WYFIWYG: Investigating Effective User Support in Aerial Videography

Christoph Gebhardt, Otmar Hilliges
Departement of Computer Science, AIT Lab
ETH Zürich

Figure 1: This paper investigates how to effectively support non-expert users in the creation of aerial video shots, comparing (A) the state-of-the-art and (B) WYFIWYG, a tool inspired by expert workflows. (C) The resulting plans can be flown on real robots.

ABSTRACT
Tools for quadrotor trajectory design have enabled single videographers to create complex aerial video shots that previously required dedicated hardware and several operators. We build on this prior work by studying film-maker's working practices which informed a system design that brings expert workflows closer to end-users. For this purpose, we propose WYFIWYG, a new quadrotor camera tool which (i) allows to design a video solely via specifying its frames, (ii) encourages the exploration of the scene prior to filming and (iii) allows to continuously frame a camera target according to compositional intentions. Furthermore, we propose extensions to an existing algorithm, generating more intuitive angular camera motions and producing spatially and temporally smooth trajectories. Finally, we conduct a user study where we evaluate how end-users work with current videography tools. We conclude by summarizing the findings of work as implications for the design of UIs and algorithms of quadrotor camera tools.

Author Keywords
robotics; quadrotor camera tools; computational design

ACM Classification Keywords
I.2.9 Robotics: Autonomous vehicles; Operator interfaces; H.5.2 User Interfaces;

INTRODUCTION

Cheap and robust quadrotor hardware has recently brought the creation of aerial videography into the reach of end-users. However, creating high-quality video remains a difficult task since users need to control the drone and the camera simultaneously, while considering cinematographic constraints such as target framing and smooth camera motion [6]. To automate this difficult control problem, several computational tools for aerial videography have been proposed [13, 16, 27], casting aerial videography as an optimization problem which takes desired camera positions in space and time as input and generates smooth quadrotor trajectories that respect the physical limits of the robot. Informed by formative feedback from photographers and filmmakers, this early work focuses on abstracting robot and camera control aspects to be able to plan challenging shots. In this paper we study if and how experts could leverage such tools in their workflows. Based on this formative feedback we design a new system that brings such workflows closer to end-users.

Aiming to translate expert working practices for end-users, we propose WYFIWYG, a new quadrotor camera tool. Based on the findings of formative interviews with film-makers and quadrotor operators, we implemented a UI that (i) enables users to design a video solely via specifying its frames (hiding quadrotor-related aspects like force diagrams or a 2D-trajectory), (ii) a camera control mechanism that encourages the exploration of a scene and (iii) a keyframe sampling method allowing to *continuously* frame a camera target according to compositional intentions.

In addition, we extend an existing algorithm [13] to generate more intuitive angular camera motions and to improve the overall smoothness of quadrotor camera trajectories. Finally, we conduct a user study in which we evaluate WYFIWYG and a state-of-the-art tool [16]. A key-finding is that current tools complicate the design of globally smooth video shots by requiring users to specify keyframes at equidistant points in time and space. We conclude by summarizing implications for UI and optimization scheme design that are important to support users in creating aerial videos.

In summary, we contribute: 1) An analysis and discussion of formative expert interviews. 2) A new UI design for aerial videography. 3) Extensions to an existing quadrotor camera trajectory optimizer [13]. 4) A discussion of implications for future UI and algorithmic research based on the study results.

RELATED WORK

Robotic Behavior Control

Automating the design of robotic systems based on high-level functional specifications is a long-standing goal in graphics and HCI. Focusing on robot behavior only, tangible UIs [33], and sketch based interfaces to program robotic systems [21, 28] have been proposed. Recently, several works introduce gestures as a mean for human-drone interaction [3, 10].

Camera Control in Virtual Environments

Camera placement [18], path planning [31, 17] and automated cinematography [20] have been studied extensively in the context of virtual environments, for a survey see [4]. Many of these papers identify the need for suitable UI metaphors so that intelligent cinematography tools can support film makers in the creative process. Most notably the requirement to let users define and control the recorded video as directly as possible, instead of controlling the camera parameters (e.g., [9, 19, 20]). In this context it is important to consider that virtual environments are not limited by real-world physics and robot constraints, hence can produce camera trajectories that could not be flown by a quadrotor.

Trajectory Generation

Quadrotor motion plan generation is a well studied problem and various approaches have been proposed, including generation of collision-free plans applied to aerial vehicles [29, 26], global forward planning approaches to generate minimum snap trajectories [22], or real-time methods for the generation of point-to-point trajectories [23].

Computational Support of Aerial Videography

With the increasing popularity of aerial videography a number of tools to support this task exist. Commercial applications are often limited to placing waypoints on a 2D map [1, 7, 30].

Several algorithms for the planning of quadcopter trajectories, taking both aesthetic objectives and the physical limits of the robot into consideration, have been proposed. These tools allow for the planning of camera shots in 3D [13, 16, 27]. Airways [13] allows users to specify keyframe-based trajectories and select a camera target for each keyframe. After generation users can inspect the trajectory and see a video preview. With

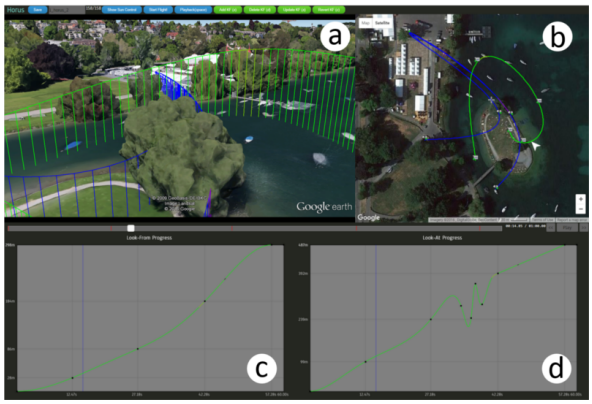

Figure 2: Horus [16] visualizes a user-specified trajectory in 3D (a) and 2D (b). Two plots visualize progress over time for the look-from / quadrotor (c) and look-at / camera targets (d) trajectories and allow users to change timing of a video.

Horus [16] users can specify a camera trajectory using a 3D preview or a 2D map (see Figure 2). The tool offers progress curves for quadrotor and camera target positions, allowing users to change the timing of a video. Horus can detect but not correct violations of the limits of the robot model. In contrast, [27] proposes a method which takes physically infeasible camera paths as input and generates quadrotor trajectories that match the intended camera motion as closely as possible.

[16] conducted an evaluation of their tool with cinematographers. We study aspects pertaining to end-users and contribute new insights on quadrotor videography form this perspective.

