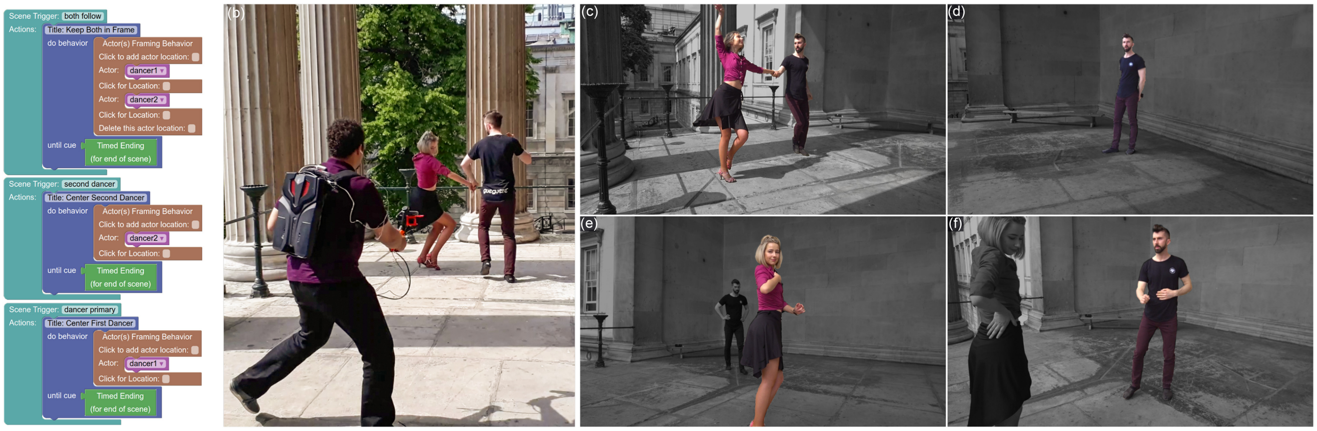
LookOut! Interactive Camera Gimbal Controller for Filming Long Takes

MOHAMED SAYED, ROBERT CINCA, ENRICO COSTANZA, and GABRIEL BROSTOW, University College London, United Kingdom



Scene Trigger: both follow
Actions: Title: Keep Both in Frame
do behavior: Actor(s) Framing Behavior
Click to add actor location: □
Actor: dancer1
Click for Location: □
Actor: dancer2
Click for Location: □
Delete this actor location: □
until cue: Timed Ending (for end of scene)

Scene Trigger: second dancer
Actions: Title: Center Second Dancer
do behavior: Actor(s) Framing Behavior
Click to add actor location: □
Actor: dancer2
Click for Location: □
until cue: Timed Ending (for end of scene)

Scene Trigger: dancer primary
Actions: Title: Center First Dancer
do behavior: Actor(s) Framing Behavior
Click to add actor location: □
Actor: dancer1
Click for Location: □
until cue: Timed Ending (for end of scene)

(a) Script Creation

(b-f) Interactive Filming

Fig. 1. LookOut can take over the task of controlling where the camera is pointing when a camera operator is overwhelmed with other duties on the go, dynamically changing the camera's behavior based on where actors are, how a scene progresses, and what the camera operator instructs it to do. (b) The user-worn LookOut rig consists of a light backpack computer, a handheld motorized gimbal, dual cameras (normal and wide view), earphones, a lapel microphone, and a joystick for initial setup. Before filming, the LookOut GUI (a) enables a user to pre-script where the camera should point and its focal length. This involves creating camera behavior blocks that can be chained together to make scripts, callable during filming. A behavior can be as simple as a pan or as complex as positioning multiple subjects in different parts of the frame, and they can be sequenced with scene-specific cues. On boot, LookOut guides the operator, through text-to-speech, to enroll actor identities to its visual tracker, perform scene-specific initialization, and calibrate audio. (c–f) Four frames from a LookOut-captured video, but with false-coloring to visualize which actor(s) LookOut is dynamically framing via its motorized gimbal to satisfy the operator's currently selected script. At the user's instruction, LookOut frames (c) both dancers, then (d) orients the gimbal to center on the male, then the female (e), and back to the male (f). The user receives audio feedback when switching between camera behaviors. Without a field monitor, users can watch where they are going while trusting our controller to handle their dynamic requests.

The job of a camera operator is challenging, and potentially dangerous, when filming long moving camera shots. Broadly, the operator must keep the actors in frame while safely navigating around obstacles and while fulfilling an artistic vision. We propose a unified hardware and software system that distributes some of the camera operator's burden, freeing the operator up to focus on safety and aesthetics during a take. Our real-time system provides solo operators with end-to-end control so that they can balance on-set responsiveness to action against planned storyboards and framing while looking where they are going. By default, we film without a field monitor.

Our LookOut system is built around a lightweight commodity camera gimbal mechanism, with heavy modifications to the controller, which would normally just provide active stabilization. Our control algorithm reacts to speech commands, video, and a premade script. Specifically, our automatic monitoring of the live video feed saves the operator from distractions. In preproduction, an artist uses our graphical user interface (GUI) to design a sequence of high-level camera "behaviors." Those can be specific, based on a storyboard, or looser objectives, such as "frame both actors." Then, during filming, a machine-readable script, exported from the GUI, ties together with the sensor readings to drive the gimbal. To validate our algorithm, we compared tracking strategies, interfaces, and hardware protocols and collected impressions from (a) filmmakers who used all aspects of our system and (b) filmmakers who watched footage filmed using LookOut.

CCS Concepts: • **Computer systems organization → Robotics;**

Additional Key Words and Phrases: Cinematography, videography, video editing, camera gimbal

Wenbin Li created the inspirational re-localizing gimbal-controlled camera under UK EPSRC's EP/K023578/1. Mohamed is funded by a Microsoft Research PhD Scholarship. This work is supported by the UK EPSRC grant EP/R513143/1 for the UCL Interaction Centre (UCLIC). Our study was approved by the UCLIC Ethics Committee (UCLIC/1617/017).

Authors' addresses: M. Sayed and G. Brostow, University College London, 169 Euston Road, London, United Kingdom, NW1 2AE; emails: mohamed.sayed.17@ucl.ac.uk, g.brostow@cs.ucl.ac.uk; R. Cinca and E. Costanza, University College London, 66-72 Gower Street, London, United Kingdom, WC1E 6EA; emails: {robert.cinca.14, e.costanza}@ucl.ac.uk.

0730-0301/2022/03-ART30 $15.00
https://doi.org/10.1145/3506693

ACM Reference format:
Mohamed Sayed, Robert Cinca, Enrico Costanza, and Gabriel Brostow. 2022. LookOut! Interactive Camera Gimbal Controller for Filming Long Takes. *ACM Trans. Graph.* 41, 3, Article 30 (March 2022), 16 pages.
https://doi.org/10.1145/3506693

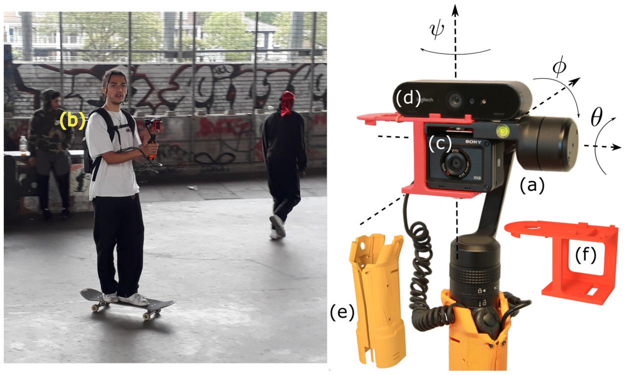


Fig. 2. A novice camera operator filming using the LookOut system: (a) An existing active camera gimbal, designed to stabilize mobile-phone filming. The mini-joystick is inactive by default, The orange 3D-printed handle channels the cables and protects the USB connectors from being bumped. (b) The backpack computer, connected to the gimbal by one USB cable and connected to (d) with another. Not shown, the backpack also has headphones and a lapel mic for two-way speech communication with the operator. (c) The primary "star" camera, recording high-quality footage to local memory. (d) The guide-camera, which has a wider field of view than (c), and whose video is fed to the backpack computer for real-time analysis. Star camera frame axes are represented with pitch (θ), roll (ϕ), and yaw (ψ). The LookOut controller drives the orientation of the camera assembly. (e) Gimbal handle enclosure to allow for wire pass-through and a comfortable grip. (f) Camera assembly engineered for balance and alignment of camera optical axes.

1 INTRODUCTION

Filming for journalism and movies is a creative and often collaborative process in which the budget dictates whether the roles of director, director of photography (DP), and camera operators are fulfilled by a team or rest on just one person's shoulders. Ultimately, the person holding the camera has the responsibility of delivering both the content and style that was agreed to in advance while safely adapting to dynamic changes on set.

After budget, time is the next biggest constraint. We consider two types of filming scenarios: one in which a journalist or documentary maker must catch a one-off unrepeatable event and the other in which actors and crew follow a storyboard with blocking, repeating the performance until the director is satisfied with it. Our system, called "LookOut," is designed to help with both types of filming if the aim is to capture a long take with a moving camera.

Long takes stand out as noteworthy and complex to choreograph in big-budget films,[1] though they are common for journalism, documentaries, and run-and-gun videos —the majority of work done by video/cinematographers. Moving the camera helps keep long takes interesting for the viewer [4, 31, 45]. Steadicams [5] and camera gimbals [55, 62] aid in filming these scenes by keeping the camera steady. Steadicams isolate translational motion through springs and arms and camera gimbals mainly isolate a camera from rotational movement of the carrier assembly by suspending the camera on a pivoted support with often motorized orthogonal axes. However, moving cameras and moving people stretch the attention of camera operators, who are trying to simultaneously walk about and adequately frame their stars. Usain Bolt was famously run over by a cameraman who suffered from task overload while steering a Segway at the World Athletics Championships in 2015.

Speaking informally with independent filmmakers, we found that there was some interest in drone cinematography systems such as those in [18, 50, 70], but a strong desire for three things: (1) to have interactive control while filming, (2) a system that tracks indoors and outdoors without special costumes, and (3) ideally, to work with lightweight handheld hardware because drones are prohibited in many populated areas and most countries require a pilot's license. This seeded our research process, which, with feedback and validation from filmmakers, has led to our proposed LookOut system (Figure 1).

The overall LookOut system serves as an interactive digital assistant for filming long takes with a camera gimbal. LookOut consists of software and three-dimensional (3D) printed hardware that augments an existing lightweight motorized camera gimbal ($130), with a video feed and rudimentary two-way speech interface connected to a backpack computer. Without innovations, some of the individual components existed in principle but would not integrate into a usable or responsive video-making algorithm. Therefore, our two main technical contributions are:

- A visual tracking system that detects and tracks actors robustly in real time for extended periods of time, relying on a dynamic cost formulation for tracker/detection assignment, strategies for creating and maintaining a robust and space-efficient appearance history, and a recovery mechanism for minimizing distractions when reacquiring actors after occlusion.
- A combined controller that dynamically balances script-induced constraints such as smoothness and intentional framing to reframe actors dynamically while still being responsive to tracker outputs that have inherent noise and drop-outs.

Camera operators often wear many other hats, but from their perspective, during the critical moments of filming, the LookOut system responds well to voice commands and follows alternative or sequential prespecified behaviors. It rotates and stabilizes the camera within its joint limits to follow the actors and to compensate for the operator's trajectory through the scene. For our experiments, operators did not see a monitor while filming. Thus, they were free to look around and keep one hand free as they walked, climbed, or cycled through different environments.

