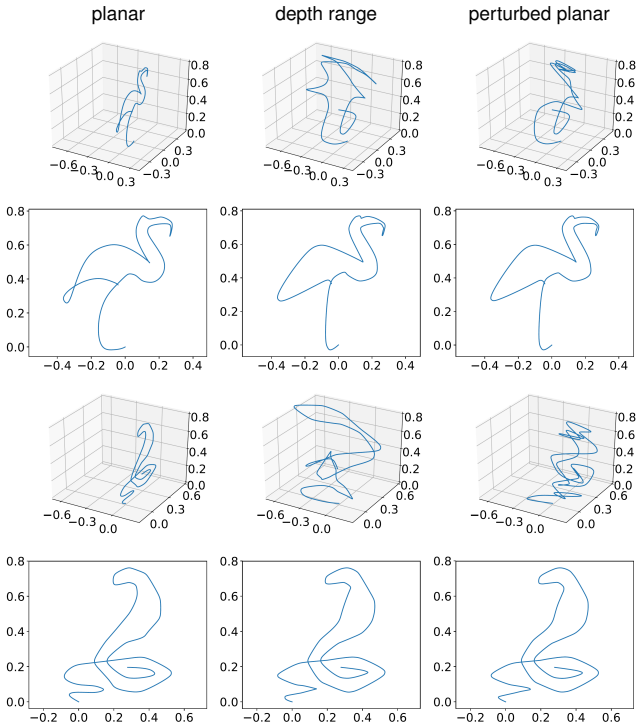

Figure 4: *The optimized trajectories for waypoints displaced in depth project to very similar images as seen by the camera.*

trajectories by a small amount. Because our procedure is making the paths longer, they will likewise take more time to complete. Figure 3 shows the durations for our animal examples are about 1.6 to 2 times as long.

The optimized trajectories for the modified waypoints will end up using different amounts of depth. For the flamingo, the depth range case results in a polynomial trajectory that uses 94 cm, while the perturbed planar cases uses 55 cm. For the snake, the depth range case uses 92 cm while the perturbed planar case uses 51 cm. Figure 4 shows the resulting trajectories.

Figure 5: *Single stroke light painted animals: elephant, camel, cheetah, kangaroo, penguin, squirrel, snake, flamingo, and reindeer. Each long exposure photo was taken over approximately 10 seconds.*

3.1. Calibration

We must know the position of the camera in the motion capture coordinate frame. While we can place markers on the camera, we typically want to place our camera outside of the motion capture volume. We have used both a measuring tape, as well as an Aruco marker placed at the origin of the motion capture volume to measure the camera position in motion capture coordinates to be 2.77 m in the x coordinate, 0.1 m in the y coordinate, and 0.95 m up in the z direction. This position is important so that the waypoints can be correctly modified with respect to the camera's center of projection, but having very precise numbers is not critical. The hovering capability of the robot has limited precision (see hover tests in previous work [GK17]), and can stray from a target by several centimeters. This position control error easily dominate other sources of error in our calibration.

3.2. System

We use a NaturalPoint motion capture system with 12 cameras. The rigid body position and orientation data of the robot is streamed over the local network to Crazyswarm software [PHSA17] running on a separate computer. We run a python script to set the robot's polynomial control trajectories which we load from a file. We likewise simultaneously send synchronized commands to control the time at which the NeoPixel turns on and off, as well as the colour. We can change the colour of the pixel throughout the single stroke, but chose to use fixed colours for the animal figures in our examples. The Crazyflie firmware required a small modification for us to use the standard controls of the LED-ring expansion board (i.e., we attach only a single NeoPixel to the serial communication lines of the expansion board, which does not respond to queries from the robot in the same way as other expansion boards).

4. Results

We use a Canon 70D camera for all of our results, for which a collection of examples can be seen in Figure 5. We used a 28 mm prime lens, closed the aperture to f/22, and set the ISO to 100 to allow for long exposure photographs under low ambient light without over exposure. The shutter is held open (B mode) for the duration of the trajectory, which varies depending on the figure. Ultimately, the different animal figures take only a few tens of seconds to draw.