Recently, several works have been published which cover the generation of quadrotor camera trajectories in real-time to record dynamic scenes. Real-time performance is attained by planning only locally [25, 24] or by reducing the problem to a lower-dimensional subspace [12, 15]. In contrast to these papers, our work focuses on the generation of quadrotor motion for city or landscape shots.

FORMATIVE INTERVIEWS

To inform our design, we conducted a series of expert interviews. Here we report on aspects which experts defined as being crucial for creating pleasing aerial video and which are not, to a satisfying extend, supported in existing tools.

We interviewed six professional users including three aerial videographers, producing for instance footage for real estate agencies and other commercial purposes, two professional camera men working on TV, movie and documentary sets and one quadrotor operator specialized on high-quality commercials and Hollywood film productions. We visited our participants in their offices or workshops during their working hours to understand their workflows, workplaces and the equipment and tools used for the planning and the execution of aerial video shots. The interviews were not restricted in duration and typically lasted between 1 and 2 hours. The interviews were semi-structured around questions on planning procedures, workflow and tool use. In addition, we introduced the participants to two existing quadrotor camera tools [13, 16]

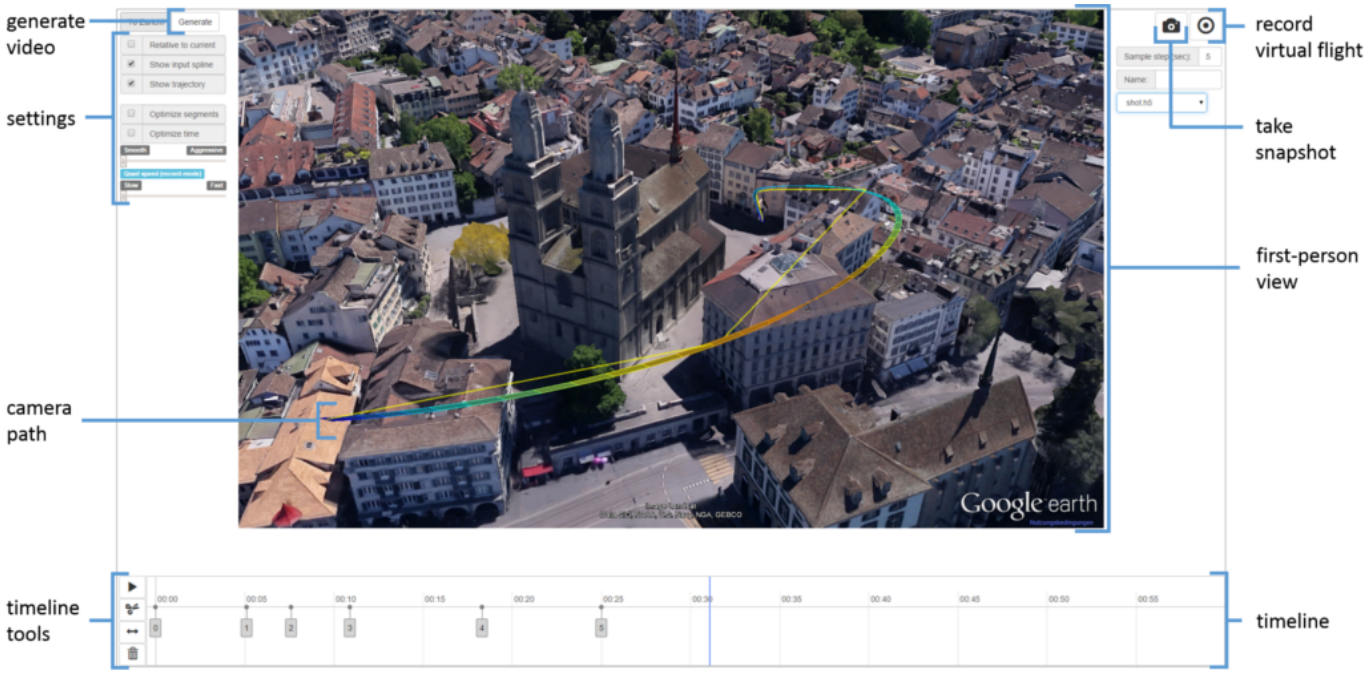

Figure 3: In WYFIWYG users can define keyframes in first-person view. They can add keyframes to a video by taking a snapshot of the current view or recording a virtual flight. A timeline enables the adjustment of a shot's timing.

via the original videos. We then asked the experts to explore with us if and how these tool could support existing workflows and which additional features would be desirable. While our experts also stated aspects already mentioned in literature [16], we now highlight previously unreported results.

Target Framing

The ability to control and fine-tune the framing of a filmed subject *continuously* and with high-precision is an essential aesthetic tool. The interviewees highlighted the importance of being able to precisely position an object in the image plane subject to a compositional intention (e.g., a simultaneously moving foreground and background). For this reason, aerial video shots are usually taken by two operators, one piloting the quadrotor and one controlling the camera, allowing to constantly fine-tune the subject framing. Several professional operators also stated that following a specific quadrotor trajectory is not a primary concern, or in the words of one of our participants *"what counts is the result [video], not the trajectory of the quadrotor"*. For instance, even when circling a filmed object, one participant explained that this is always performed based on the live camera stream and flying a perfect circle may even be counterproductive.

Smooth Camera Motion

The key to aesthetically pleasing aerial video is described by one of our participants as *"[...] the camera is always in motion and movements are smooth"*. Another expert stated that smoothness is considered *the* criteria for shots with a moving camera (see also [2, 14]), whereas the dynamics of camera motion should stay adjustable. We stress this point since current algorithms keep the temporal position of keyframes fixed, hence can only generate smooth motion locally and produce varying camera velocities in-between different sections of a trajectory (see section Method, Smooth Camera Motion).

Exploration

In practice, aerial shots are often defined in-situ in an exploratory fashion. In professional settings so-called 'layout-drones' are used to initially record a scene from various perspectives and only after reviewing the results, high-end equipment is used for the final shot. Most interviewees stressed that this phase is of fundamental importance to find good shots.

USER INTERFACE DESIGN

Based on above findings, we propose a new tool, aiming to translate expert working practices for end-users via an easy-to-use UI design. In the following, we will explain UI, camera control, and virtual flight mode of WYFIWYG and highlight how they are derived from the expert interviews.

Video UI

To reduce complexity we design the UI in a way that it transforms the general task of specifying a robot movement plan into a task more akin to creating a video. Therefore, we take the design decision to hide all quadrotor-related aspects like a 2D-trajectory or input-force diagrams. Users see the virtual world through a first-person-view and can freely position this view within a 3D virtual environment (see Figure 3). Once satisfied with a viewpoint, it can be added to the timeline as a video frame. After each keyframe insertion, an optimization algorithm generates a trajectory and the resulting video can be previewed immediately. Similar to common video editing tools, we also provide a timeline and functionality to edit the shot timings (e.g., moving keyframes in time). Due to this

example-centric approach our tool does not provide an editable trajectory visualization (camera path is still rendered in 3D) and users need to specify keyframes in the image plane to design a video. Taking up the "circling around an object" example from the expert interviews, we designed our UI to lead users in positioning keyframes based on what they see in the preview, focusing on framing and not worrying about the geometric shape of the trajectory.

