Supplementary Material for: LookOut! Interactive Camera Gimbal Controller for Filming Long Takes Videos at http://visual.cs.ucl.ac.uk/pubs/lookOut/scenes.html)

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

1 INTERFACE FOR FILMING LONG TAKES

There is existing work in the literature that allows for expressing film scenes in a standardized form [3, 8, 9]. The Prose Storyboard Language [8] provides a grammar that's readable by both humans and machines for expressing all aspects of a film scene, including where actors are located with respect to one another and the scene, camera framing and positioning, and how multiple shots are sequenced and stitched. The Prose Storyboard Language and other languages like it are powerful, because they provide a standardized communication method throughout the entire film-making pipeline, ensuring the overall vision or goal of each scene and the overall film is met.

Instead, we focus on one key component of the pipeline: helping the camera operator film the shot. At the heart of LookOut's utility is the ability to offload the task of pointing the camera at actors from the camera operator. We call camera pointing like this a camera *behavior* (Section 1.1). We house such behaviors in a *script* (Section 1.3). Each script is a series of sequential camera behaviors that are triggered in-turn when a *cue* (Section 1.2) is hit. Cues can be as simple as a spoken word or as nuanced as an actor entering a particular part of the frame. The simplest script has no cues and only one behavior (keeping a single actor in frame for example), but they can be as complex as the operator wishes. The operator can also switch between multiple scripts during filming, allowing for flexibility depending on the circumstances. LookOut provides audible feedback through earphones to the operator, informing them when a cue is triggered or when a command is heard (Section 1.4). Figure 2 shows the GUI with an example script housing multiple behaviors and cues.

1.1 Behaviors

Within the LookOut GUI, the continuous domain of camera motions is organized into a menu of discrete and parameterized behaviors. Examples include standing still or panning left 30°. The existing behaviors in LookOut come from requirements gathering with two film-makers, but further behaviors can be programmed in the future, and users can already cover a broad range of use cases by chaining behaviors together.

Actor Based Framing: In most filming scenarios, one or more humans is the focus of attention. So a shot is driven either by actor monolog/dialog or by them performing physical movements. Unsurprisingly, a user designing a script with an actor-based behavior must first attach a specific ID to that behavior. An actor ID is mostly just a name for now, and the discriminative appearance info for each actor in the pool will only be filled in on-set at the startup phase. The user then specifies how this behavior frames that particular actor by placing a dot for the actor in the desired part of screen-space, e.g. on the left of the frame (see Figure 1). Often, scenes involve multiple actors. The interface for a multi-actor behavior is essentially the same, and LookOut will later optimize, striving to satisfy all the behavior's constraints.

Not all movements the actors make on screen require the camera to move. Camera movements must be *motivated* [1, 5]. A motivated camera movement or pan draws the attention of the viewer. Action shots may require tight and fast camera pans to keep a subject in frame, whereas a slow indoors shot would not benefit from quick camera pans when a subject rocks side to side.

For each actor's location, the user can specify an elliptical area of leniency, where actor movements are not immediately converted into camera movements. The LookOut GUI provides further behaviors for actor-framing. Especially for storyboarded takes, a path behavior lets the user spell out the framing of the actor over time. For example, a camera may pan to follow where an actor gazes when searching, or may pan to look ahead in a dynamic running shot. The path is constructed in screen space from a set of dots, with the distance between each point signifying how fast the camera moves in that part of the path.

Non-Actor Behaviors: A few of the available behaviors are independent of actor framing. A panning behavior makes the camera "scan" the scene, with a specified yaw or pitch direction, speed, and range. The banking behavior rolls the camera (in response to measurements from the IMU), simulating the effect of an aircraft dipping one wing while changing direction. The UI simply exposes the options for these behaviors using drop-down menus and text-entry fields.

1.2 Cues

Camera movements are often deliberate, triggered by changes in the scene or the progression of the action. LookOut responds to these changes in the scene to initiate each successive behavior in the chain. To do this, LookOut monitors events that the user designated in the GUI as being significant cues for each context. LookOut informs the user through quick audio feedback when the next cue is hit. LookOut can currently monitor for the following cues:

Actor Appearance and Disappearance: Actors coming into and out of frame can signify a new camera behavior. This cue could signal that a new actor is to be followed or that an informative pan is to take place. The user can specify which actor LookOut should monitor and how sensitive LookOut is to that change.

© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
0730-0301/2022/03-ART30 $15.00
https://doi.org/10.1145/3506693

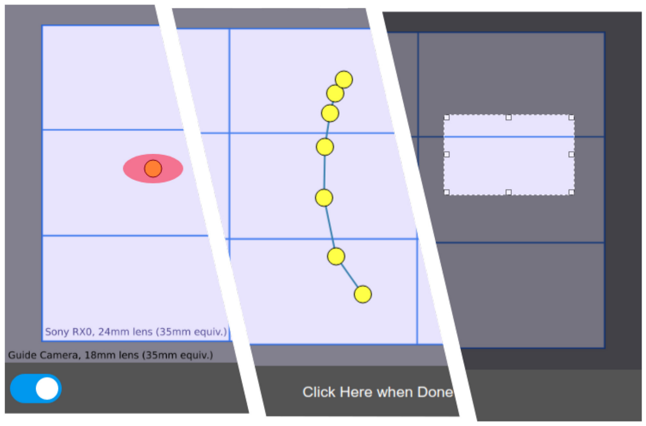

Sony RX0, 24mm lens (35mm equiv.)
Guide Camera, 18mm lens (35mm equiv.)
Click Here when Done

Fig. 1. The operator can select where an actor must be positioned on screen with a yellow dot for location and red ellipse for leniency (left), string together multiple points for an actor path behavior (middle), and define an area for a landing zone cue (right). The blue grid represents the star camera's image space, including aspect ratio.

Landing Zone: We adapt the concept of a landing zone into a cue. This cue is triggered when the requisite actor enters a specific user defined part of screen space (see Figure 1).

Elapsed Time: Although rigid, we also allow control over how long a behavior runs, using an Elapsed Time cue.

Speech: Speech recognition is included as a cue in LookOut, paired with speech synthesis. The aim is to give LookOut basic dialog-system capabilities, analogous to the verbal instructions between the director and a camera operator. The user types trigger words into the GUI when connecting a cue to a behavior, and either the operator or an actor with pre-defined lines wears a lapel microphone. The UI rejects trigger words that are too close to distinguish.

Relative Actor Size: This cue is useful for shots where the subject's relative image frame size is important to the narrative or action on frame. For example, a subject may appear from the distance, but the camera is to remain agnostic to them until they appear with a large enough screen size.

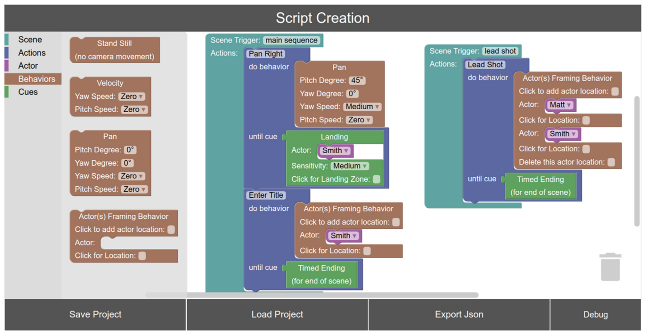
1.3 Scripts

change this language. The user puts together different scripts (or just one) to design his or her long take. A script is a preprogrammed sequence of camera behaviors. Ahead of filming, the user designs their long take using the LookOut GUI, typically on a laptop. They then export one or more scripts to the controller.

The chain of behaviors within a script is linked together by cues, which are explicit audible or visible events. The controller monitors for these events during filming, so a long take can be storyboarded and followed precisely, or it can be improvised in response to the operator's play-by-play instructions. Transitions between behaviors and between scripts are tuned to be responsive yet smooth.

1.4 Control and Feedback

During filming, the system carries out the user's requests and provides audio feedback about which script is being used and which



Stand Still (no camera movement)

Velocity — Yaw Speed: Zero; Pitch Speed: Zero

Pan — Pitch Degree: 0°; Yaw Degree: 0°; Yaw Speed: Zero; Pitch Speed: Zero

Actor(s) Framing Behavior — Click to add actor location; Actor:; Click for Location

Scene Trigger: main sequence
Actions: Pan Right
do behavior: Pan — Pitch Degree: 45°; Yaw Degree: 0°; Yaw Speed: Medium; Pitch Speed: Zero
until cue: Landing — Actor: Smith; Sensitivity: Medium; Click for Landing Zone
Enter Title
do behavior: Actor(s) Framing Behavior — Click to add actor location; Actor: Smith; Click for Location
until cue: Timed Ending (for end of scene)