2 RELATED WORK

The graphics community has a long history of exploring camera placement [7] and control systems [21], striving to be automatic and cinematic. For "offline" scenery special effects, motion control

[1] See the films *1917* [47] and *Birdman* [27], both filmed to look like one take, versus Michael Bay's average shot length of 3 seconds [51].

camera systems have been used since the work of computer graphics pioneer John Whitney in the 1950s [71]. While programmable camera trajectories can help with stop-motion animation and with layered compositing of scenery and special effects, they require hiring specialized crew, are usually constrained to a short track, and the systems ignore actors and other dynamic events. Therefore, we focus this review on the context of our system: following and framing of actors in video. This includes stabilizing gimbals, visual active tracking, and efforts in drone cinematography.

2.1 Steadicam, Stabilizing Gimbals, and Active Tracking

Camera gimbals are essential for smooth video capture, especially when the whole assembly is held by a walking camera operator.

Garrett Brown invented the Steadicam [5] in 1975. The Steadicam allows a camera operator to physically move the camera and simultaneously capture smooth footage. It has been famously used in many Hollywood film productions, including *Rocky* (1976) [2], *Goodfellas* (1990) [56], and *Indiana Jones and the Temple of Doom* (1984) [59]. Steadicams provide an extra layer of isolation from the camera operator compared with gimbals in that they also dampen camera translation. Some are motorized to provide active stabilization and manual motorized control over the direction of the camera. Although the camera operator no longer has to worry about keeping the camera steady, the operator must still point the camera while moving, either electronically through a joystick or manually by rotating the camera assembly.

BaseCam Electronics [14] developed different hardware and software components for the construction of stabilizing gimbals. Their firmware offers control and flexibility over every stabilization parameter. We build on top of their BaseCam Handy gimbal, which offers 3-axis control over camera orientation. Communication to the gimbal is achieved through a serial API that allows for online control and settings changes on the fly.

Many early active tracking systems focus on surveillance applications. The pan-tilt camera control by Daniilidis et al. [9] orients a camera to focus on motion in a static scene. Dinh et al. [11] and Funahasahi et al. [16] propose multi-camera or multi-focal length camera systems for identifying pedestrians through facial recognition. These systems are among the many that actively controlled pan, tilt, and zoom.

Closest to our own hardware is the DJI Osmo Mobile [12]. It is a commercial real-time, handheld active tracker. It uses a motorized gimbal and inertial measurement units (IMUs) to control a smartphone camera's orientation. The gimbal enables the user to create stabilized camera footage and select a single object to actively track. A smartphone is used as the camera and processing unit. The tracking algorithm is not made public. Unlike our system, users have no control over framing and complex scripting, and no ability to track multiple targets.

2.2 Tracking

Generally, the ability of a tracker can be measured based on some high-level performance criteria. Among them are speed; accuracy, including robustness to ID switching — confusing another object with the target — or drift; number of trackable objects (usually one vs. many); robustness to appearance changes; and the ability to be run online. Most trackers in the literature are designed to some, but not all, of these. Our application requires robust tracking of a handful of targets for long durations (>20 minutes). Robustness to ID switches and target reacquiring after occlusion, especially in busy and cluttered environments, are crucial to our use case since a target swap during filming would very likely ruin a take and cause delays. We focus almost entirely on trackers that can approach real-time speeds.

The VOT challenges [35–38] cater to single-target tracking of any class and include benchmarks for RGBD and thermal trackers. The VOT Short-Term Challenge allows tracks to be reset, with a penalty and a timeout of five frames, to make use of the entire dataset. Trackers in the main VOT challenge are not required to deal with longer-term occlusion and confidence reporting. In our use case, actors often appear and disappear as filming progresses. While the VOT Long-Term Challenge evaluates trackers with metrics that put a greater emphasis on longer-term tracking (the average video is 2 min 04 s long and contains 10 occlusions lasting 52 frames [38]). Other tracking datasets also contain long videos with occlusion [49, 64]. Unfortunately, benchmarks on these datasets are either not maintained or trackers submitted are not required to share implementation details. These benchmarks do not run trackers in a multiple-object regime.

A family of single-object trackers are built on top of relatively lightweight Siamese network architectures [3, 63, 66, 76]. Most notably, SiamMask [66] achieves state-of-the-art performance on the VOT2018 challenge at 50 Hz. DaSiamRPN [76] includes a "distractor aware"module for reducing track loss errors after occlusion; it achieves first place on the VOT2018 real-time challenge and second place on the VOT2018 long-term tracking challenge [37, 44] at 110 Hz. We experiment with both trackers and show how they are both prone to imposters of the same object type in long takes and cluttered environments.

The MOT challenge [48] provides performance metrics on trackers for multiple people in crowded scenes. The average shot length in MOT is ~31 seconds with most targets exhibiting shorter life spans. While MOT includes metrics that measure ID swaps, resumed tracking with a new ID is still rewarded.

Among the lightweight high-scoring MOT multi-person trackers, DeepSORT [67] and MOTDT [43] stand out. Both incorporate a tracking-by-detection paradigm and use a combination of IOU and appearance costs via ReID networks for assignment. Assuming that detections are precomputed in advance, they could theoretically operate at 120 Hz and 60 Hz, respectively. In Section 6, we compare against these trackers and show that while they are capable of tracking in dense scenes with short-lived tracks, as in MOT, they are not robust to ID switches when tracking people in frame for longer videos, making them inadequate for our use case.

While these trackers offer good performance across a wide range of metrics and for different classes of objects, no one tracker satisfies all requirements of our use case, especially for people tracking.

2.3 Automatic Drone Cinematography

Though drones are contentious, with safety restrictions in many countries, we share many objectives with drone-based cinematography. Skydio [58] and DJI [13] provide multiple commercial drones with autonomous flying, self-localization, and single actor tracking capabilities.

Drone cinematography is an active area of research. Ideas explored include actor pose–driven drone flying [25], controlling subject framing autonomously [29, 50], constructing drone paths around user-defined way-points [70], learning or mimicking shot style and kinematics from expert drone pilots [1, 26], and using the Prose Storyboard Language (PSL) [54] for actor framing and plotting drone paths [18]. Although these methods either use limited UIs and/or non-visual means of actor tracking (GPS and infrared markers), they showed promise for the concept of scripted and actor-driven camera control.

While we share the excitement around drone-based filming, drones are not always the correct, safe, or perhaps even legal tool for the task. Most actor-driven shots take place in close quarters, with the camera closely following actors in the middle of the action. Further, while dubbed audio may be used in scenes, the noise they produce will ruin on-set audio.

2.4 Post-Filming Video Directing and Editing

While this work proposes getting the right framing during filming, other work focuses on either fixing framing in post or automating some or all of the editing process [10].

Many methods correct erratic and shaky camera movement in video [22, 33, 41, 46]. Grundmann et al. [22] formulate L1-optimal camera paths in handheld footage while incorporating framing constraints, notably, a constraint on incorporating important features via a relevant saliency map, for example, output from a face detector.

Gaddam et al. [17] propose a system for both real-time and offline user-controlled framing in high-resolution video. Su and Grauman [60] improve the state-of-the-art for automated 360° to narrow field of view video editing by allowing for varied style, enabling zoom, and improving computational efficiency. Gandhi et al. [19] propose a system for automatically extracting multiple clips from a single camera angle to assist editing.

Leake et al. [40] formulate a system for automatically editing together multiple takes of a dialogue-driven scene with guidance on style taken as input from the user [10]. Wright et al. [68] describe and evaluate Ed, a system for automated camera and framing selection for live events. These methods are complementary to ours.

3 LOOKOUT SYSTEM OVERVIEW

At a very high level, the proposed LookOut system lets a user specify what the user wants to track and then aims the camera gimbal at that target during filming. Achieving that aim required many iterations of hardware, software and user interfaces, especially (1) innovations in long-term visual tracking and (2) a novel control system. Here, we outline the components of the system and how they help the operator to design and safely film the long takes they want.

A solo camera operator, without specialized programming skills, uses our graphical user interface (GUI) for offline pre-production, and our rig for live filming. We consider post-production only as part of future work. Interestingly, Leake et al. [40], Wang et al. [65], and Zhang et al. [73] built interfaces that use learning to assist precisely with film editing of existing clips. Instead, through our GUI, the user defines one's intentions up front — somewhat like telling an assistant what to expect. Those intentions are saved into scripts that are later parsed by the LookOut control system during filming. On set, the camera operator wears a backpack computer (Figure 2) as the control center and sensor hub. The user also holds the camera gimbal in one hand and has dialog with the LookOut controller by wearing a microphone and headphone.

3.1 High-Level Components

We give a brief overview of these components here before providing their specifics in Section 4, Section 5, and the supplemental material.

GUI: Before filming takes place, the camera operator uses LookOut's GUI to "tell" the camera gimbal how to behave and what to expect. The behaviors are chained together into a relative timeline. Instead of absolute times, user-specified cues will conclude and then trigger each subsequent behavior in turn. Through the script file saved by the GUI, non-programmer users instruct the LookOut control system with what to look for in the audio and video sensor inputs and how to react. See the supplemental materials, in whic we show the Blockly-based LookOut GUI for designing long takes. There, we explain how non-programmer users build script files by assembling chains of behaviors. A resulting script file encapsulates how one or more actors (and even non-actors) should be framed while filming. The script file switches between behaviors when triggered by user-controlled cues that LookOut checks for continuously: Speech cues, Elapsed Time, Actor Appearance/Disappearance, Actor in Landing Zone, and Relative Actor Size. We are proud of the GUI for being easy to learn and for matching many of the wishes voiced by consulted film-makers.

System Startup and Setup: When the LookOut hardware is first switched on, the user selects which scripts to load into the system. LookOut then parses these scripts and asks the user, through guided audio feedback, to enroll actors for tracking. The user adds an actor by pointing the camera roughly in the actor's general direction and pressing a button on a small joystick. LookOut guides the user for each additional actor. The system then prompts the user to utter each script-relevant speech trigger. This ensures that all speech triggers are registered by LookOut using the user's current hardware audio configuration. LookOut informs the user that the setup is complete and remains in *Manual Mode* until the user requests *Automatic Mode*. Every mode switch and behavior trigger is met with audio feedback.

Controller: The controller reconciles the input script(s) with incoming sensor data to dynamically drive the gimbal motors. When a script sets out the camera behaviors, the controller listens for the relevant audio cues and analyzes the video feed to monitor spatial relationships between enrolled actors. It then dynamically drives the gimbal to achieve the desired framing and smoothness. Finally, it gives audio feedback to the user so that the user knows that the LookOut system is correctly following the script and the current actions. The control loop is described visually in Figure 3.

Visual Tracking: Dynamic framing of one or more actors requires our system to follow along, monitoring where people are on-screen, even when they are briefly occluded or on the edge of the field of view (FoV). For these aims, we needed a visual tracker that can detect people and distinguish between them for long periods of time despite imposter objects, for example, people or things that could resemble the main actor(s). Our tracker balances the need for accuracy against the need to feed low-latency tracks to the controller.

3.2 Hardware

Here, we describe the hardware and low-level software on which LookOut is built. See Figure 2 for a close-up of hardware.

Backpack: Our system requires low-latency feedback control in the wild. We use a VR backpack computer with a Quadcore Intel i7 7820HK CPU@2.90 GHz and a mobile NVIDIA GTX 1070 GPU. The backpack can operate for 1.5 to 2 hours, allowing for very long shots and multiple takes, and is light at 3.6 kg.

Stabilizing Gimbal: We use the BaseCam Handy gimbal to carry the camera assembly. The gimbal is programmable through a serial API and allows high-speed, low-latency control and telemetry data transfer up to 80 Hz. The gimbal has an Inertial Measurement Unit (IMU) on the camera frame assembly and an encoder for each axis for tight closed-loop feedback control. We have exclusive control over velocities on yaw (ψ), pitch (θ), and roll (ϕ) on the camera frame assembly regardless of the orientation of the handle. We disable any internal low-pass filters on velocity to ensure controllability. We tune the gimbal's internal proportional integral derivative (PID) [32] loop for the tightest possible axis velocity control while ensuring loop stability, given our camera array.