We have evaluated the variability of the results produced by the robot under the different approaches compared to the planar flight

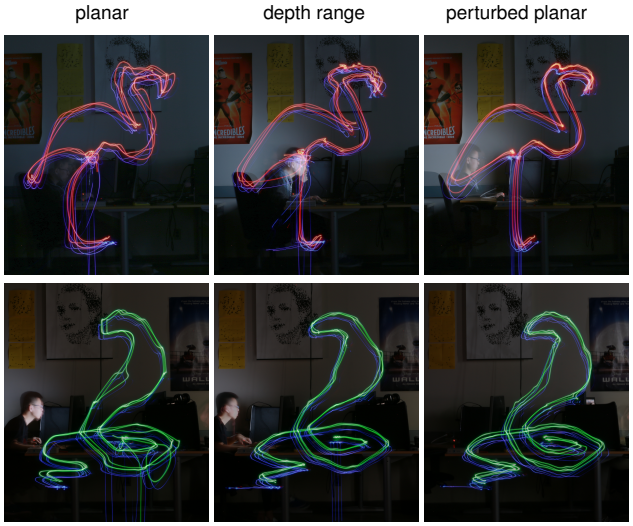

Figure 6: *Multiple photos superimposed shows the variability between flights across the three types of optimized trajectories.*

Table 1: *Frechet distance between planned trajectory and photo.*

	planar	depth range	perturbed depth
flamingo	0.053	0.036	0.041
snake	0.070	0.071	0.057

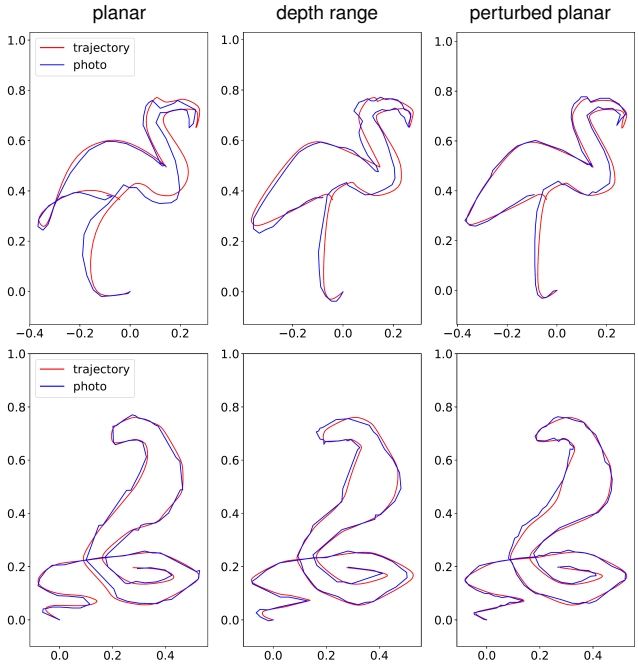

Figure 7: *Comparison of the planned trajectory with a trajectory traced from a photo for the flamingo and snake under the three conditions.*

trajectory condition. Figure 6 shows the variability of the flights for the flamingo and snake under the three conditions. We generally observe that flights with trajectories that remain in the plane have more variability, which we believe is due to a variety of factors, such as the slow flight segments near high curvature regions in the figures. When the depth direction is exploited, the flight trajectory can have much lower curvature.