Integrated Camera Control

Unpacking the need for precise target framing, experts highlighted that in professional settings, two operators work together to adjust a camera's position as well as its pitch and yaw angle simultaneously. To enable a similar way of working in our single-user tool, we provide a control mechanism which integrates translational and rotational degrees of freedom. Research has shown that integrating translational and rotational degrees of freedom gives users more fine-grained control over 3D movements [32] and should lead to better compositional abilities when framing a camera target. For our tool, we implemented a 3D-camera control which can be used with a variety of input devices that allow for simultaneous control of 5-DoF (quadrotor cameras do not allow for roll), such as game pads or multi-touch controls (cf. video). In addition, the experts also highlighted the importance of environment exploration for finding interesting perspectives and planning an aesthetically pleasing camera path. By providing an integrated camera control in combination with a first person view, users can virtually fly through the 3D scene like in a flight simulator. With this gamified interaction, we intent to encourage users to explore the environment when designing a shot. In contrast, Airways only shows a 3D preview after trajectory generation. Horus offers a preview at planning time which would generally allow for exploration. Nevertheless, we believe that mouse interaction (which separates translational and rotational movement) makes exploration cumbersome compared to a gamepad.

Virtual Flight

A final finding relates to the need to continuously re-fine subject framing over an entire shot. To allow for continuous target framing, we implemented an extension to the basic keyframe-based setting which we dub *virtual flight* mode. In this mode, the user directly records the entire shot by flying in first person view through the virtual environment (without specifying discrete keyframes). Behind the scenes, we automatically sample the camera's position and orientation (at an adjustable time interval). Our algorithm adopts the positions of the virtual camera motion, optimizing and smoothing only its dynamics. Based on the suggestion of a participant, the resulting motion plan can also be played-back and edited in situ to fine-tune target framing. This mode lends the paper its title: WYFIWYG or "what you fly is what you get".

METHOD

In addition to the UI design we also contribute extensions to existing trajectory generation methods allowing for more fine-grained target framing and easy creation of smooth camera motion. Our algorithm is based on the method presented in [13]. A recap can be found in this paper's appendix.

Target Framing

The context analysis highlighted the importance of fine tuning target framing. In the real world setting the camera is oriented and positioned to align a target in image plane, in order to achieve a desired compositional effect. In contrast, Airways and Horus orient the camera based on user-defined target positions and generate a look-at trajectory in-between them. In Airways, these look-at positions are always centered in image plane, taking away all compositional abilities. Horus provides the possibility to adjust target framing by moving a camera's look-at position with respect to a camera target. Nevertheless, orienting the camera based on a look-at trajectory can yield undesirable effects. First, optimizing the camera orientation based on a shortest path interpolation in-between look-at positions can cause unexpected camera tilting. We illustrated

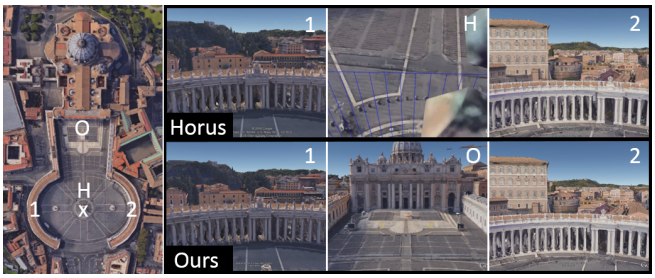

Figure 4: On the left the position of the virtual camera (x), the two specified look-at positions (1, 2) and the look-at position of the generated intermediate frames (H,O) are shown. The first row on the right shows the generated video of Horus with a camera tilt due to the shortest path interpolation between the two look-at points (H). The second row shows the result of our optimization method for the same input, framing St Peter's Basilica in the middle of the shot (O, cf. video).

this problem in Figure 4, where the shortest path interpolation in-between keyframes causes the camera to miss large parts of St Peter's Basilica[1]. A problem which occurs more often are undesirable camera dynamics. Orienting the camera based on a timed trajectory causes its motion to be faster when the reference point on the trajectory is close to the position of the camera and slower when the reference point is more distant. Although, in both cases the covered distance in camera angle is the same, thus smooth camera motion could be generated[2]. To overcome these problems, we model pitch and yaw angle of the camera (roll is not desired in a videography setting) and optimize them based on the orientation of the virtual camera of user-specified keyframes. Modeling the gimbal with:

$$\dot{\psi}_g = u_{g,\psi} \tag{1}$$
$$\dot{\phi}_g = u_{g,\phi}$$
$$[\psi_{g,min}, \phi_{g,min}]^T \leq [\psi_g, \phi_g]^T \leq [\psi_{g,max}, \phi_{g,max}]^T \tag{2}$$
$$\mathbf{u}_{g,min} \leq [u_{g,\psi}, u_{g,\phi}]^T \leq \mathbf{u}_{g,max},$$

where the inputs $u_{g,\psi}$, $u_{g,\phi}$ represent the angular velocities of the yaw ψ_g and pitch ϕ_g of the gimbal and both the inputs and the absolute angles are bounded according to the dynamics

[1]The example is chosen specifically to visualize the problem.
[2]See video from 1:50 min to 2:50 min.

and range-of-motion of the physical gimbal. Using this gimbal model, we now introduce an additional cost-term

$$E^o = \sum_{j=1}^{M} ||(\psi_{g,\eta(j)} + \psi_{q,\eta(j)}) - \psi_j||^2 + \sum_{j=1}^{M} ||\phi_{g,\eta(j)} - \phi_j||^2. \tag{3}$$

Where ψ_j and ϕ_j are the desired yaw and pitch orientation of the camera at each keyframe, $\psi_{g,\eta(j)}$, $\psi_{q,\eta(j)}$ and $\phi_{g,\eta(j)}$ are the gimbal and quadrotor yaw angle as well as the gimbal pitch angle at a keyframe's corresponding time point on the trajectory. By modeling the yaw angle of the quadrotor and the gimbal separately and adding it up in Eq. (3), the generated trajectories can exploit the full dynamic range of the quadrotor and the gimbal around the world frame z-axis. Furthermore, by separating the reference tracking of pitch and yaw in Eq. (3), we can prevent undesired camera tilt in-between keyframes for most cases (see example in the bottom row of Figure 4). We now rewrite the gimbal model Eq. (1) as a discretized first-order dynamical system, formulate this system as equality constraints, state its bounds (Eq. (2)) as inequality constraints and incorporate both into the original optimization problem (Eq. (11), appendix). We add E^o to the objective function of [13] and include a penalizing term on higher derivatives of the yaw angles and the gimbal pitch (cf. Eq. (10), appendix).