Camera: We use two cameras in our system. One serves as a guide camera for visual tracking over a 90° field of view. It operates at 60 Hz and at a resolution of 1280 × 720. We decouple roles a camera must perform by using a separate camera for capturing high-quality footage, which we call a *star camera*. This configuration was preferred by filmmakers in our initial scoping. It allows for cinematic freedom over camera parameters used for filming, without sacrificing preferred parameters and hurting the performance of the visual tracking pipeline. We design and 3D print a carrier assembly for the cameras, shown in Figure 2. It maximizes the balance on all gimbal axes while minimizing the distance between the optical centers of both cameras within the gimbal's confined space.

Remote Screen: We use a remote HDMI transmitter and screen when turning the system on. Once the system is set up, the screen is put away.

Audio: The user wears a lapel mic and earphones to speak commands to the system during filming and to receive feedback throughout actor enrollment and filming. We use an online wake word detection framework, Porcupine [52], for recognizing speech commands.

4 TRACKER

To achieve LookOut's aim of framing actors, the system needs to know their locations in screen space. The tracking component must work reliably for filming impromptu run-and-gun situations. Attaching real tags to actors, such as in [18, 50], is often impractical. To this end, the tracker must be completely visual in nature. The requirements of the tracker are that it must:

(1) be capable of locating multiple targets of interest simultaneously, with a focus on actors;
(2) reacquire actors when they appear back in frame, while being robust to ID switches; and
(3) maintain a high online refresh rate (>30 Hz) and low latency to ensure that fast actor movements are captured and acted on by the control feedback loop discussed in Section 5.

We cover the current state-of-the-art in Section 2.2. Broadly, the trackers that are fast enough (>20 Hz) fall into two categories: (1) single-object trackers aimed at the VOT [6] and OTB [69] challenges and (2) multi-target trackers from the MOT [48] challenge. We compare against the best trackers from these challenges in Section 6.1. Notably, while single-object trackers such as DaSiamRPN [76] and SiamMask [66] perform well when keeping track of an object in frame, they are prone to tracking imposters when an object is occluded and reappears in frame, not satisfying Requirement (2). To satisfy Requirement (1), a different instance of each tracker would need to run separately for each actor; this compromises Requirement (3) since the runtime now scales linearly with the number of actors.

For trackers competing in the MOT [48], almost all trackers use tracking by detection. These trackers suffer a relatively small penalty for each additional target but require a real-time detector with a good compromise of accuracy and speed.[2] However, the MOT benchmark is run on scenes whose mean length is only ~31 seconds, where targets only occasionally change view throughout their short life and rarely reappear after long-term occlusion, with a small penalty given for ID switching. We take inspiration from high-scoring trackers in the MOT benchmark, DeepSORT [67] and MOTDT [43], but add three contributions:

- a reworked cost structure for detection/track assignment, with a concentration on tracking a handful of targets robustly;
- a recovery phase and mechanism; and
- a set of lightweight, long-term appearance-encoding history-management strategies.

Our tracker relies on an appearance-encoding history for differentiating actors and other people during filming. A reliable per-actor appearance-encoding gallery is important for tracking and recovery. All three components, explained below and in pseudocode in the Supplementary Material, focus on maintaining correct IDs for each actor, especially after occlusion.

Cost Formulation and Data Association: Our tracker minimizes the the cost of assigning targets $T = t_1, \dots, t_i$, including appearance and bounding box information, to a set of detections in the current frame $D = d_1, \dots, d_j$. Taking inspiration from DeepSORT [67], we combine c^{IOU}_{ij}, the IOU bounding box cost [28, 48], with c^{f}_{ij}, the cosine distance on appearance features, derived from the Siamese network in [75]. We do not use a Kalman filter state-based cost, as detections from our choice of lightweight detector, tiny-YOLOv3, are very noisy spatially over time.

[2] MOT trackers take detection bounding boxes for granted in the benchmark.

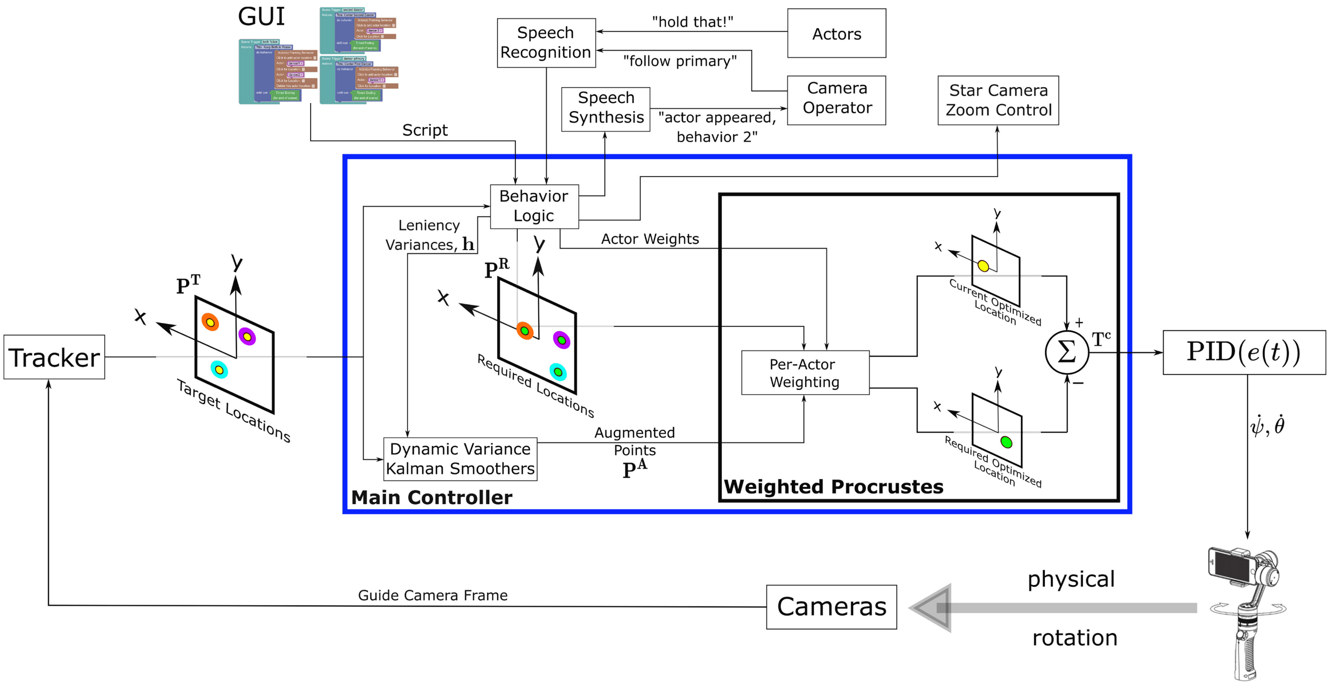


Fig. 3. High-level control loop view of how LookOut fulfills subject framing. On top, user inputs come in the form of the GUI during pre-production and through the use of speech commands on set. At the bottom, the tracker converts guide camera footage into raw tracks, $\mathbf{P_T}$. All of these inputs enter the main controller (highlighted in blue and explained in Section 5), whose job is to provide an error signal that will drive the gimbal through the PID [32] controller. By modulating process variances, $\mathbf{h}$, the controller balances between responsiveness and smoothness for one or more actors. $\mathbf{h}$ is among the outputs from behavior logic, which had access to augmented track points from the last timestep (not pictured) and current target points, $\mathbf{P_T}$. $\mathbf{h}$ helps compute the augmented points, $\mathbf{P_A}$, which go into the weighted Procrustes module (weighting explained in Section 5.2). The other main input to the Procrustes module is the required locations for each actor, $\mathbf{P_R}$. Finally, the weighted difference between required locations and augmented locations drives the gimbal update. Not seen here is a velocity fading module that fades between different velocities at the transition from one type of behavior to another.

IOU costs are useful when a target is in isolation, but useless when overlaps occur or when coming out of a long occlusion. Appearance costs, on the other hand, are crucial for re-identifying the target after long occlusion, but a collection of appearance features, capturing the appearance of the target under different lighting and self-occlusion, must be accumulated before they can be relied on. To this end, we formulate a dynamic cost structure specific to each target that emphasizes robustness by relying on IOU when no more than one detection competes for the same target and the appearance cost when a target is crowded. Nominally, the cost for associating a particular target an detection, $c(t_i, d_j)$, is

$$c(t_i, d_j) = \begin{cases} c_{ij}^{IOU} & \text{if } c_{ik}^{IOU} > \tau^{\text{overlap}} \\ c_{ij}^{\text{feature}} + c_{ij}^{IOU} & otherwise. \end{cases} \tag{1}$$

where $k \neq j\tau^{\text{overlap}}$ is the cost of assigning the track i to another detection k and is set to a high strict value to prevent ID switches when a target is occluded by other people (other ks). To further reduce target switches, a track/detection pair is deemed incompatible if either the IOU cost or the appearance cost exceed defined low maximums.

We assign c_{ij}^{f} the cost of the lowest cost match between a target's appearance encodings and that of a detection d_j. Although we take measures to exclude rogue imposter encodings, a single matched feature encoding can produce an incorrect match. To mitigate this, we take an average of the N lowest appearance costs from the target's history and we disallow a match between this combination of track and detection if it exceeds a predefined maximum.

Finally, all costs are passed along to a linear assignment step [39], where globally optimal target and detection assignments are found.

Recovery: Actors of interest will go into planned or unplanned short- and long-term occlusion throughout filming. During occlusion, the tracker must not confuse imposters with actors, and should then recover these actors when out of occlusion. We use appearance costs, c_{ij}^{f}, exclusively for this step. However, appearance encodings are temporally noisy; thus, an imposter detection might present a noisy appearance encoding in one frame that matches to a lost target. To prevent these types of false matches, we define a recovery phase that is begun when a detection is matched to a lost target. For a target to come out of recovery, it must be matched to a detection for R sequential timesteps, where R is decided dynamically. This mechanism sacrifices a few frames of tracking for recovery in the short term, but greatly improves the tracker's long-term tracking ability and its resistance to ID switching. We test our tracker without a recovery step in Table 2.

Feature History Management: In dense scenes and in a target's recovery phase, the tracker relies solely on each target's appearance encoding gallery, $\mathcal{R}_i = \{r_1, \dots, r_L\}$ for data association. Ideally, an infinitely sized history would allow for the most accurate representation of the target's appearance. However, encoding comparisons for calculating appearance costs would get expensive with longer target life cycles —10 minutes at 30 Hz yields 18,000 appearance encodings. A common solution [43, 67] is to restrict the gallery to the last L encodings. This strategy works well for short track life in short sequences, as in the MOT challenge. However, this is less successful for longer sequences, in which a target may reappear either with a different lighting or pose than when the target went into occlusion. Table 2 shows the performance of a tracker with a naïve last-L_k encodings history. We address the rapid increase in the gallery's size by selectively adding encodings to the appearance gallery on every timestep. An encoding is added only if it is sufficiently distant, via the cosine distance, from all other encodings in the gallery. This slows down the growth of the gallery by an order of magnitude and prioritizes space and time on informative encodings.

When a target is crowded by many detections, encodings produced with occluded bounding boxes might later allow an impostor to match the target incorrectly. To address this, encodings are added exclusively in normal tracking —when only one detection competes spatially for the current target. In Table 2, a tracker without this check is referred to as Faulty Encodings.

Although these steps help reduce the expansion of the gallery's size and maintain its integrity, they only delay the pruning problem when the gallery is full. Informed techniques that cluster encodings to select the most informative encodings are iterative and time-consuming in this high-dimensional space — k-means consumes 7 ms for each target. Alternatively, a simple and effective solution is to randomly sample L_k from the gallery when it is 10% larger than L_k. This has the effect of maintaining new appearances of a target while keeping a fading memory of older appearances for longer since, with every sampling step, encodings of an older age stamp are less likely to be propagated forward.