Scene Trigger: lead shot
Actions: Lead Shot
do behavior: Actor(s) Framing Behavior — Click to add actor location; Actor: Matt; Click for Location; Actor: Smith; Click for Location; Delete this actor location
until cue: Timed Ending (for end of scene)

Fig. 2. Workspace GUI for Script creation. Two scripts (shown with light blue "blocks") can be seen in this workspace. The left panel houses different structures for defining camera behavior. The Behaviors tab is open and displays some of the camera framing modes (in brown) available to the operator. Green "blocks" are cues, i.e. events that are being monitored, to then conclude a behavior and/or start the next one.

behavior the system is performing. These requests come in the form of speech commands, spoken by the user to interrupt a script, restart it, or jump to alternate scripts.

Leniency from User Radii. For advanced users, $\mathbf{q}$, $\mathbf{v}$, and $\mathbf{a}$ would be made available for fine control over leniency. However, in our UI implementation, leniency curves are abstracted into a single radii pair, (r_x, r_y), for each axis that the user can specify via an ellipse in the UI. Note that this pair is normalized by guide camera image space size. We calculate each axis of the finer values using $\mathbf{r}$ as

$$a = 1.2r - 0.005,$$
$$v = 42 * (r - 0.5)^8, \text{ and}$$
$$q = \frac{20}{r + 0.01}.$$

2 IMPLEMENTATION DETAILS

2.1 Scripting and the GUI

Implementation. We use the Blockly [4] library for constructing the GUI for stitching together behaviors and cues into scripts. The user selects a script file for LookOut to read at startup. We make HTML/JavaScript extensions outside Blockly for actor framing, path framing, and landing zone cue. The scripts are output as Json formatted files and read by the system at startup.

Cues. With the exception of timed cues, all cues will continue waiting to be fired and LookOut defaults to the behavior preceding the respective cue meanwhile. When a cue is hit, the user receives audio feedback. If a cue is not met, then the user can push things along using an alternative mini-script they had programmed earlier.

Actor Following Behavior. The present version of LookOut relies only on bounding boxes output from the tracker, so each actor's tracked point, $\mathbf{p}^{\mathbf{T}}$, is set to halfway the width of the box in the x-axis and an offset below the top of the bounding box in the y-axis. This offset is set to 20% of the vertical height of the bounding box. We have found that setting the y-value to the midway point between the min/max of the bounding box gives undesired framing when the target is close to the camera and the bounding box's bottom frame is clipped by the image boundaries. This can be improved with pose information.

2.2 Voice Recognition

Script triggers can be fired anytime, but cue specific voice triggers can only be fired when relevant. For speed requirements, we require fast and reasonably accurate word or phrase recognition. This makes off-device services like Google's Cloud API unusable. Instead, we make use of the Porcupine [7] wake word detector.

2.3 Camera Zoom

Our combination of hardware limits zoom control to three attainable distinct levels of sensor-based zoom: 1.0×, 1.5×, and 2.0×.

2.4 Latency and Threading

Table 1 contains timing information for LookOut's main threads. Python's global interpreter lock prevents these threads from running truly in parallel, but there is a still a benefit to threading, especially since some threads are blocked due to heavy I/O with either gimbal hardware or the GPU. The threads are as follows:

1. The main thread handles camera I/O, gimbal metrics and control, and main LookOut scripting logic.
2. Tracker GPU tasks. Receives a camera frame from the main thread and passes detections and their appearance encodings to the CPU tracker thread.
3. Tracker CPU tasks. Receives detections and appearance encodings from the main thread, handles tracker logic, and sends results to the main thread.
4. Voice recognition logic, including Porcupine. Answers to main thread with recognized phrases.
5. Debug video storage and the debug display. Recieves current frame and control information from main thread.
6. Camera Zoom hardware interface. Answers to main thread with estimated zoom position and recieves desired zoom levels.

The overall latency from camera frame to tracker output and main LookOut thread refresh rate varies depending on a few factors, including camera exposure time, number of detectable objects in the scene, and the number of pending audio cues.