In the original method the non-linearities introduced by the camera target tracking required the usage of a computationally expensive iterative quadratic programming scheme [13]. In contrast our method remains quadratic and can be solved directly. This reduces optimization run times for camera target tracking problems from tens of seconds to seconds (a camera trajectory with 20 seconds runtime is generated in 2 seconds compared to 14 seconds with [13]).

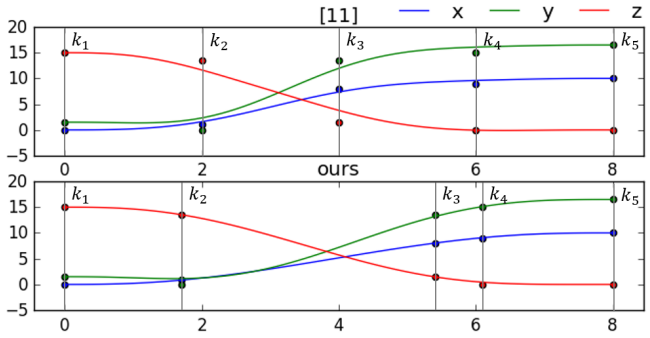

Figure 5: Comparison of trajectory generation methods. Compared to [13], our method adjusts timings to better fit positional distances of keyframes ($k_1, ..., k_5$).

Smooth Camera Motion

Smooth camera motion over an entire sequence is a quality criteria for aesthetically pleasing aerial videos (see expert interviews). With current tools' optimization schemes this is not easy to achieve since the timings of user-specified keyframes are kept fix and are not optimized when generating a trajectory. Therefore, the resulting camera motion is only smooth locally and still can vary in-between keyframes which results in visually unpleasant video[3]. To generate smooth motion over an entire shot with the existing tools, users need to ensure that the ratio of distance in time to distance in space is similar in-between all keyframes. [16] tackles this problem by providing look-at (camera look-at position) and look-from (quadrotor position) progress curves, allowing to edit the relative progress on a trajectory over time. An even slope over the entire curve indicates a smoothly moving camera. Nevertheless, the effect of manipulating these progress curves on camera motion can sometimes be difficult to understand (see Figure 2, d). To help even novice users to produce globally smooth temporal behavior, we extend our method to not only optimize the positions of keyframes in space but also in time. This can be stated as

$$\underset{\mathbf{t}}{\text{minimize}} \quad f(\mathbf{t}) + Nw \tag{4}$$

$$\text{subject to} \quad t_{i-1} < t_i < t_{i+1}, \tag{5}$$

where $f(\mathbf{t})$ is the minimum of the objective function of [13] for the keyframe times $\mathbf{t} = [t_2, t_3, ..., t_M]$ (t_1 is always 0 and not optimized) and w is a user specified weight factor. N is the number of discrete time steps and an implicit decision variable as it depends on the last keyframe time.

Intuitively, setting the weight w allows users to trade-off smooth but long with aggressive but short trajectories (in time). For example, setting $w > max(D^3 x_i)$ (the maximum jerk in a single time step), would force the quadrotor to fully exhaust its force limits in each time step. Making N an optimization variable and including weight w for each discretized step prevents degenerate solutions of infinitely long trajectories, where the optimization adds steps with $D^3 x_i \approx 0$ which are free with respect to the optimization's objective. In case users want to optimize the segment timings of fixed length trajectories, the formulation also allows to remove the last keyframe t_m from Eq. (4) and set w to zero (following [22]). Eq. (4) is solved via gradient descent. The directional derivatives for each keyframe denoted by g_i are computed numerically

$$\nabla_{g_i} f = \frac{f(\mathbf{t} + h g_i) - f(\mathbf{t})}{h},$$

where h is a small number and g_i is constructed in such a way that the ith element is 1 and all other elements are 0. By summing up the directional derivatives $\nabla_{g_i} f$ of all keyframes we compute the gradient $\nabla_g f$. We then perform gradient descent via line-search on the optimization problem of Eq. (4), enforcing its constraint Eq. (5). Figure 5 illustrates the effect of this time optimization by comparing our approach with the standard method. For the same set of keyframes and optimization weights as well as a fixed trajectory end time, our method adjusts the timings such that larger positional distances in-between keyframes are reflected by larger temporal distances. This leads to a better positional fit with the reference x, y, z-coordinates of the keyframes (e.g., see z-coordinate of k_3). To compare smoothness between both methods quantitatively, we calculate the accumulated jerk of both trajectories normalized by the horizon length[4]. This measures is smaller for our

[3] See video from 2:54 min to 3:08 min.

[4] Minimizing jerk is common practice to smoothen motion (cf. [11])

Figure 6: Visual results of our method. Top: snapshots from planning tool. Bottom: corresponding results from real quadrotor.

method (ours: $1.73\frac{m}{s^3}$, [13]: $2.63\frac{m}{s^3}$), indicating a smoother camera motion. Note that the global time optimization prevents real-time performance. However, it is fast enough to be employed in the user study.

Visual Results

We evaluate the functionality of our system qualitatively by designing a number of aerial video shots and executed the resulting plans on a real quadcopter (unmodified Parrot Bebop 2). Figure 6 shows selected frames from the preview and resulting footage (cf. accompanying video).

EVALUATION

To better understand the effectiveness of particular UI- and optimization scheme features in terms of supporting end-users in the creation of aerial footage, we conduct a preliminary user study where we evaluate two variants of our system and Horus [16] (see Figure 2). This tool was chosen as representative of the-state-of-the-art, since other work either solely focuses on the optimization aspects of quadrotor camera tools [13] or is not available as open-source [27].

Participants: Twelve participants (5 female, 7 male) were recruited from our institution (students and staff). The average age was 25.3 (SD = 3.1, aged 19 to 32). We included one expert, working part-time as a professional quadrotor operator, the remaining participants reported no considerable experience in aerial nor normal photo- or videography. Five participants reported prior experience with 3D games, four had limited experience and three reported no experience.

Experimental conditions: We investigate *Horus* and two variants of WYFIWYG. The first variant takes keyframes from the basic snapshot-mode as input (*snapshot*). In the second variant, users directly specify the camera path (equidistant keyframe sampling) (*virtual-flight*). *Horus* is controlled via mouse and keyboard, whereas *snapshot* and *virtual-flight* are controlled using a gamepad. We use a within-subjects design with fully counterbalanced order of presentation to compensate for learning effects.