Speed: As mentioned previously, MOT provides detection for granted and trackers do not report detection time. A survey of the detection field shows that single-shot object detectors [42, 53] are best suited for their trade-off of speed and performance. We use people, cars, and bicycle detections from tiny-YOLOv3 [53] in our system. We tune the Kalman Filters used for tracking updates to reduce temporally noisy detections from tiny-YOLOv3 before being passed to any control loops down the pipeline.

Subject Enrollment: Our tracker requires one frame to enroll an actor and can track subjects immediately. An extra step can be taken to build up an initial appearance history by having the subject turn around and ideally walk once through the scene.

As shown in the Supplementary Videos, we also experimented with DaSiamRPN [76], which allows enrollment of novel objects, such as a shop window and a garden gnome.

5 CONTROL SYSTEM FOR FRAMING ACTORS

We drive the camera orientation to re-frame actors dynamically over time. The controller reconciles live tracker data with the user's instructions and then drives motors on the gimbal to adjust the camera assembly's orientation to achieve the user's desired framing of one or more actors. The interface for user instructions is discussed in the supplementary materials, and actor location tracking is discussed in Section 4.

The visual servoing community has made tremendous progress in constructing methods for moving cameras and robotic arms to desired positions in space and/or orienting them based on some external visual signal [34]. The bulk of visual servoing use-cases are in robot end effector control in manufacturing. Usually, these methods involve the solution of a Jacobian matrix [8, 15] that encodes tasks and joint movement constraints. While some work explores modulating the variance of the mean position of all visual points of interest in image space [20, 72], none has provided a transparent formulation for controlling per target variance nor does one provide a framework for gradual change between different tasks and constraints. We borrow themes from the visual servoing literature while constructing a task-specific control scheme.

Appealing camera positioning and orientation is essential for effective video game design. As such, the video gaming industry has generated methods and implementation tricks for employing dynamic cameras that follow in-game action on-the-fly [23]. While these methods assume that targets are known with certainty and that control over camera properties is instantaneous, we take hints from the community when designing our own control scheme and incorporate strategies to both mitigate and cope with real-world noise.

At a high level, the controller is a closed-loop feedback system with PID [32] controllers that minimize an error signal, $e(t)$, by modifying the camera frame's yaw and pitch over time. $e(t)$ is an abstraction of the error between real-time dynamic actor locations and desired user framing encapsulated in the script. If we simplify camera space conversions, ignore noise, and assume only a single tracked target, then $e(t)$ is just the screen space difference between the actor's tracker location and the user's screen space requirement for actor framing, with both x and y components. Errors in x and y are corrected by changing the camera frame's yaw ($\dot{\psi}$) and pitch ($\dot{\theta}$), respectively. The corrections are handled by PID controllers; thus,

$$\dot{\psi} = \mathrm{PID}(e_x(t)) \text{ and } \dot{\theta} = \mathrm{PID}(e_y(t)). \tag{2}$$

We tune our PID controllers using a relaxed version of the Ziegler-Nichols procedure [77] to achieve the tightest response possible while minimizing overshoot, given delay and processing constraints. Note that these are camera frame radial velocities and not direct motor torque commands. The underlying gimbal camera assembly radial velocity stabilization is tuned in the gimbal's firmware and is not discussed here.

This abstracted version of e(t) is suitable for a single actor and will produce erratic camera motion since raw tracker locations are noisy either due to tracker inaccuracy or due to subtle actor movements. This is fine if the preferred style is very erratic, unnerving camera motion with a random component due to noise, but not for any other desired style. Tuning the PID controllers to be lazy would ignore noise and allow for a lazy camera but would erode control over all actor-driven camera behavior and remove responsiveness when responsive corrections are required. Other design

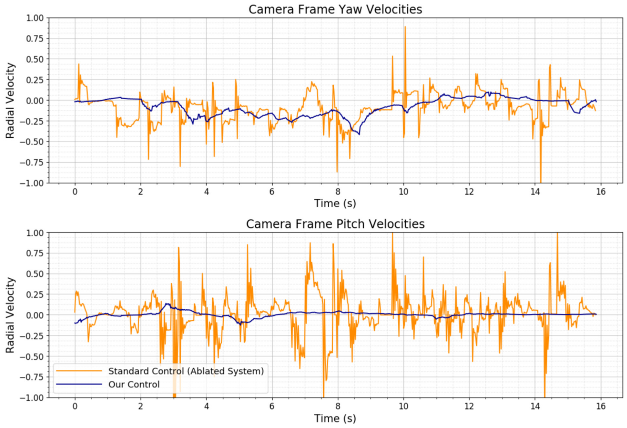


Fig. 4. Yaw (top) and pitch (bottom) camera frame radial velocities for both our full system and ablated control throughout the control ablation running scene. The standard controller (ablated system) does not ignore tracker noise and translates changes in perceived actor location from the tracker directly to an error in the PID controller. This leads to massive overcorrections and an uncontrollably erratic camera. Instead, our full controller can handle tracker noise and camera translation by using modifiable leniency (Section 5.1) on raw tracker locations and applying control weight adjustment (Section 5.2). Note that these are not gimbal motor velocities. Rather, these are target radial velocities for the camera frame to achieve. Raw gimbal axes velocities and torque are a function of required camera frame velocities and external forces acting on the gimbal assembly.

considerations include handling behavior transitions and tracker dropouts, where potential camera jerks are likely and a single one would ruin a take. The controller must handle a variety of filming scenarios and behaviors —from single actor to multi-actor, from action scenes to calmer, slower-paced scenes, and the transitions between them. Therefore, we design the controller to pursue the following design objectives.

(1) **Achieve desired user framing:** On every loop, the system should minimize the difference in required actor framing versus actual actor framing. Logical compromises should occur when framing multiple actors at once.
(2) **Move the camera only if motivated:** The user can provide an ellipse for each actor in each behavior, indicating an area around the actor. In this area, their movements do not result in camera frame reorientation. The controller should also ignore noise from raw tracker estimates so that the camera does not oscillate and produce unpleasant motion. This is discussed in Section 5.1.
(3) **Enable smooth transitions:** As behaviors change, different actors come in and out of scene. The transitions between different actors must be smooth. This is discussed in Section 5.2.

The $e(t)$ signal driving the PID controller is computed based on these objectives. Specifically, $e(t)$ comes from a weighted Procrustes module, which we simplified: it aligns the current 2D actor location(s) with the location(s) required by the user, subject to the available degrees of freedom. We found that in-plane rotation for framing was not helpful. Therefore, in all of our experiments, we used a Procrustes model that simply finds the translation vector $\mathbf{T^c}$ as the weighted difference of the average actual locations and the average required location; $\mathbf{T^c}$ acts as our error vector $e(t)$. We also extend this to account for star camera zoom by scaling the input points appropriately. To address Objective (2), we add "Leniency," in which instead of passing raw tracker components to compute $\mathbf{T^c}$, we instead produce dynamically decoupled and filtered Augmented locations in Section 5.1. To address Objective (3), we modify the influence each actor has in current framing given transitions and tracker confidence and introduce filtering on required script behavior in Section 5.2. Figure 3 provides a high level overview of the control system's components.

5.1 User-Defined Motivated Camera Movement

Minimizing the difference between actor locations given by the tracker, $\mathbf{P^T} = \{\mathbf{p^T_1}, \ldots, \mathbf{p^T_n}\}$, and required actor locations, $\mathbf{P^R} = \{\mathbf{p^R_1}, \ldots, \mathbf{p^R_n}\}$, using Procrustes satisfies Objective (1). However, this would not filter noise, either from the tracker or abrupt camera translation, and would not allow for selectively ignoring small actor movement. Instead, to satisfy Objective (2), we define *tracked-smoothed-augmented* points ("Augmented") $\mathbf{P^A}$ that are smoothed versions of $\mathbf{P^T}$ and use those to compute $\mathbf{T^c}$. The obvious means of making augmented versions $\mathbf{P^A}$ is via Kalman filtering. Thus,

$$\mathbf{p^A_i} = \text{KalmanFilter}\left(\mathbf{p^T_i}, \mathbf{h_i}\right), \tag{3}$$

where $\mathbf{h_i}$ is a process variance. A high $\mathbf{h_i}$ means that an augmented point follows its tracked point quickly, allowing for an immediate change in the error term for that actor and an immediate correction signal from Procrustes resulting in a very responsive camera-to-actor movement. A small $\mathbf{h_i}$ allows for the opposite: each $\mathbf{p^A_i}$ lazily follows its track point $\mathbf{p^T_i}$, resulting in a less immediate corrective signal and less eager camera panning.

However, fixing $\mathbf{h_i}$ limits user control. Ideally, there should be definable areas of forgiveness around an actor where small movements are ignored. Setting a small $\mathbf{h_i}$ allows this, but this would ignore actor movements outside of this area when they do matter. Instead, we modulate each $\mathbf{h_i}$ based on $\mathbf{d^{LE}_i}$, the current discrepancy between a tracked point $\mathbf{p^T}$ and its augmented point $\mathbf{p^A}$ from the previous timestep. We make $\mathbf{h_i}$ proportional to $\mathbf{d^{LE}_i}$ so that with a small $\mathbf{h_i}$, the Kalman filter will ignore new updates given by $\mathbf{p^T_i}$ and instead choose to maintain the older location of $\mathbf{p^A_i}$. As a point $\mathbf{p^T_i}$ moves too far from its $\mathbf{p^A_i}$, the distance, $\mathbf{d^{LE}_i}$, ramps up $\mathbf{h_i}$ and the Kalman filter is more sensitive to new incoming updates via $\mathbf{p^T_i}$. Thus, $\mathbf{p^A_i}$ follows $\mathbf{p^T_i}$ more closely.

The relationship between $\mathbf{d^{LE}_i}$ and $\mathbf{h_i}$ is user definable and based on a family of exponential functions. Here, we define a set of aesthetic parameters for each axis.

- Zero Error Lift, v: This forces a non-zero value when $\mathbf{d^{LE}_i}$ is at zero. The result of a high v is an immediate responsive pan from the camera when the actor moves small distances from rest.
- Agnostic Gap, a: This defines how much distance the actor has to travel before the camera pans.
- Curve Profile, q: This defines the ramp up at the edge of the allowed area of leniency and determines how sharply the camera will pan when an actor begins to leave that leniency area.

We also set a hard limit on $\mathbf{h}$ via η. This cap limits the impact from temporal instabilities in the tracker and was experimentally set to 0.01 for vertical motion and 0.05 on horizontal motion in all experiments. We include a qualitative experiment for demonstrating these smoothing functions and raw tracker values in the supplemental videos. For each actor, each component of $\mathbf{h} = (h_x, h_y)$ is computed as:

$$h_x = \eta_x \operatorname{clamp}(0, 1, e^{q_x(d_x^{LE} - a_x)} + v_x) \text{ and}$$

$$h_y = \eta_y \operatorname{clamp}(0, 1, e^{q_y(d_y^{LE} - a_y)} + v_y).$$

These equations are not obvious, but the three input parameters have interpretable connections to the radii (r_x, r_y) of each ellipse drawn by the user in the GUI. The functions relating radii $\mathbf{r}$ to each of these parameters are given in the Supplementary Material. In brief and for a single axis, for a large r, v is reduced such that almost no movement occurs at zero error, a is made to satisfy the distance defined by r, and q is set so that the transition is smooth. Conversely, for a small r, v is kept high for immediate reaction, a is set so that the point at which the curve increases happens earlier, and q is set so that the curve is sharp. See Figure 6 for different curves corresponding to different user input radii. We include an example of multiple-actor leniency in the supplemental video.

Note that the naïve solution of simply weighting the error associated with the i-th actor to zero when the actor is in some allowed radius will not achieve multi-actor leniency. Most situations result in a suboptimal optimization in which required actor locations are not fulfilled perfectly due to physical limitations. A zero weight for an actor would result in a new optimization and, counterintuitively, produce camera motion when none was required. See Figure 5 for an illustration of this.