As a quantitative evaluation, we compute a discrete approximation of the Frechet distance [EM94] to compare photographed trajectories to target trajectories. Due to timing variability in the exact moment that the light is turned off at the end of the trajectory, we trim the end of traced flight curves in order to not focus only at the distance between endpoints. While other integral measures of curve distance may arguably be better measures of distance for performance, we note that the L1-style Frechet distance suggests that the trajectories which exploit depth are sometimes better than the planar trajectory. Figure 7 shows traced trajectories from photos along with the planned trajectory, while Table 1 provides the Frechet distance numbers. Note that the figure and table only consider the best of 5 flights in each of the 3 conditions. The units are in meters because we compute an optimal scale of the traced photograph to have it align with the planned trajectory as closely as possible. Finally, note that the trajectories differ slightly across the 3 cases because of the variation in the optimized polynomial curve segments (the trajectories are interpolating different sets of points that project to the desired points in the image).

Finally we explore a number of creative long exposure photographs that include human participation. Figure 8 shows the results, while time lapse videos in the supplementary video show the creation process. In each scenario, the human participant takes a pose within the field of view of the camera during the long exposure photograph. We briefly illuminate a studio light to expose the participant, and in the case of the *bursting skeleton* example, we illuminate twice in two different poses.

4.1. Limitations

We do not take lens distortion into account in our image formation model. However, the current main source of error comes from our small robot's limited ability to accurately fly a trajectory or hover at a given position. An interesting avenue for future work would be to plan trajectories for wide angle lenses and 360 degree VR cameras.

In some cases, we note that the quality of images was influenced by control latency and what we believe to be radio communication interference in the lab. When a the flight control was poor or failed, we would need to simply try the flight again (sometimes immediately, sometimes later). We addresses radio communication in our previous work [GKAK16] by creating a robot firmware patch, but we did not merge this change into the crazyswarm firmware that we used in this project.

5. Conclusion

Our inspiration for this work comes from the unusual and beautiful effects created by amateur photographers, as well as the single-stroke light paintings performed by Picasso [Cos12]. We believe we have succeeded in the creation of a set of interesting photographs,