Tasks: The study comprises two tasks: 1) Participants were asked to faithfully reproduce an aerial video shot shown to them by the experimenter (T1). The shot was designed with the help of an expert as a shot only possible with airborne camera. 2) Participants were asked to design a video of their liking with a maximum duration of one minute (T2).

Procedure: In the beginning, participants were introduced to the systems and asked to design a short video in each condition. During this tutorial they could ask the experimenter for help. After that participants first solved T1 and then T2, each in all conditions. Both tasks were completed when participants reported to be satisfied with the similarity to the reference (T1) or the designed video (T2). Participants were encouraged to think aloud. For each task and condition participants completed the NASA-TLX and a questionnaire on satisfaction with the result and the system. At the end an exit interview was conducted. A session took on average 92 min (SD = 29 min) (tutorial $\approx$ 26 min, T1 $\approx$ 29 min, T2 $\approx$ 22 min).

RESULTS

Here we discuss quantitative results of our study (for further results see Appendix B). Following [5, 8], we abstain from null hypothesis significance testing and report interval estimates[5]. We test conditions according to the findings of the expert interviews and analyze their usability and user experience.

Target Framing

In T1 we asked participants to reproduce a given video. The idea is that by setting the reference and comparing video similarity, we are able to reveal potential advantages and drawbacks of the different target framing approaches used in our conditions. To quantitatively assess similarity of videos from T1 we compare resulting trajectories with the reference. Due to differences in underlying algorithms, we only compare trajectory positions and not their dynamics. We normalize the length of all trajectories to the duration of the reference. Figure 7 plots the average trajectories by UI in comparison to the reference. The inset summarizes position and orientation error over all trajectories and users. Initially, participants perceived *virtual-flight* as difficult to control. However, on average this mode produces the closest positional match with the lowest mean error and the tightest CI. It is followed by *Horus* and *snapshot*. For the angular error *Horus* and *snapshot* have the

[5] standard deviation = SD, 95% confidence interval = CI.

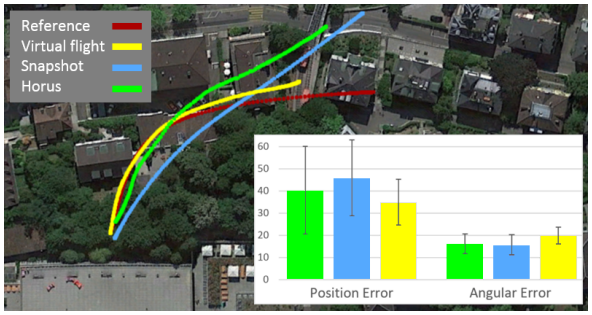

Figure 7: Visualization of the average trajectory of each condition and the reference. Inset shows average errors and CIs.

best result followed by *virtual-flight*. Figure 8 shows participant responses on perceived similarity to the reference video on a scale from 1 (very different) to 7 (very similar). Comparing means and confidence intervals in between all conditions for positional and angular error as well as for perceived similarity, no significant quantitative differences in target framing can be determined for the given task. Nevertheless, using *Horus* two participants mentioned their struggle with unintended camera tilt and non-smooth camera motion as effects of optimizing target framing based on look-at positions (referring to section Method, Target Framing). Both were not able to generate the video they intended to design.

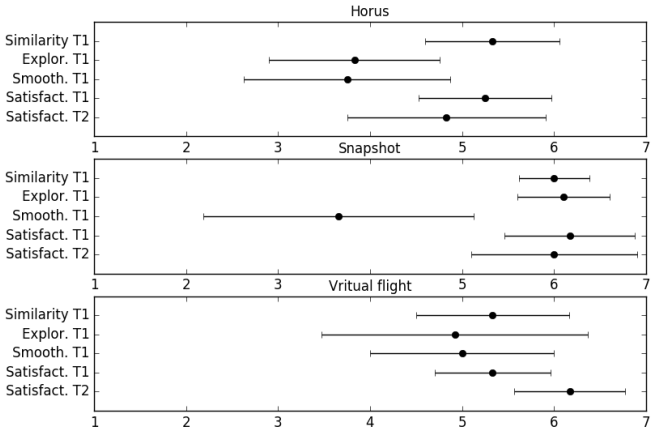

Figure 8: Visualizing participant responses and their CIs.

Smooth Camera Motion

Figure 8 summarizes rankings of the perceived smoothness on a scale ranging from 1 (non-smooth) to 7 (very smooth). Our participants regularly adjusted the timing of shots to attain smooth camera motion. As expected, several participants (not the expert) had problems to attain globally smooth camera motion paths. They were not able to position keyframes such that the ratio of distance in time to distance in space is similar, resulting in non-smooth footage (see video from 3:54 to 4:08 min). In this context, observations and participants thinking-aloud revealed that most of them expected the optimization to generate smooth camera motion over all specified keyframes. However, only few used the global time optimization, which actually provided this functionality. This may be due to (i) the longer runtime of the procedure and (ii) this being an on-demand feature and participants may not have been aware of it (although shown in the tutorial). The two participants that did use the feature were very positive about its utility in particular after discovering that with this method fewer keyframes are necessary to achieve appealing videos. Both used the segment times optimization such that the temporal length of the original and the time-optimized motion path stays the same. Still, jerk and angular jerk of the time-optimized trajectory is smaller in both cases, compared to the trajectory generated by using unmodified [13] (see Table 1), quantitatively verifying smoother camera motion.

Participant	Method	Jerk ($\frac{m}{s^3}$)	Angular jerk ($\frac{°}{s^3}$)
1	[13]	0.07	2.29
	time-opt.	0.06	0.04
2	[13]	1.15	4.01
	time-opt.	0.74	3.44

Table 1: Comparison of jerk and angular jerk for trajectories generated with [13] and with our time optimization.

Exploration

To assess support for freeform exploration, we logged the camera positions over all participants in T1. This is visualized as heatmap in Figure 9, clearly showing that participants cover more ground and experiment more in both WYFIWYG conditions than with Horus. This is also reflected in the participants

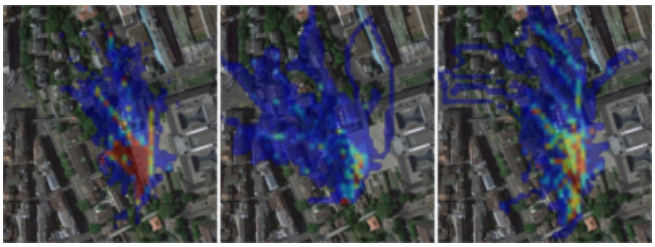
Figure 9: Heatmap of camera positions in *Horus* (left), *snapshot* (middle) and *virtual-flight* (right).

perception. On a scale from 1 (does not encourage exploration) to 7 (strongly encourages exploration), they rated *snapshot* first, followed by *virtual-flight* and *Horus* (cf. Figure 8). Users commented that being able to evaluate different perspectives quickly helped in solving T1 as they could better match which views were present in the reference.