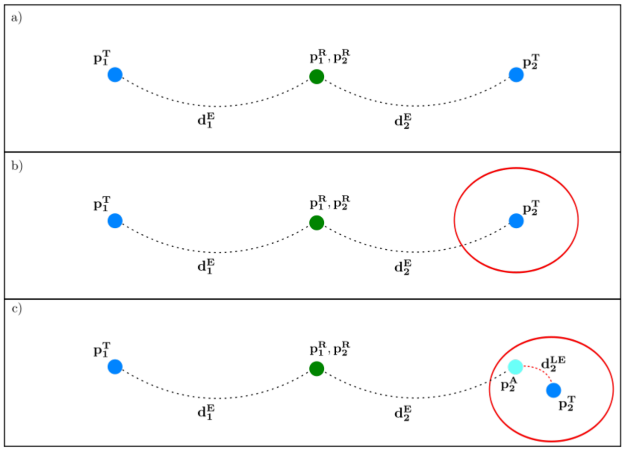


Fig. 5. (a), (b), and (c) show an example in which the user specifies that both actors should be framed in the center given by required locations $\mathbf{p}_1^R$ and $\mathbf{p}_2^R$. However, the actors' relative locations at $\mathbf{p}_1^T$ and $\mathbf{p}_2^T$ make it impossible for that requirement to be fulfilled. As such, the best framing possible at steady state is where both are equidistant from the center. In (a), no leniency is defined; thus, a movement by either $\mathbf{p}_1^T$ or $\mathbf{p}_2^T$ will need a new optimization and the camera pans. In (b) and (c), leniency is required on actor 2, given by the red ellipse defined by the user — that is, if the actor at $\mathbf{p}_2^T$ moves within the ellipse, the camera should not respond. In (b), a naïve solution to achieve leniency is to attenuate the error term $\mathbf{d}_2^E$ when the target $\mathbf{p}_2^T$ is close to the point of the optimization at steady state (where $\mathbf{p}_2^T$ sits). However, since this is a less than ideal framing with both required points at the center, a new optimization will be found that improves $\mathbf{d}_1^E$ and the camera pans, disregarding leniency. Instead, in (c), we formulate a new augmented point $\mathbf{p}_2^A$ that is output from a Kalman filter on $\mathbf{p}_2^T$ whose process variance is modulated by the distance $\mathbf{d}_2^{LE}$ with regard to the ellipse. The actor and ellipse at $\mathbf{p}_2^T$ can move around the augmented point, $\mathbf{p}_2^A$, and as long as the augmented point is in the ellipse, the process variance $\mathbf{h}_2$ remains low and the augmented point at $\mathbf{p}_2^A$ does not move with $\mathbf{p}_2^T$. The error term $\mathbf{d}_2^E$ remains low since $\mathbf{p}_2^A$ remains stationary although the actor at $\mathbf{p}_2^T$ has moved; thus, the camera does not pan to compensate. When the actor leaves this ellipse, $\mathbf{h}_2$ is ramped up, $\mathbf{p}_2^A$ moves to follow $\mathbf{p}_2^T$, the error term $\mathbf{d}_2^E$ changes, a new optimized framing is found, and the camera pans.

5.2 Actor Transitions and Path Behavior

To allow for smooth transitions between subjects, satisfying Objective (3), each actor is assigned a weight, w_i, that modifies the actor's error term in the Procrustes optimization. When actors appear in frame and are part of the current behavior, their weight is increased progressively. Their weight is decreased when they either disappear from frame, because of occlusion or tracking failure, or are no longer included in the behavior.

For each actor, we also apply Kalman filters on user-selected required points as they transition between different behaviors so that no discontinuities occur. Separately, a behavior can be intentionally shaky to give the viewer a hand-held impression. To achieve shakiness or intentional banking behavior (like an airplane changing course), the controller reads the gimbal IMU accelerations on the camera's horizontal axis, applies smoothing, and actuates the roll axis. See the supplemental video for an example of a path behavior.

5.3 Focal Length Control

We make star camera focal length control available to the user in a few ways. Standard operation modifies the set of required points, $\mathbf{P^R}$, to match star camera framing at the current zoom level. We do this by applying a scaling matrix with knowledge of the camera intrinsics at each zoom interval. The new required points used as input to the optimization are now

$$\mathbf{P^R}' = S_z \mathbf{P^R}.$$

This keeps star camera framing consistent with what the user has defined on the main framing panel regardless of zoom level.

The simplest form is where the user can specify a zoom level for a script and use required points, $\mathbf{P^R}$, placed in guide camera image space as is. This gives the user creative freedom over actor framing at the edge and beyond of the star camera's frame. In this case, the points $\mathbf{P^R}$ are unchanged. Other settings allow for the zoom to automatically change depending on the size of the actor in frame.

6 RESULTS AND EVALUATION

The LookOut system has been used to film over 12 hours of footage. To measure its strengths and find its weaknesses, we split up validation into five components:

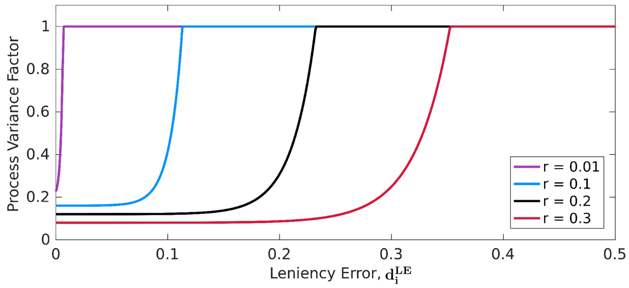


Fig. 6. Curve profiles for different user-prescribed leniency radii. These radii represent areas around the actor where camera panning is attenuated if the actor moves in that area. The y-axis is applied to η to produce each Kalman filter's process variance, $\mathbf{h}$. The x-axis is the difference, $\mathbf{d}_i^{LE}$, between the augmented version of the actor's location from the previous timestep, $\mathbf{p}_i^A$, and the raw tracker location, $\mathbf{p}_i^T$, and is normalized relative to screen space size. A smaller ellipse radius limits the area where the actor can move without a camera pan, as the process variance ramps up immediately. A larger ellipse allows for more actor movement before the camera starts panning.

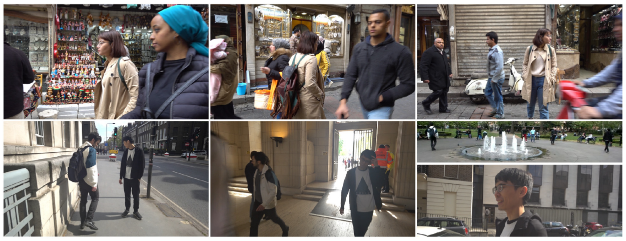


Fig. 7. Sample frames from annotated videos used for benchmarks. Top: Market, a 3 min 20 s scene of the actor in the beige coat walking through a crowded market. There are many occlusions in this scene, including where the target appears in frame with a different appearance than when the target went into occlusion. Bottom: TwoPeople, a 10 min 30 s scene of two actors on a walk through a campus and a park. Both actors wear similar-looking clothes, occlude one another, disappear from the frame entirely, are seen at different scales, and walk at various distances away from the camera.

(1) Tracker performance
(2) Controller Evaluation
(3) Hands-on evaluation by filmmakers
(4) Discussion of LookOut footage with senior filmmakers
(5) Qualitative showcase of LookOut in different scenarios

For Validation Components (1) and (2), we also compare performance against the DJI Osmo Mobile 3 in the supplemental material. Note that illustrated footage in the supplemental website is slowed down to make ingesting telemetry data easier.

6.1 Tracker

We test our tracker's performance on the VOT Long-Term Challenge [38] and on two long manually annotated videos that better represent our film-production use case. Market (one-actor scene at 3 min 20 s with annotations every frame) and TwoPeople (two-actor scene at 10 min 30 s with annotations every 5 frames) are challenging scenes with representative clutter, many occlusions by distractors, variable appearance before and after occlusion, and lighting changes (Figure 7). Crucially, the subjects' appearance changes to something not seen before when emerging after an occlusion. While our tracker and others can sometimes be shown the subject from all angles to build a representative history, this test also checks for pickup-and-go filming performance. Thus, no such five-second grace training period is given. We ultimately advocate our tracker for the tracking of people in our use case. However, we include all videos from the VOT challenge in the comparison.

We describe in detail how these videos are annotated and the exact details of the associated metrics in the supplementary material. Broadly, a tracker is awarded a true-positive (TP) point for a frame if it either correctly predicts the bounding box of the actor or correctly predicts that the actor is occluded. If a tracker outputs an incorrect bounding box, regardless of whether or not the actor is occluded, it is given a false-positive point (FP) for that frame. If a tracker does not output a bounding box when the actor is not occluded, it is given a missed-track (MT) point. We distinguish between FP and MT in this way to highlight errors that would point the camera away from the targets of interest, as is expressed with FP. We also compute the pixel distance between the center of the ground truth box and the center of the track, D, and obtain a mean over all updates, $\overline{D}$. We report raw unnormalized results for Market and TwoPeople (average of both actors) and normalized results on VOT-LT2019 [38] sequences in Table 1.

We also ran a qualitative experiment with the leading VOT2018 real-time tracker, DaSiamRPN [76]. We filmed an actor walking in a pedestrian area using both our tracker and DaSiamRPN [76] in separate takes. The rest of LookOut is kept constant, including actor weighting and actor-specific leniency, which both help to mitigate tracker noise and errors (but do not affect tracking). We run two takes each and show all takes in the supplementary video. While DaSiamRPN fails to track the actor in both takes, our tracker does. These takes show the importance of our robustness to imposters in filming.

6.2 Controller Evaluation

In order to evaluate the controller components responsible for translating script commands to target camera frame radial velocities, we film multiple qualitative videos and also run an ablated version of the system.

We film two takes of the same running scene at the same location and with the same predetermined path, making sure to keep the relative motion between the camera and actor consistent. One take was filmed using our full system, including a minor leniency that is close to the minimum allowed. The second take was filmed with an ablated version of the control system, or Standard Control. The ablated version of the system passes raw tracker values as is to the PID controller without actor control weight adjustment (see Section 5.2) and the leniency mechanism (see Section 5.1). Figure 4 shows camera frame radial velocities for both modes throughout this scene. Overall, the full controller satisfies scripted actor framing and largely ignores both actor track noise and camera translational motion that manifests itself as screen space motion. Following these internal and external noise sources would

Table 1. Evaluation of Our Tracker and Other Leading State-of-the-Art Real-time Trackers on the VOT Long-term Tracking Dataset

	Market, 3 min 20 s, one actor					TwoPeople, 10 min 30 s, two actors					VOT-LT2019 [38], ~2 min 24 s, one target				
	$TP \uparrow$	$MT \downarrow$	$FP \downarrow$	$\overline{D} \downarrow$	T (ms)↓	$TP \uparrow$	$MT \downarrow$	$FP \downarrow$	$\overline{D} \downarrow$	T (ms)↓	$TP \uparrow$	$MT \downarrow$	$FP \downarrow$	$\overline{D} \downarrow$	T (ms)↓
Our Tracker	**4765**	769	**71**	**17.5**	18.5	**2655**	562	**132**	**34.8**	20.0	0.200	0.764	**0.036**	65.0	11.7
MOTDT [43]	4468	777	362	25.5	31.6	1770	1258	320	42.1	32.3	0.229	0.690	0.081	61.3	23.0
SiamMask [66]	4213	78	1316	56.1	14.9	1783	72	1493	91.7	30.1	**0.554**	0.153	0.292	85.5	12.8
DaSiamRPN [76]	2259	1732	1616	68.0	**7.9**	1709	528	1110	87.7	**14.2**	0.494	0.275	0.230	80.9	5.9
DeepSORT [67]	3961	554	1092	57.5	21.1	765	439	2143	147.0	22.6	0.186	0.752	0.061	68.8	13.2
KCF [24]	623	3552	1432	94.2	87.3	377	2819	151	123.3	73.7	0.173	0.736	0.090	**47.2**	4.4
TLD [30]	18	**56**	5533	239.4	32.2	670	**1**	2676	146.1	57.5	0.152	**0.010**	0.837	159.2	37.2
SiamDW_LT [74]	4751	589	267	28.9	412.6	2693	175	479	42.7	1123.7	-	-	-	-	-

Other algorithms outperform ours on VOT. However, the VOT videos are qualitatively different in appearance from our use cases. So we introduce two further test sequences with 12,300 manually labeled annotations. These videos are more representative because of their cinematic style, both long- and short-term occlusions, and the presence of distractors, including people in cluttered environments. A high TP (true positive) is obviously advantageous. A low FP discourages the camera from moving onto a distractor. Some missed tracks, MTs, are tolerable, but especially after a long occlusion, missing the target could lead to catastrophic target loss. While a low MT score is important, a trivial tracker that always outputs a bounding box, whether or not the target is occluded, would allow the tracker to be distracted. In the short term, this will lead to FPs, and in the long term, it will pollute that actor's appearance representation. FPs are especially detrimental for LookOut, because the camera is controlled by tracker output. An inaccurate position will move the camera away, further decreasing the chances of recovery and ruining a take. All runtimes include detector latency when appropriate. Detection-based trackers are all run on tiny-YOLOv3 output. All trackers are run in a single thread, including ours.