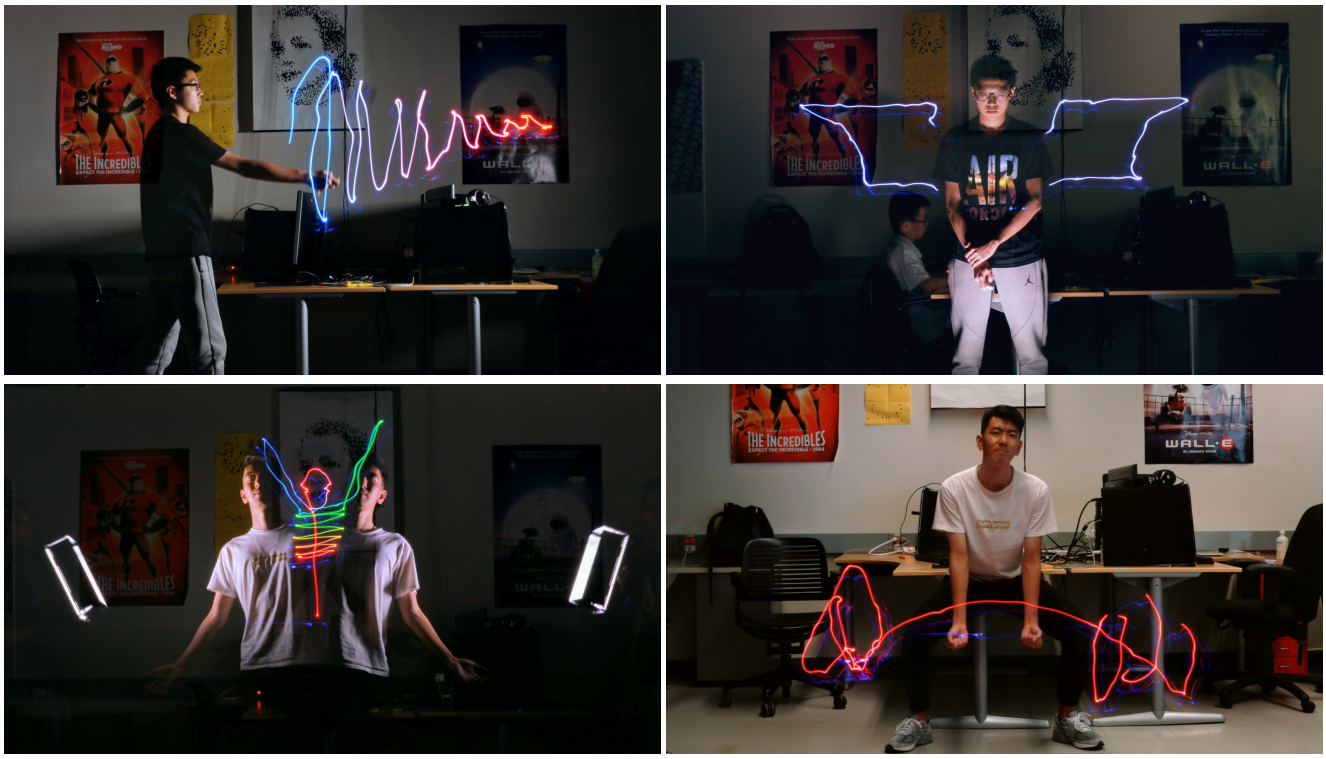

Figure 8: *Human participation in long exposure photographs to create whimsical scenarios: Hadouken magic, batman wings, bursting skeleton, and lifting a barbell.*

in part due to the quality of one-line animal illustrations that we used as initial waypoints. We displace waypoints in the camera projection direction, and show that our two strategies are successful at reducing the cost of minimum snap piecewise polynomial quadrotor trajectories. The new trajectories are slightly longer, have lower curvature, and differ slightly from each other. A discrete approximation of the Frechet distance was used to evaluate the robot's ability to follow the trajectories generated with different strategies. While the Frechet distance does not provide a strong indication of better performance in the final photograph, it would likely be better to evaluate error across the full trajectory rather than measuring only the worst point. Qualitative comparisons of the photographs, nevertheless, suggest a moderate level of success. Ultimately, we speculate that this kind of quadrotor light painting could have interesting applications at festivals, theme parks, or corporate events.

5.1. Future Work

An interesting avenue of future work is to formulate an optimization problem that not only minimizes snap by exploiting depth, but also has objectives that penalize trajectories whose projection deviate from the desired form when viewed by a fixed camera. Adding this shape penalty would reduce the variability we currently see in different optimized trajectories. It would also be interesting to explore modifications to low level feedback control so as to work in task-space with compliance in the camera projection direction [SVKS14]. Finally, in our preliminary experiments with light painting, we considered using multiple strokes, which introduces a variety of interesting computational problems, such as the need for Hamiltonian paths or Eulerian cycles for efficiently drawing a set of strokes.

Acknowledgment

We thank Colin Gallacher for assistance with robot repairs and insightful discussions, and Nick Iversen for a loan of additional equipment to evaluate alternative tracking systems.

References

[BO*15] BURRI M., OLEYNIKOVA H., , ACHTELIK M. W., SIEGWART R.: Real-time visual-inertial mapping, re-localization and planning onboard MAVs in unknown environments. In *Intelligent Robots and Systems (IROS 2015), 2015 IEEE/RSJ International Conference on* (Sept 2015). 2

[CM14] CROSSMAN J., MCPHAIL K.: Industrail light painting. CMU HCI Institute student project, 2014. 2

[Cos12] COSGROVE B.: Behind the picture: Picasso 'draws' with light. *Time* (Jan 2012). 5

[DWK14] DAMES P., WONG D., KUCHENBECKER K. J.: Teaching forward and inverse kinematics of robotic manipulators via MATLAB. ICRA 2014 workshop: MATLAB/Simulink for Robotics Education and Research, 2014. 2

[EM94] EITER T., MANNILA H.: *Computing discrete Fréchet distance*. Tech. Rep. Tech. Rep. CD-TR 94/64, Christian Doppler Labor für Expertensyteme, TU Wien, 1994. 5