Usability

To asses usability differences between the three tools we asked our participants to fill out the NASA-TLX questionnaire. Looking at the NASA-TLX scores, summarized in Table 2, we see lower task load scores for the WYFIWYG conditions. Since the large majority of interactions are due to camera positioning (8122 (cam) vs 99 (rest) avg per participant and task), a lower task load can be linked to better camera controls. Interesting to see is the drop-off in task load and the growing result satisfaction over the two tasks for the *virtual-flight* condition, suggesting a steep learning curve for this mode. The lower task load of WYFIWYG conditions is

Task	Horus	Snapshot	Virt. flight
1	37.4±10.3	**26.9±8.3**	36.3±10.6
2	38.5±12.2	**22.2±6.3**	25.7±5.9

Table 2: N.-TLX scores per task with CI-ranges (bold is best).

also supported by lower execution times of T1 in *snapshot* (476.25 sec, SD = 398.74) and *virtual-flight* (584.5 sec, SD = 432.82), compared to *Horus* (669.25 sec, SD = 471.18).

DISCUSSION

In this chapter we discuss the findings of work, summarized as implications for the design of UIs and optimization schemes of future quadrotor camera tools. We split the discussion into UI and optimization related aspects. Participant statements come from the exit interview and the thinking aloud protocol.

UI Design

Visualizing and manipulating the camera path: Our general idea of setting the focus on the video content rather than the trajectory was appreciated by our participants with statements like *"in WYFIWYG I think more about what I can do with the camera because I see what it is seeing"*, or *"[...]in WYFIWYG you focus more on the shot"*. One participant also commented positively on the simplicity of WYFIWYG implying that a single view reduces levels of abstraction: *"In Horus you need to abstract more, you need to think where you are in space. With WYFIWYG it's more intuitive"*. Nevertheless, 9 out of 12 participants mentioned the need for a 2D-map like in Horus. They highlighted its importance to identify discrepancies of distances in time and space or to specify straight movements in-between keyframes. Horus' feature of visualizing the camera motion on progress curves caused contradicting reactions. While some participants perceived them as complicated, others (e.g. the expert) appreciated the workflow they enable, setting camera positions first and then adjust their timing to achieve intended dynamics. We propose that future quadrotor camera tools should implement the 3D view as main component of the user interface but also need to provide a 2D map, e.g. as a world-in-miniature rendering (as proposed by participants). In addition, providing progress curves as on-demand feature allows experienced users to manually fine tune camera dynamics while novices are not deterrent by their complexity.

Virtual flight: Similar accurate results compared to other conditions in T1 and better results in terms of smooth camera motion indicate the value of adjusting target framing continuously. Participants valued the fact that with *virtual flight* they have full control on camera motion: *"In virtual flight I always knew what will happen"*. This positive view was shared by the expert participant: *"Its nice that I can specify movements and that I don't need to think in terms of keyframes and what to do next"*. Nevertheless, the high task load scores of this mode in T1 show that practice is necessary in order to use it. Therefore, we propose that future quadrotor camera tools should provide *virtual flight* in addition to a keyframe-based camera path specification approach.

Integrated camera control: We argue that the lower task load values of WYFIWYG conditions compared to Horus are mainly caused by the difference in virtual camera control. In addition, we assume that the better exploratory behavior of WYFIWYG conditions is largely due to the integrated camera control as it gamifies interaction. This was also perceived by participants who commented on using WYFIWYG with *"feels like a game"* or *"is like playing a video game"*. Therefore, we propose that future quadrotor camera tools should provide integrated positional and rotational camera control.

Optimization Scheme Design

Target framing: Undesired camera tilt and non-smooth camera motion due to generating the camera orientation based on look-at positions (referring to section Method, Target Framing) became a problem for two participants. Therefore, we suggest that quadrotor camera tools optimize camera orientations based on reference angles instead of look-at positions.

Global smoothness: Existing methods do not optimize the timing of keyframes causing users difficulties in specifying smooth camera motion over an entire sequence. Our observations indicate that most participants did not think about keyframes in space and time, but expected the underlying method to automatically generate globally smooth camera motion over all specified spatial positions. The method proposed in this paper somewhat achieves this goal but long optimization runtimes prevented adaption. We think that reformulating the quadrotor camera trajectory optimization problem to automatically generate timings such that the camera moves smoothly through all user-specified positions would be a more user-friendly approach. This could be implemented by optimizing progress on a time-free trajectory subject to a quadrotor's model, similar to [24]. Please note that this does not conflict with the requirement of giving users precise timing control, established in [16]. The suggested workflow is to produce a feasible trajectory with generated timings. These timings should then be editable via progress curves or other means, with a second optimization method guaranteeing that the trajectory remains feasible or returning the closest feasible match (cf. [27]). Investigating the potential of such a method poses an interesting direction for future work.

CONCLUSION

In this paper we investigate how to improve end-user support in quadrotor camera tools. We highlight important aspects for the creation of aesthetically pleasing aerial footage, revealed in formative expert interviews. Based on these results, we design a new quadrotor camera tool, WYFIWYG, and develop extensions to an existing trajectory generation algorithm that allow for the generation of more intuitive angular camera motion and globally smooth trajectories over a sequence of keyframes. To better understand the effectiveness of particular UI- and optimization scheme features in terms of user support, we conduct an exploratory user study evaluating variants of our system and [16]. The study revealed that current tools complicate the design of globally smooth video shots by requiring users to specify keyframes at equidistant points in time and space. We conclude by discussing the findings of work and summarizing them as implications for the design of UIs and optimization schemes of future quadrotor camera tools.

APPENDIX A - APPROXIMATE QUADROTOR MODEL AND TRAJECTORY GENERATION

For algorithmic motion plan generation a model of the quadrotor and its dynamics are needed. Incorporating a fully nonlinear model results in a high computational cost and negates convergence guarantees [22]. Following [13] we use a linear approximation, modelling the quadrotor as a rigid body, described by its mass and moment of inertia along the world frame z-axis (i.e. pitch and roll are fixed):

$$m\ddot{\mathbf{r}} = \mathbf{F} + m\mathbf{g} \in \mathbb{R}^3 \qquad (6)$$
$$I_\psi \ddot{\psi_q} = M_\psi \in \mathbb{R},$$

where $\mathbf{r}$ is the center of mass, ψ_q is the yaw angle, m is the mass of the quadrotor, I_ψ is the moment of inertia about the z-axis, $\mathbf{u}_r$ is the the force acting on $\mathbf{r}$ and M_ψ is the torque along z.