Table 2. Ablation Study of Our Tracker on the Two Test Sequences and the Metrics We Establish in Section 6.1

	$TP \uparrow$	$MT \downarrow$	$FP \downarrow$	$\overline{D} \downarrow$	T (ms)↓
Our Tracker	**0.822**	0.152	**0.026**	**22.6**	19.3
No Recovery	0.745	**0.088**	0.166	35.8	**18.7**
Faulty Encodings	0.785	0.140	0.075	26.1	19.0
Greedy Encodings	0.698	0.234	0.068	41.2	19.0
Simple History	0.688	0.182	0.131	50.7	19.0

Simple history is a flavor of our tracker but with no feature history management, only the last seen L encodings are stored in memory. *No recovery* is our tracker but without a recovery stage. If a detection matches a target once, it is accepted as the target, leading to stray incorrect tracks on distractors, a high FP score, and a lower TP score in the long term. *Greedy Encodings* stores a new incoming encoding into the feature gallery even if similar ones exist, filling up the gallery faster, thus leading to a restrictive appearance memory. *Faulty Encodings* accepts detection encodings that are overlapped with other detections in the scene. This pollutes the gallery with noisy encodings and detracts from the tracker's ability to avoid distractors. Since the gallery sampling strategy is random, all trackers are run 40 times to ensure fairness.

lead to an uncontrollably erratic camera as shown in the video and side-by-side radial velocities in Figure 4. Please see this side-by-side comparison in the supplemental validation video and in the website as the video pair named *Fully Ablated Control* under *Control Ablation* for both the illustrated visualization and the star footage of this targeted A/B ablation comparison.

In the video *Ablated Multi-Actor Weighting* under *Control Ablation*, we show how using binary weights for actor script transitions produces a nervous erratic camera at best and usually leads to a broken take. This happens because the error T_c goes from being entirely Actor2 focused to entirely Actor1 focused, and vice versa, in one timestep. This spikes the PID controllers, leading to erratic corrections and a nervous camera. All other videos with multiple actors will show behavior with the method outlined in Section 5.2 and with leniency from Section 5.1.

We also film scenes to show the effect of variable leniency on a single actor and for multiple actors — *Hampstead Leniency Switch* and *Clown and Calm* — in the supplemental website. These videos demonstrate LookOut's ability to frame targets according to user-defined leniency. In the supplemental website, see other video illustrations of actor control weights, leniency ellipsis, and actor process variances displayed when available in filming metrics.

6.3 Hands-on and End-to-End Evaluation

We designed and ran a small field study on an intermediate prototype, composed of two parts. Part 1 consisted of participants building a script using the LookOut GUI, while Part 2 involved the same participants filming the scene they have programmed.

Participants: In total, we had 5 participants: 4 participants completed both parts, while 1 participant only completed Part 1. We recruited the five volunteers (two female) by posting an advertisement on an amateur filmmakers' group and through our own social networks. Three of them work within the film and entertainment industry (one lighting technician, one backstage support person, and one director), while two are university students.

All participants had prior experience with filming, from beginner to amateur. Filming experience ranged from filming static scenes to action shots using Steadicams, from short clips for the Web to short movies. None of the participants was familiar with computer vision, nor had they been exposed to the system before the study. One of the participants reported being familiar with Blockly from toys such as the Sphero$^{\text{TM}}$, which she previously encountered in her part-time work.

Experimental Design: The study was designed to expose participants to the full operation of the system, from the creation of the configuration scripts using the GUI to the actual filming of the action. To harmonize task complexity across participants, we asked them to film a predefined sequence, communicated to them through a storyboard (printed in color on a single A3 page). Note that this form of study does not check creativity in run-and-gun scenarios. Rather, it checks productivity [57] when a DP is working solo.

Designing a suitable storyboard required careful consideration to balance conflicting requirements. On the one hand, we wanted a setting that really challenges visual tracking algorithms and the storyboard to be particularly complex for a single operator to film in one shot. These requirements were to assess the system's ability to deal with challenging filming situations and the ability of the GUI to expose a spectrum of behaviors.

On the other hand, the storyboard design was constrained by concerns around the health and safety of participants (and to satisfy our research ethics review requirements). These concerns made us rule out any sequences involving stairs, streets with vehicles, or any other scenes that could be deemed unsafe. We also limited the number of actors required to two and the overall study duration for each participant to one hour.

After a number of iterations involving consultation with a separate filmmaker, we agreed on the storyboard. Like many long takes, it incorporates a variety of shots, some of which would be quite hard to implement with standard filming techniques. One such difficult shot implements a sudden camera transition between the two actors, followed by the participant having to run to keep up with the actor named "Blue."

Another hard shot is the swooping pan, in which the camera starts low and ends up high as the participant moves around the tree until the individual is behind actor "Red." This would normally be hard to execute, as it involves the camera operator moving from a crouched to a standing position while ensuring that the actor is kept within frame. With LookOut, the camera angle is adjusted automatically to frame the actor, letting the camera operator focus on one's own movement. The storyboard can be found in the supplemental material.

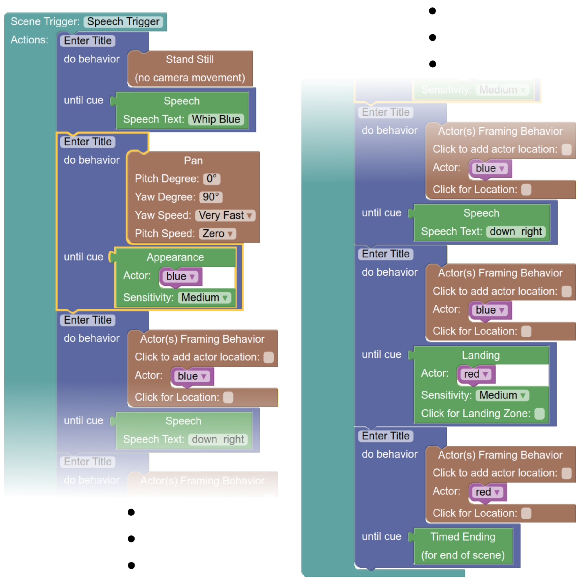
As confirmation that the story and park setting were challenging themselves, two of our participants commented that, if they had the option, they would split the scene into separate shots ("I would segment the scene into different shots" and "normally I would split the scene into several parts").

Procedure: Participants were given verbal instructions providing a brief overview of the user interface and the scene they were required to film. The setting was a local park, in late afternoon through dusk. Participants were handed a copy of the storyboard and left on a bench to create the required configuration scripts on a laptop running the GUI. Figure 8 shows an example of a script created by a participant.

Once participants declared that they were satisfied with the scripts, they were provided with a quick overview of how the rest of LookOut works and invited to start filming. As they tested their scripts, they were allowed to go back to the GUI and change aspects they thought did not work very well. For example, one participant went back and changed the speed of transitions, having realized that the "very fast" setting might miss locating the actor entirely.

Within 50 minutes of the start of the study, or as soon as participants filmed a scene they were satisfied with, the filming ended, and participants were asked to take part in a short interview (10 minutes) about their experience.

Configuration Scripts and GUI: All five participants who attempted Part 1 of the study were able to successfully use the GUI to create configuration scripts to match the storyboard. This process lasted between 20 to 25 minutes, and was carried out independently by participants, although they were allowed to ask the experimenters questions. Participants were generally pleased with the UI's functionality. One participant commented that "programming the framing was like coding, so it was simple enough" while another participant stated that he was happy with the UI's behavior possibilities: "already a lot with actor recognition and speech recognition." However, some participants did mention the need for a "zoom function or focal length change." In addition, one participant wanted a feature to track objects: "e.g. if you wanted to track a statue while walking around it." Although LookOut supports object tracking, the UI did not offer this possibility at the time, letting them select only actors.



```
Scene Trigger: Speech Trigger
Actions:
  Enter Title
  do behavior   Stand Still
                (no camera movement)
  until cue     Speech
                Speech Text: Whip Blue
  Enter Title
  do behavior   Pan
                Pitch Degree: 0°
                Yaw Degree: 90°
                Yaw Speed: Very Fast
                Pitch Speed: Zero
  until cue     Appearance
                Actor: blue
                Sensitivity: Medium
  Enter Title
  do behavior   Actor(s) Framing Behavior
                Click to add actor location:
                Actor: blue
                Click for Location:
  until cue     Speech
                Speech Text: down right
  Enter Title
  do behavior   Actor(s) Framing Behavior
  •
  •
  •

  •
  •
  •
                Sensitivity: Medium
  Enter Title
  do behavior   Actor(s) Framing Behavior
                Click to add actor location:
                Actor: blue
                Click for Location:
  until cue     Speech
                Speech Text: down right
  Enter Title
  do behavior   Actor(s) Framing Behavior
                Click to add actor location:
                Actor: blue
                Click for Location:
  until cue     Landing
                Actor: red
                Sensitivity: Medium
                Click for Landing Zone:
  Enter Title
  do behavior   Actor(s) Framing Behavior
                Click to add actor location:
                Actor: red
                Click for Location:
  until cue     Timed Ending
                (for end of scene)
```

Fig. 8. An example of a configuration script programmed by one of the participants. The participant opted to use whip pans and actor-based cues to automate most of the camera's behavior change.

In some cases, after one or more attempts at filming the scene, participants realized that they were not happy with some of the details in their configuration scripts. In these cases, participants edited the configuration scripts using the GUI. In one case, a participant realized that the duration for a timed cue was too short; thus, the participant adjusted the value. In another case, the participant was not happy with the angle of the yaw in a pan; thus, the participant increased it. The adjustments took less than 5 minutes, as the performed changes were minor parameter settings. No issues were reported or observed with the interface.

These findings confirm that the task of scripting the behavior of the LookOut controller can be completed with minimal training by novice users. The editing of the parameters after a script was tested indicates that participants were able to relate the two and could refine the script behavior to match their needs.

Filming and Resulting Footage: In the remaining 15 to 25 minutes, two of four participants had enough time to record a long take that they were happy with for this scene. In the other two cases, there were issues with the tracking of actors that led to the scene not being adequately filmed within the prescribed time frame. This was caused by the lighting being uncharacteristically bad: on most film sets, there would be procedures in place to reduce the effect of strong sunlight filtering through the trees, to keep the actors consistently lit.