3 TRACKER DETAILS AND ALGORITHM PSEUDO-CODE

3.1 Recovery Details

In the main article, we outlined the recovery mechanism in the LookOut tracker. The length of recovery R, i.e., how many sequential frames a tracker must match to a detection before being counted as come out of recovery, is variable. If the actor was only lost for a short period of time, typically a four tracker updates depending on system latency, then the R is only two frames long. However, if the track is lost for longer, then the recovery length increases to four successful successive track hits. We do this to balance between being robust to distractors and accounting for short detector hiccups that would otherwise break tracking and require long recovery resulting in lost tracking.

3.2 Tracker Steady State Pseudo-Code

See Algorithm 1 for pseudo-code of the LookOut tracker's cost formulation strategy. The main explanation of the tracker is in the main article.

3.3 Tracking Evaluation

We evaluate trackers on two long representative scenes of our use case: Market (one actor scene at 3m20s with annotations every frame) and TwoPeople (two actor scene at 10m30s with annotations every five frames).

We obtain a ground truth bounding box by manually annotating input videos at 740×416. Market is annotated every frame, and TwoPeople has annotations for both actors every five frames. Similarly to the VOT-LT challenge, we do not include estimated annotations for when the actor is completely occluded. Based on VOT [2] and MOT [6], we employ metrics suited for actor tracking. All trackers are instantiated only once. Tracking-by-detection trackers are given the detection best fitting the ground truth as a start point and other trackers are instantiated using the first ground truth bounding box. Detection based trackers are all run on tiny-YOLOv3 output. Although trackers with multiple sequential components can be run in multiple threads for increased throughput, all trackers, including ours, are run in a single thread for fairness.

We use a bounding box IOU threshold to determine if the correct bounding box is output. For matching tracker bounding-box output to the ground truth target and distracters, we use an IOU of 0.5. 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, then 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, then 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, $\overline{D}$. The center of frame is used instead of the tracker's output when the tracker is lost and in case the tracker outputs some bounding box but the target is actually occluded.

Since our tracker has a random component, we average 40 runs on the same LookOut backpack computer.

4 USER STUDY

The storyboard given to participants during the user study is in Figure 3.

5 RESULT VIDEOS

For videos, please see the supplemental website, http://visual.cs.ucl.ac.uk/pubs/lookOut/scenes.html, that contains guide camera footage with overlays of the LookOut system's inner processes. Every video also has an associated higher quality “star" camera footage from a Sony RX0, albeit with a reduced bit-rate for consumption via web.

The telemetry/illustrated footage is displayed at a reduced frame rate to make ingesting data easier by eye. Available telemetry data are listed on the video's page.

Table 1. Breakdown of Task Times in LookOut

Operation	T (ms)↓
Frame Grab	8.2
Frame Resize	3.1
Gimbal Control	0.7
Gimbal Metrics Retrieval	8.9
Miscellaneous	1.6
Total	22.5
YOLOv3 and NMS	13.1
Cosine Encoding Network	3.8
Total	16.9
Tracker, Detection/Track assignment	3.3

There are three main threads. One main thread handles communication and control of the gimbal along with scripting logic. Gimbal Metrics Retrieval and Frame Grab are softly run in parallel. Tracker tasks are split into two threads (one handles GPU computation and the other handles the final detection/track assignment) and are run in a pipelined manner.

6 QUESTIONS ASKED OF SENIOR FILM-MAKERS

These questions were put to our three most senior film-makers, while getting them to think-aloud while critiquing Videos A, B, and C in Figure 4, all filmed using LookOut. These interviews and all experiments pre-date the Covid-19 pandemic.

- How many people would you need to shoot a scene comparable to this?
- How many specialists would you need for it?
- What is the level of skill necessary for the operators to have for this type of shot? (1–5)
- What equipment would you need to shoot a scene comparable to this?
- Would you need to change the set? As in, bolt something to the floor, drill holes, dig, and so on.
- Would you have a steadicam/gimbal operator do this or maybe have a crane?
- How much would it cost to get that equipment on site?
- How many takes / how consistent is the framing?
- How much planning goes into a shot like this?

To get natural reactions from the participants, we did not insist that they answer our specific questions. They did answer some of them. We made sure they at least gave their impressions about (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.

Here are responses from each of the three film-makers, in turn. These responses were written in short-hand, and we omit various stories and deviations recounted during the interview, as had been agreed in advance, because some stories are unflattering, and we felt this could help make the responses more candid.