To ensure that robot and gimbal can reach specified positions and camera orientations within a given time and without exceeding the limits of the quadrotor hardware, bounds on maximum force and torque are introduced:

$$\mathbf{u}_{min} \leq \mathbf{u} \leq \mathbf{u}_{max} \in \mathbb{R}^4, \qquad (7)$$

where $\mathbf{u} = [\mathbf{F}, M_\psi]^T$ is the input to the system. Details on how to choose the linear bounds can be found in [13]. This quadrotor model is reformulated as a first-order dynamical system and discretized in time with a time-step Δt assuming a zero-order hold strategy, i.e. keeping inputs constant in between stages:

$$\mathbf{x}_{i+1} = A_d \mathbf{x}_i + B_d \mathbf{u}_i + c_d, \qquad (8)$$

where $\mathbf{x}_i = [\mathbf{r}, \psi, \dot{\mathbf{r}}, \dot{\psi}]^T \in \mathbb{R}^4$ is the state and $\mathbf{u}_i$ is the input of the system at time $i\Delta t$. The matrix $A_d \in \mathbb{R}^{8x8}$ propagates the state $\mathbf{x}$ forward by one time-step, the matrix $B_d \in \mathbb{R}^{8x4}$ describes the effect of the input $\mathbf{u}$ on the state and the vector $c_d \in \mathbb{R}^8$ that of gravity after one time-step.

The algorithm takes M positions k_j at a specific time $\eta(j)\Delta t$ as input, where $\eta : \mathbb{N} \to \mathbb{N}$ maps between keyframe indices and corresponding time-point. Time is discretized into N stages with stepsize Δt over the whole time horizon $[0, t_f]$. The variables which are optimized are the quadrotor state x_i and the inputs u_i to the system Eq. (8) at each stage $i\Delta t$. For the camera motion to follow the user-specified positions as closely as possible, we seek to minimize the following cost

$$E^k = \sum_{j=1}^{M} ||r_{\eta(j)} - k_j||^2. \qquad (9)$$

A small residual of E^k indicates a good match of the generated quadrotor position and the specified keyframe. Furthermore, we wish to generate smooth motion, which is related to the derivatives of the quadrotor's position. To this end we introduce a cost for penalizing higher position derivatives

$$E^d = \sum_{i=q}^{N} ||D^q \begin{bmatrix} x_i \\ \dots \\ x_{i-q} \end{bmatrix} ||^2, \qquad (10)$$

where D^q is a finite-difference approximation of the q-th derivative over the last q states. The combined cost $E = \lambda_k E^k + \lambda_d E^d$ with weights $\lambda_{k|d}$ is a quadratic function, enabling us to formulate the trajectory generation problem as a quadratic program.

$$\begin{aligned} &\underset{X}{\text{minimize}}\ \frac{1}{2} X^T H X + f^T X \qquad (11) \\ &\text{subject to } A_{ineq} X \leq b_{ineq} \\ &\text{and } A_{eq} X = b_{eq}, \end{aligned}$$

where X denotes the stacked state vectors x_i and inputs u_i for each time-point, H and f contain the quadratic and linear cost coefficients respectively which are defined by Eq. (9) and Eq. (10) , A_{ineq}, b_{ineq} comprise the linear inequality constraints of the inputs Eq. (7) and A_{eq}, b_{eq} are the linear equality constraints from our model Eq. (8) for each time-point $i \in 1, \dots, N$. This problem has a sparse structure and can be solved by most optimization software packages.

APPENDIX B - UEQ SCORES AND TOOL PREFERENCE

We also asked participants to fill out the User Experience Questionnaire (UEQ). Its scores reveal a distinct ranking in between conditions. *Snapshot* ranks first on all dimensions, followed by *virtual flight* and *Horus* (see Table 3). Reasoning about the cause of the scores is difficult. We assume that the higher level of attractiveness of the WYFIWYG-conditions is caused by the simplicity of the UI, having a single view to design the video. The better efficiency scores of the WYFIWYG-conditions are likely caused by the integrated camera control. Finally, we asked participants which condition they prefer. 9 out of 12 participants preferred WYFIWYG (6×*snapshot*, 2×*virtual-flight*, 1×either) with the remaining 3 stating equal preference for Horus and one of the WYFIWYG conditions.

Dimension	Horus	Snapshot	Virtual flight
Attractiveness	0.35±0.64	**1.91±0.39**	1.19±0.72
Perspicuity	-0.29±0.61	**2.0±0.32**	1.48±0.59
Efficiency	0.42±0.54	**1.52±0.41**	1.13±0.59
Dependability	0.56±0.54	**1.38±0.43**	0.52±0.59
Stimulation	0.73±0.49	**1.63±0.5**	1.4±0.53
Novelty	0.25±0.86	**1.31±0.62**	1.13±0.56

Table 3: UEQ dimension scores with CI-ranges (bold is best).

REFERENCES

1. APM. 2016. APM Autopilot Suite. (2016). Retrieved September 13, 2016 from http://ardupilot.com
2. Ty Audronis. 2014. How to Get Cinematic Drone Shots. (2014). Retrieved August 29, 2017 from https://www.videomaker.com/article/c6/17123-how-to-get-cinematic-drone-shots
3. Jessica R. Cauchard, Jane L. E, Kevin Y. Zhai, and James A. Landay. 2015. Drone and Me: An Exploration into Natural Human-drone Interaction. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 361–365. DOI: http://dx.doi.org/10.1145/2750858.2805823