One participant pointed out that even though they did not have a viewfinder during filming, she could tell from the physical movement of the camera that it was smooth: “from the physical movement of camera it looked smooth.”

Participants also spoke about the convenience of having automatic movement of the camera, as it meant that they could focus on other aspects of the filming, such as keeping up with the actors. One participant described the task of keeping the camera focused on an actor as “you can just track someone without caring about it.”

Participant Comments: The aim of the storyboard was to have several different types of shots, some of them that would be harder to execute with traditional filming equipment. One of these shots involved having the camera quickly panning between the two actors: “the whip pan was easier with the AI, it found and tracked the subject automatically. Otherwise I would have to rehearse that 3–4 times to get it correctly.” By using LookOut, the participant was able to correctly capture the shot from the first take.

Participants were particularly pleased with using voice as a trigger for the next action in the scene: “voice activating the cues worked very well.” One participant stated that they “could see directors using that to program in actor’s lines.” This feature simplified the filming process for participants, with all participants who attempted Part 2 using speech triggers within their scripts.

When asked if there were other camera behaviors they’d like to see in LookOut, one participant mentioned tracking other objects, which we experimented with using a generic class object tracker (see “Other Trackers” on the supplemental website). Three participants mentioned zoom; although our current hardware limits optical zoom, we have made use of sensor-based zoom (see “Zoom”). The fifth participant said that the existing behaviors were already a good toolbox and specifically pointed out voice triggers as useful building blocks.

Shot Breakdown: Takes were ruined either due to faulty tracking in bad lighting on the early version of the tracker used (46%), participants forgetting to fire triggers they placed (12%), voice recognition failure (8%), and another 26% due to miscellaneous issues (bystanders getting in the shot, actor mistakes, batteries running out, etc.). The rest (12%) yielded usable takes. The bulk of ruined takes come from tracker error, which motivated the development of our final proposed tracker and the underlying principles we lay out in Section 4. We have used the latest version of the proposed tracker for filming visually challenging scenes, including those in equally harsh lighting —“Zoom Run” and “IRL Tracker Comparison.”

6.4 Critique by Senior Filmmakers

We sought out three senior filmmakers, separate from the filmmakers who influenced the design of LookOut and separate from those who did the Hands-On Evaluation (Section 6.3). Each of them has been working as a professional DP for 9, 13, and 25 years, respectively. Each has a mix of experience, in both scripted scenes with crew and actors and run-and-gun filming for documentaries or journalism. We interviewed them separately, each time showing the same three unedited video examples shot using the LookOut system (see Figure 9). We asked the same predefined set of questions to prompt them to think aloud while watching the videos.

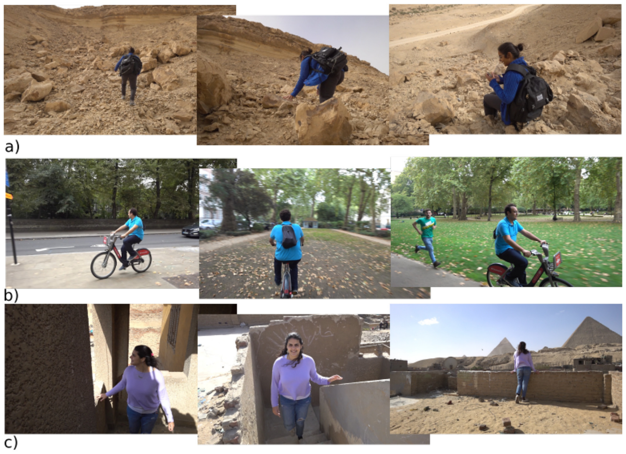


Fig. 9. Videos shown to senior filmmakers. (a) Rocky escarpment — camera operator climbing on foot and with one hand free. (b) Bike ride — camera operator also riding a bike. (c) Pyramids — camera operator walking backwards on stairs.

The questions are listed in the supplementary material but can be broadly grouped as concerning (i) the equipment and people needed to film these long takes normally (without LookOut) and (ii) critiques of both the footage and current LookOut capabilities.

First, to shoot such takes without LookOut, two of the filmmakers have used drones and would consider using them here if a licensed pilot were available and the noise was not prohibitive. Two of them said they would use cranes for video-A if the budget allows. One complained, however, that multiple cranes have bad placement of viewfinders, resulting in them shooting blindly for long periods. For video-B and video-C, one said that he would use a Steadicam and the other two had specific two- or one-handed gimbals (like that modified for LookOut) that they would try again despite having small and awkward viewfinders.

They each would need a second person at minimum, and usually more, to help with typical stabilization-only filming. Independently, they all said that if only one extra helper were available, then that person would be the spotter for the operator. A spotter physically guides the operator around obstacles.

Second, their views of the footage and the LookOut system were very positive, with some caveats. The two more senior DPs expressed the sentiment that LookOut would have no place in a big-budget project because the Director and DP can give orders verbally that get carried out eventually. Also, those two would need to use LookOut multiple times before they would trust its reliability and, ideally, prefer that colleagues make some films with it first. Transcribed interview quotes are in the supplemental material, which include comments such as “That would be so helpful! Especially in those run-and-gun situations, documentary, travel, journalism. If you’re filming something that won’t happen again, you can focus on the other things” and “I could be more creative once I got used to it.”

6.5 Qualitative LookOut Results

LookOut has been used by the authors, by test subjects, and by novices who usually (but not exclusively) filmed using existing behavior scripts. A representative cross-section is shown in the supplemental videos webpage. Some noteworthy examples include sports in which the operator is participating, such as skateboarding or using one hand while playing frisbee, scrambling, or cycling, for example. For the Gnome and Plumbing-shop sequences, we filmed, as an exception, using the DaSiamRPN [76] tracker within LookOut, to cope with unusual object categories, though this required multiple takes. In contrast, the vast majority of takes using our tracker worked out on the first try.

7 LIMITATIONS AND DISCUSSION

The LookOut premise, software, and hardware each have limitations. While it would be informative to do the end-to-end evaluation under run-and-gun conditions, which represent the vast majority of users, those situations are rarely repeatable and are considered dangerous from an ethical experimentation perspective. That led us to use simple scripted scenarios for that evaluation. The senior filmmakers are likely right that big-budget productions will be reluctant to use LookOut. The field study tested with participants from our low-budget demographic of filmmakers with a fixed storyboard, but an ideal comprehensive user study would focus on adventure-athletes and journalists in somewhat dangerous conditions to check real run-and-gun scenarios.

The LookOut GUI worked better and more intuitively than expected. The detector and tracker combination, too, performs admirably across highly diverse scenarios, though they are designed initially for tracking actors across occlusions in handheld films and are unremarkable on the standard Computer Vision benchmarks MOT [48] and VOT [38]. The single weakest component across the LookOut system is the detector. We have seen it confuse the tracker when the actor hides or gets too small, there is too much motion blur, or actors wear the same uniform. For now, better detectors are available but not with the low-latency required by the controller. LookOut is built in Python, which is not optimized for real-time and multiple threads. We chose this for easier comparison with other trackers and rapid prototyping; thus, efficiency gains are possible. Like other appearance encodings, ours is sometimes susceptible to harsh and variable lighting (Figure 10), which makes the system most vulnerable at dusk or dawn, and possibly when switching between indoors and outdoors. On-the-fly camera image processing optimized to improve vision task performance similar to that in [61] may help.

There are potentially two improvements for the hardware. First, some users requested that LookOut also manage focus-pulling and zooming. This would require a star camera for which both focus and focal length is software controllable in real time. While we have showcased a hardware-limited version of zooming with our star camera, further real-time control of both focus and focal length is required to better realize this improvement. We have not found a suitable model yet. Further, we use a guide camera with a limited field of view; 360° cameras are rarely used for cinematic filming due to limited resolution but could function as guide cameras. Then, new behaviors could better "anticipate" actors that are not in frame for the star camera yet. Extra sensing capability on the guide camera either through depth or infrared would further improve tracking and cinematic control. We will release the LookOut blueprints and downloadable system.

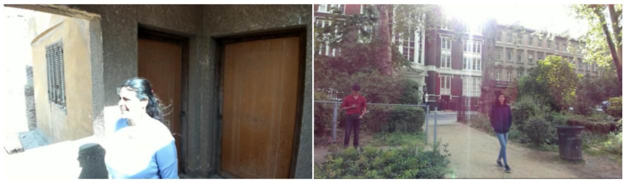


Fig. 10. Harsh light and lens flares can upset the detector and lead to gaps in tracking. If such a lighting change is fast enough and then long lasting, the tracker may not adequately associate new encodings with known actors, leading to a loss of tracking.

ACKNOWLEDGMENTS

We're grateful for thoughtful input of established filmmakers and videographers, including Luke Palmer, Gary Sinyor, Dave Ward, Tim Gardner, Tom Forsey, Jon Challicom, and Matt Judd. Thanks to all the user study participants and to our talented actors, including dancers Mark Turner and Diana Liasuk, and skateboarders Oliver Fine and Michael Jango'an. Through many development iterations, we are grateful for filming on-the-go, by Eric, Dan, Philipp, Pradyumna, Yushiang, David, Alex, Robert, Arjun, Emily, Ahmed, Habiba, Shahd, Asaad, Elbetity, Yomna, and Hala. Thanks to Peter Hedman and Graham Thomas for valuable discussions.

REFERENCES

[1] Amirsaman Ashtari, Stefan Stevšić, Tobias Nägeli, Jean-Charles Bazin, and Otmar Hilliges. 2020. Capturing subjective first-person view shots with drones for automated cinematography. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 39, 5, Article 159 (Aug. 2020), 14 pages. https://doi.org/10.1145/3378673

[2] John G. Avildsen, Irwin Winkler, and Robert Chartoff. 1976. *Rocky*. United Artists.

[3] Luca Bertinetto, Jack Valmadre, Joao F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. 2016. Fully-convolutional Siamese networks for object tracking. In *European Conference on Computer Vision*. Springer, 850–865.

[4] B. Brown. 2016. *Cinematography: Theory and Practice: Image Making for Cinematographers and Directors*. Taylor & Francis.

[5] Garrett Brown. 2018. Garrett Brown. Retrieved March 29, 2018 from http://www.garrettbrown.com/.

[6] Luka Čehovin, Aleš Leonardis, and Matej Kristan. 2016. Visual object tracking performance measures revisited. *IEEE Transactions on Image Processing* 25, 3 (2016), 1261–1274.

[7] Marc Christie, Patrick Olivier, and Jean-Marie Normand. 2008. Camera control in computer graphics. In *Computer Graphics Forum*, Vol. 27. 2197–2218.

[8] Peter I. Corke. 1994. Experiments in high-performance robotic visual servoing. In *Experimental Robotics III*, Tsuneo Yoshikawa and Fumio Miyazaki (Eds.). Springer, Berlin, 193–205.

[9] Kostas Daniilidis, Christian Krauss, Michael Hansen, and Gerald Sommer. 1998. Real-time tracking of moving objects with an active camera. *Real-Time Imaging* 4, 1 (1998), 3–20.

[10] M. Davis. 2003. Editing out video editing. *IEEE MultiMedia* 10, 02 (Apr 2003), 54–64. https://doi.org/10.1109/MMUL.2003.1195161

[11] Thang Ba Dinh, Nam Vo, and Gérard Medioni. 2011. High resolution face sequences from a PTZ network camera. In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG'11)*. IEEE, 531–538.

[12] DJI. 2018. OSMO MOBILE 2-Share Your Story. Retrieved March 29, 2018 from https://www.dji.com/osmo-mobile-2.

[13] DJI. 2019. DJI Camera Drones. Retrieved January 26, 2020 from https://www.dji.com/uk/camera-drones.

[14] BaseCam Electronics. 2018. BaseCam Electronics. Retrieved March 29, 2018 from https://www.basecamelectronics.com/.

[15] B. Espiau, F. Chaumette, and P. Rives. 1992. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation* 8, 3 (1992), 313–326.