[GK17] GALEA B., KRY P. G.: Tethered flight control of a small quadrotor robot for stippling. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Sept 2017), pp. 1713–1718. 1, 4

[GKAK16] GALEA B., KIA E., AIRD N., KRY P. G.: Stippling with aerial robots. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering* (2016), Expressive '16, Eurographics Association, pp. 125–134. 2, 5

[GLC*18] GALVANE Q., LINO C., CHRISTIE M., FLEUREAU J., SERVANT F., TARIOLLE F.-L., GUILLOTEL P.: Directing cinematographic drones. *ACM Trans. Graph. 37*, 3 (July 2018), 34:1–34:18. 2

[Gol11] GOLDBERG K.: *Artist Portfolio*. Catharine Clark Gallery, San Francisco, California, 2011. 1

[GSH18] GEBHARDT C., STEVŠIĆ S., HILLIGES O.: Optimizing for aesthetically pleasing qadrotor camera motion. *ACM Trans. Graph. 37*, 4 (July 2018), 90:1–90:11. 2

[HTWL18] HUANG Y., TSANG S.-C., WONG H.-T. T., LAM M.-L.: Computational light painting and kinetic photography. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2018), Expressive '18, ACM, pp. 14:1–14:9. 2

[KO13] KEATING S., OXMAN N.: Robotic immaterial fabrication. In *Rob | Arch 2012* (Vienna, 2013), Brell-Çokcan S., Braumann J., (Eds.), Springer Vienna, pp. 256–266. 2

[LMPD15] LINDEMEIER T., METZNER J., POLLAK L., DEUSSEN O.: Hardware-based non-photorealistic rendering using a painting robot. *Computer Graphics Forum (Eurographics) 34*, 2 (2015). 1

[MK11] MELLINGER D., KUMAR V.: Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation* (May 2011), pp. 2520–2525. 1, 2

[NMD*17] NÄGELI T., MEIER L., DOMAHIDI A., ALONSO-MORA J., HILLIGES O.: Real-time planning for automated multi-view drone cinematography. *ACM Trans. Graph. 36*, 4 (July 2017), 132:1–132:10. 2

[PHSA17] PREISS* J. A., HÖNIG* W., SUKHATME G. S., AYANIAN N.: Crazyswarm: A large nano-quadcopter swarm. In *IEEE International Conference on Robotics and Automation (ICRA)* (2017), IEEE, pp. 3299–3304. Software available at https://github.com/USC-ACTLab/crazyswarm. 2, 4

[RBR16] RICHTER C., BRY A., ROY N.: Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In *Robotics Research*. Springer, 2016, pp. 649–666. 2

[RH16] ROBERTS M., HANRAHAN P.: Generating dynamically feasible trajectories for quadrotor cameras. *ACM Trans. Graph. 35*, 4 (July 2016), 61:1–61:11. 2

[RSD*17] ROBERTS M., SHAH S., DEY D., TRUONG A., SINHA S. N., KAPOOR A., HANRAHAN P., JOSHI N.: Submodular trajectory optimization for aerial 3d scanning. In *ICCV* (2017), pp. 5334–5343. 2

[SBD14] SRIKANTH M., BALA K., DURAND F.: Computational rim illumination with aerial robots. In *Computational Aesthetics (Expressive 2014)* (2014). 2

[SLE17] SALAMON N. Z., LANCELLE M., EISEMANN E.: Computational light painting using a virtual exposure. *Comput. Graph. Forum 36*, 2 (May 2017), 1–8. 2

[SVKS14] SADEGHIAN H., VILLANI L., KESHMIRI M., SICILIANO B.: Task-space control of robot manipulators with null-space compliance. *IEEE Transactions on Robotics 30*, 2 (April 2014), 493–506. 6

[TL12] TRESSET P. A., LEYMARIE F. F.: Sketches by Paul the robot. In *Proceedings of the Eighth Annual Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging* (2012), CAe '12, pp. 17–24. 1