4. Marc Christie, Patrick Olivier, and Jean Marie Normand. 2008. Camera control in computer graphics. *Computer Graphics Forum* 27, 8 (2008), 2197–2218. DOI: http://dx.doi.org/10.1111/j.1467-8659.2008.01181.x
5. Geoff Cumming. 2014. The New Statistics: Why and How. *Psychological Science* 25, 1 (2014), 7–29. DOI: http://dx.doi.org/10.1177/0956797613504966
6. T.J. Diaz. 2015. Lights, drone... action. *Spectrum, IEEE* 52, 7 (July 2015), 36–41. DOI: http://dx.doi.org/10.1109/MSPEC.2015.7131693
7. DJI. 2016. PC Ground Station. (2016). Retrieved September 13, 2016 from http://www.dji.com/pc-ground-station
8. Pierre Dragicevic. 2016. Fair statistical communication in HCI. In *Modern Statistical Methods for HCI*. Springer, 291–330. DOI: http://dx.doi.org/10.1007/978-3-319-26633-6_13
9. Steven M. Drucker and David Zeltzer. 1994. Intelligent Camera Control in a Virtual Environment. In *Proceedings of Graphics Interface '94*. 190–199. DOI: http://dx.doi.org/10.1109/SIBGRA.2002.1167167
10. Jane L. E, Ilene L. E, James A. Landay, and Jessica R. Cauchard. 2017. Drone and Wo: Cultural Influences on Human-Drone Interaction Techniques. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 6794–6799. DOI: http://dx.doi.org/10.1145/3025453.3025755
11. Tamar Flash and Neville Hogan. 1985. The coordination of arm movements: an experimentally confirmed mathematical model. *The journal of Neuroscience* 5, 7 (1985), 1688–1703.
12. Q. Galvane, J. Fleureau, F. L. Tariolle, and P. Guillotel. 2016. Automated Cinematography with Unmanned Aerial Vehicles. In *Proceedings of the Eurographics Workshop on Intelligent Cinematography and Editing (WICED '16)*. Eurographics Association, Goslar Germany, Germany, 23–30. DOI: http://dx.doi.org/10.2312/wiced.20161097
13. Christoph Gebhardt, Benjamin Hepp, Tobias Nägeli, Stefan Stevšić, and Otmar Hilliges. 2016. Airways: Optimization-Based Planning of Quadrotor Trajectories According to High-Level User Goals. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 2508–2519. DOI: http://dx.doi.org/10.1145/2858036.2858353
14. John Hennessy. 2015. 13 Powerful Tips to Improve Your Aerial Cinematography. (2015). Retrieved August 29, 2017 from https://skytango.com/13-powerful-tips-to-improve-your-aerial-cinematography/
15. Niels Joubert, Dan B Goldman, Floraine Berthouzoz, Mike Roberts, James A Landay, Pat Hanrahan, and others. 2016. Towards a Drone Cinematographer: Guiding Quadrotor Cameras using Visual Composition Principles. *arXiv preprint arXiv:1610.01691* (2016).
16. Niels Joubert, Mike Roberts, Anh Truong, Floraine Berthouzoz, and Pat Hanrahan. 2015. An Interactive Tool for Designing Quadrotor Camera Shots. *ACM Trans. Graph.* 34, 6, Article 238 (Oct. 2015), Article 238, 11 pages. DOI: http://dx.doi.org/10.1145/2816795.2818106
17. Tsai-Yen Li and Chung-Chiang Cheng. 2008. Real-Time Camera Planning for Navigation in Virtual Environments. In *Smart Graphics*, Andreas Butz, Brian Fisher, Antonio Krger, Patrick Olivier, and Marc Christie (Eds.). Lecture Notes in Computer Science, Vol. 5166. Springer Berlin Heidelberg, 118–129. DOI: http://dx.doi.org/10.1007/978-3-540-85412-8_11
18. Christophe Lino and Marc Christie. 2012. Efficient Composition for Virtual Camera Control. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '12)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 65–70. DOI: http://dx.doi.org/10.1145/1409060.1409068
19. Christophe Lino and Marc Christie. 2015. Intuitive and Efficient Camera Control with the Toric Space. *ACM Trans. Graph.* 34, 4, Article 82 (July 2015), 12 pages. DOI: http://dx.doi.org/10.1145/2766965
20. Christophe Lino, Marc Christie, Roberto Ranon, and William Bares. 2011. The Director's Lens: An Intelligent Assistant for Virtual Cinematography. In *Proceedings of the 19th ACM International Conference on Multimedia (MM '11)*. ACM, New York, NY, USA, 323–332. DOI: http://dx.doi.org/10.1145/2072298.2072341
21. Kexi Liu, Daisuke Sakamoto, Masahiko Inami, and Takeo Igarashi. 2011. Roboshop: Multi-layered Sketching Interface for Robot Housework Assignment and Management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 647–656. DOI: http://dx.doi.org/10.1145/1978942.1979035
22. Daniel Mellinger and Vijay Kumar. 2011. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2520–2525. DOI: http://dx.doi.org/10.1109/ICRA.2011.5980409
23. Mark Mueller and Raffaello D'Andrea. 2013. A model predictive controller for quadrocopter state interception. In *2013 European Control Conference (ECC)*. IEEE, 1383–1389.
24. Tobias Nägeli, Lukas Meier, Alexander Domahidi, Javier Alonso-Mora, and Otmar Hilliges. 2017. Real-time Planning for Automated Multi-view Drone Cinematography. *ACM Trans. Graph.* 36, 4, Article 132 (July 2017), 10 pages. DOI: http://dx.doi.org/10.1145/3072959.3073712

25. T. Ngeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges. 2017. Real-Time Motion Planning for Aerial Videography With Dynamic Obstacle Avoidance and Viewpoint Optimization. *IEEE Robotics and Automation Letters* 2, 3 (July 2017), 1696–1703. DOI: http://dx.doi.org/10.1109/LRA.2017.2665693

26. A Richards and J How. 2004. Decentralized model predictive control of cooperating UAVs. In *43rd IEEE Conference on Decision and Control (CDC)*. IEEE, 4286–4291 Vol.4. DOI: http://dx.doi.org/10.1109/CDC.2004.1429425

27. Mike Roberts and Pat Hanrahan. 2016. Generating Dynamically Feasible Trajectories for Quadrotor Cameras. *ACM Trans. Graph.* 35, 4, Article 61 (July 2016), 11 pages. DOI: http://dx.doi.org/10.1145/2897824.2925980

28. Daisuke Sakamoto, Koichiro Honda, Masahiko Inami, and Takeo Igarashi. 2009. Sketch and Run: A Stroke-based Interface for Home Robots. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 197–200. DOI: http://dx.doi.org/10.1145/1518701.1518733

29. D. H. Shim, H. J. Kim, and S. Sastry. 2003. Decentralized nonlinear model predictive control of multiple flying robots. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, Vol. 4. 3621–3626 vol.4. DOI: http://dx.doi.org/10.1109/CDC.2003.1271710

30. VC Technology. 2016. Litchi Tool. (2016). Retrieved September 13, 2016 from https://flylitchi.com/

31. I-Cheng Yeh, Chao-Hung Lin, Hung-Jen Chien, and Tong-Yee Lee. 2011. Efficient camera path planning algorithm for human motion overview. *Computer Animation and Virtual Worlds* 22, 2-3 (2011), 239–250. DOI: http://dx.doi.org/10.1002/cav.398

32. Shumin Zhai and Paul Milgram. 1998. Quantifying coordination in multiple dof movement and its application to evaluating 6 DOF input devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '98)*. ACM, New York, NY, USA, 320–327. DOI: http://dx.doi.org/10.1145/274644.274689

33. Shengdong Zhao, Koichi Nakamura, Kentaro Ishii, and Takeo Igarashi. 2009. Magic Cards: A Paper Tag Interface for Implicit Robot Control. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 173–182. DOI: http://dx.doi.org/10.1145/1518701.1518730