[16] Takuma Funahasahi, Masafumi Tominaga, Takayuki Fujiwara, and Hiroyasu Koshimizu. 2004. Hierarchical face tracking by using PTZ camera. In *Proceedings of the 6th IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE, 427–432.

[17] Vamsidhar Gaddam, Ragnar Langseth, Håkon Stensland, Pierre Gurdjos, Vincent Charvillat, Carsten Griwodz, Dag Johansen, and Pål Halvorsen. 2014. Be your own cameraman: Real-time support for zooming and panning into stored and live panoramic video. *Proceedings of the 5th ACM Multimedia Systems Conference, (MMSys'14)*, 168–171. https://doi.org/10.1145/2557642.2579370

[18] Quentin Galvane, Christophe Lino, Marc Christie, Julien Fleureau, Fabien Servant, François-Louis Tariolle, and Philippe Guillotel. 2018. Directing cinematographic drones. *ACM Transactions on Graphics* 37, 3, Article 34 (July 2018), 18 pages. https://doi.org/10.1145/3181975

[19] Vineet Gandhi, Remi Ronfard, and Michael Gleicher. 2014. Multi-clip video editing from a single viewpoint. In *Proceedings of the 11th European Conference on Visual Media Production,* London, United Kingdom, *(CVMP'14)*. ACM, New York, NY, Article 9, 10 pages. https://doi.org/10.1145/2668904.2668936

[20] N. R. Gans, G. Hu, and W. E. Dixon. 2008. Keeping objects in the field of view: An underdetermined task function approach to visual servoing. In *2008 IEEE International Symposium on Intelligent Control*. 432–437.

[21] Michael Gleicher and Andrew Witkin. 1992. Through-the-lens camera control. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'92)*. 331–340.

[22] M. Grundmann, V. Kwatra, and I. Essa. 2011. Auto-directed video stabilization with robust L1 optimal camera paths. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11)*.

[23] Mark Haigh-Hutchinson. 2009. *Real Time Cameras: A Guide for Game Designers and Developers*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[24] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. 2014. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 3 (2014), 583–596.

[25] Chong Huang, Fei Gao, Jie Pan, Zhenyu Yang, Weihao Qiu, Peng Chen, Xin Yang, Shaojie Shen, and Kwang-Ting Cheng. 2018. ACT: An autonomous drone cinematography system for action scenes. In *IEEE International Conference on Robotics and Automation (ICRA'18)*. 7039–7046.

[26] Chong Huang, Chuan-En Lin, Zhenyu Yang, Yan Kong, Peng Chen, Xin Yang, and Kwang-Ting Cheng. 2019. Learning to film from professional human motion videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19)*.

[27] Alejandro G. Iñárritu. 2014. *Birdman*. United States: Fox Searchlight Pictures.

[28] Paul Jaccard. 1912. The distribution of the flora in the alpine zone. 1. *New Phytologist* 11, 2 (1912), 37–50.

[29] Niels Joubert, Dan B. Goldman, Floraine Berthouzoz, Mike Roberts, James A. Landay, Pat Hanrahan, et al. 2016. Towards a drone cinematographer: Guiding quadrotor cameras using visual composition principles. *arXiv:1610.01691*.

[30] Z. Kalal, K. Mikolajczyk, and J. Matas. 2012. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 7 (July 2012), 1409–1422. https://doi.org/10.1109/TPAMI.2011.239

[31] S. D. Katz. 2004. *Cinematic Motion: A Workshop for Staging Scenes*. Michael Wiese Productions.

[32] M. King. 2016. *Process Control: A Practical Approach*. Wiley.

[33] Johannes Kopf, Michael F. Cohen, and Richard Szeliski. 2014. First-person hyper-lapse videos. *ACM Transactions on Graphics* 33, 4 (2014), 1–10.

[34] Danica Kragic and Henrik I. Christensen. 2002. *Survey on Visual Servoing for Manipulation*. Technical Report. Computational Vision and Active Perception Laboratory.

[35] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kamarainen, Luka Čehovin Zajc, Martin Danelljan, Alan Lukezic, Ondrej Drbohlav, Linbo He, Yushan Zhang, Song Yan, Jinyu Yang, Gustavo Fernandez, et al. 2020. *The 8th Visual Object Tracking Challenge Results (VOT'20)*.

[36] Matej Kristan, Aleš Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Čehovin Zajc, Tomas Vojir, Gustav Häger, Alan Lukežič, Abdelrahman Eldesokey, and Gustavo Fernandez. 2017. *The Visual Object Tracking Challenge Results (VOT'17)*.

[37] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pfugfelder, Luka Cehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, Gustavo Fernandez, et al. 2018. *The 6th Visual Object Tracking Challenge Results (VOT'18)*.

[38] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kamarainen, Luka Čehovin Zajc, Ondrej Drbohlav, Alan Lukezic, Amanda Berg, Abdelrahman Eldesokey, Jani Kapyla, and Gustavo Fernandez. 2019. *The 7th Visual Object Tracking Challenge Results (VOT'19)*.

[39] Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2, 1–2 (1955), 83–97.

[40] Mackenzie Leake, Abe Davis, Anh Truong, and Maneesh Agrawala. 2017. Computational video editing for dialogue-driven scenes. *ACM Transactions on Graphics* 36, 4, Article 130 (2017), 14 pages.

[41] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. 2013. Bundled camera paths for video stabilization. *ACM Transactions on Graphics* 32, 4 (2013), 1–10.

[42] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. 2016. SSD: Single shot multibox detector. In *European Conference on Computer Vision*. Springer, 21–37.

[43] Chen Long, Ai Haizhou, Zhuang Zijie, and Shang Chong. 2018. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*.

[44] Alan Lukežič, Luka Čehovin Zajc, Tomáš Vojíř, Jiří Matas, and Matej Kristan. 2018. Now you see me: Evaluating performance in long-term visual tracking. *arXiv:1804.07056*.

[45] J. V. Mascelli. 1976. *The Five C's of Cinematography: Motion Picture Filming Techniques Simplified*. Cine/Grafic publications.

[46] Yasuyuki Matsushita, Eyal Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. 2006. Full-frame video stabilization with motion inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 7 (2006), 1150–1163.

[47] Sam Mendes. 2019. *1917*. United Kingdom: Universal Pictures.

[48] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. 2016. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*.

[49] Abhinav Moudgil and Vineet Gandhi. 2018. Long-term visual object tracking benchmark. In *Asian Conference on Computer Vision*. Springer, 629–645.

[50] Tobias Nägeli, Lukas Meier, Alexander Domahidi, Javier Alonso-Mora, and Otmar Hilliges. 2017. Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics* 36, 4, Article 132 (July 2017), 10 pages. https://doi.org/10.1145/3072959.3073712

[51] Vashi Nedomansky. 2013. *Average Shot Length of 6 Famous Directors*. Accessed: 2019-08-25.

[52] Picovoice. 2019. On-device Wake Word Detection Powered By Deep Learning. Retrieved January 26, 2022 from https://github.com/picovoice/porcupine.

[53] Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An incremental improvement. *arXiv*.

[54] Remi Ronfard, Vineet Gandhi, and Laurent Boiron. 2015. The prose storyboard language: A tool for annotating and directing movies. *arXiv:1508.07593*.

[55] Noriaki Saika, Ryan Harrison, Joshua Todd Druker, Himay Rashmikant Shukla, Nenad Uzunovic, Edward Gordon Russell, and Gary Fong. 2018. Camera System Using Stabilizing Gimbal. US Patent 9,874,308.

[56] Martin Scorsese and Irwin Winkler. 1990. *Goodfellas*. United States: Warner Bros.

[57] Ben Shneiderman. 2007. Creativity support tools: Accelerating discovery and innovation. *Communications of the ACM* 50, 12 (Dec. 2007), 20–32.

[58] Skydio. 2020. *Skydio – Introducing R2*. Accessed: 2020-01-16.

[59] Steven Spielberg and Robert Watts. 1984. *Indiana Jones and the Temple of Doom*. Paramount Pictures.

[60] Y. Su and K. Grauman. 2017. Making 360° video watchable in 2D: Learning videography for click free viewing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 1368–1376. https://doi.org/10.1109/CVPR.2017.150

[61] Ethan Tseng, Felix Yu, Yuting Yang, Fahim Mannan, Karl St. Arnaud, Derek Nowrouzezahrai, Jean-Francois Lalonde, and Felix Heide. 2019. Hyperparameter optimization in black-box image processing using differentiable proxies. *ACM Transactions on Graphics* 38, 4 (7 2019). https://doi.org/10.1145/3306346.3322996

[62] Natalia E. Ursan, William E. Fenton, and Dennis K. Ho. 2004. Camera Gimbal. US Patent 6,708,943.

[63] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip H. S. Torr. 2017. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2805–2813.

[64] Jack Valmadre, Luca Bertinetto, Joao F. Henriques, Ran Tao, Andrea Vedaldi, Arnold W. M. Smeulders, Philip H. S. Torr, and Efstratios Gavves. 2018. Long-term tracking in the wild: A benchmark. In *Proceedings of the European Conference on Computer Vision (ECCV'18)*.

[65] Miao Wang, Guo-Wei Yang, Shi-Min Hu, Shing-Tung Yau, and Ariel Shamir. 2019. Write-a-video: Computational video montage from themed text. In *ACM Transactions on Graphics (Proceedings SIGGRAPH-Asia)*, Vol. 38. Article No. 177.

[66] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip H. S. Torr. 2019. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1328–1338.

[67] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. 2017. Simple online and realtime tracking with a deep association metric. In *IEEE International Conference on Image Processing (ICIP'17)*. IEEE, 3645–3649.

[68] C. Wright, J. Allnutt, R. Campbell, M. Evans, R. Forman, J. Gibson, S. Jolly, L. Kerlin, S. Lechelt, G. Phillipson, and M. Shotton. 2020. AI in production: Video analysis and machine learning for expanded live events coverage. *SMPTE Motion Imaging Journal* 129, 2 (2020), 36–45. https://doi.org/10.5594/JMI.2020.2967204

[69] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. 2013. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'13)*.

[70] Ke Xie, Hao Yang, Shengqiu Huang, Dani Lischinski, Marc Christie, Kai Xu, Minglun Gong, Daniel Cohen-Or, and Hui Huang. 2018. Creating and chaining camera moves for quadrotor videography. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 37, 4 (2018), 88:1–88:13.

[71] Gene Youngblood and R. Buckminister Fuller. 1970. *Expanded Cinema*. P. Dutton and Co.

[72] M. Zarudzki, H. Shin, and C. Lee. 2017. An image based visual servoing approach for multi-target tracking using an quad-tilt rotor UAV. In *International Conference on Unmanned Aircraft Systems (ICUAS'17)*. 781–790.

[73] Xuaner Zhang, Kevin Matzen, Vivien Nguyen, Dillon Yao, You Zhang, and Ren Ng. 2019. Synthetic defocus and look-ahead autofocus for casual videography. *ACM Transactions on Graphics* 38, 4, Article 30 (July 2019).

[74] Zhipeng Zhang and Houwen Peng. 2019. Deeper and wider Siamese networks for real-time visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19)*.

[75] Liang Zheng, Zhi Bie, Yifan Sun, Jingdong Wang, Chi Su, Shengjin Wang, and Qi Tian. 2016. MARS: A video benchmark for large-scale person re-identification. In *European Conference on Computer Vision.*

[76] Zheng Zhu, Qiang Wang, Li Bo, Wei Wu, Junjie Yan, and Weiming Hu. 2018. Distractor-aware Siamese networks for visual object tracking. In *European Conference on Computer Vision.*

[77] John G. Ziegler and Nathaniel B. Nichols. 1942. Optimum settings for automatic controllers. *Trans. ASME* 64, 11 (1942).

Received December 2020; revised December 2021; accepted December 2021