Quotes from film-maker "W":

- To shoot a scene comparable to this? (video-A and overall): Depends on the budget. More people if you can afford it. Probably try to make due with 2. Myself and spotter, but still too rough a terrain and would need to rehearse a lot.

ALGORITHM 1: Cost Matrix Formulation Pseudo-code for LookOut tracker.

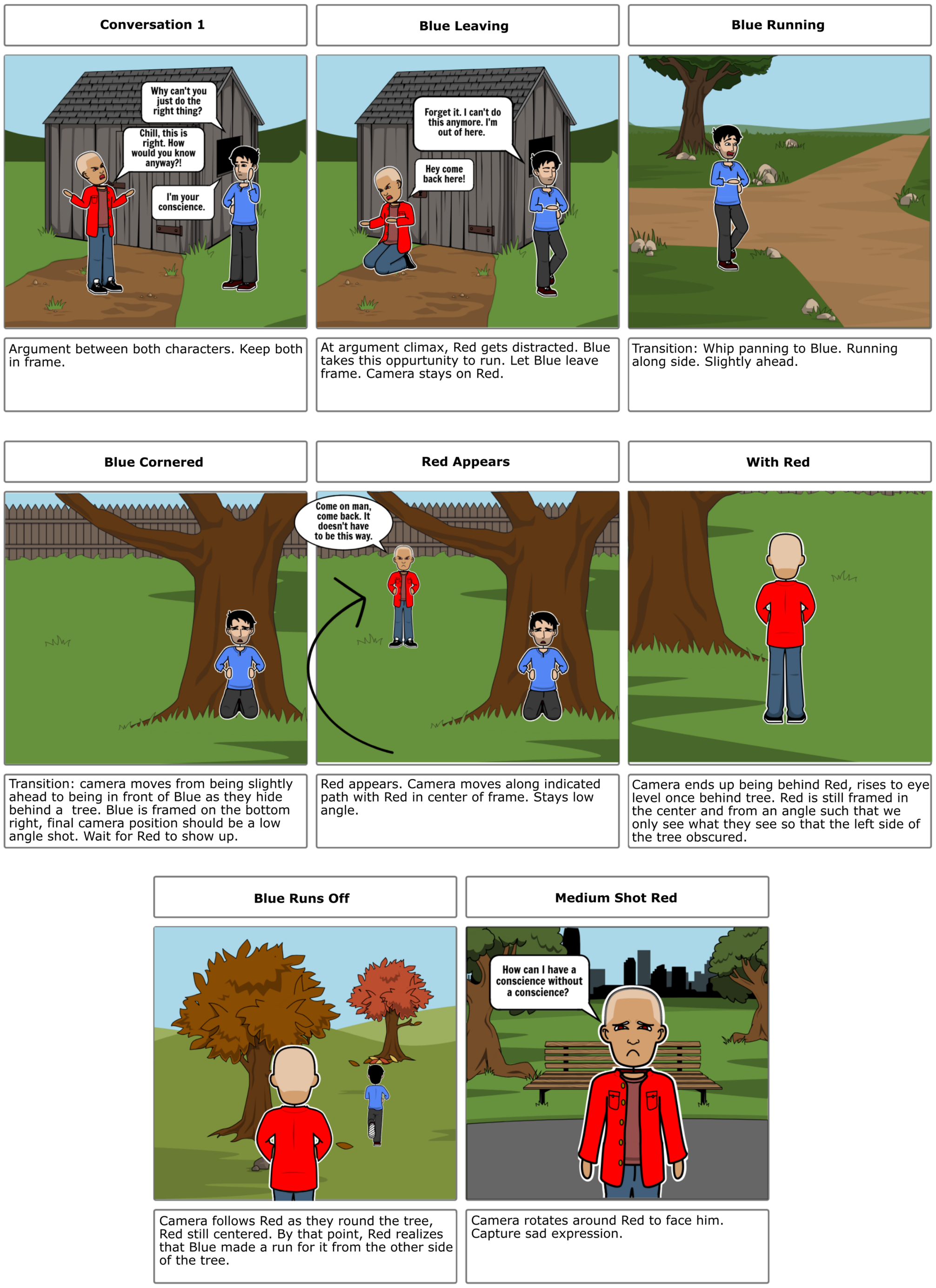
```
Input  : A set of tracks $T = \{t_i, i \leq N\}$. A set of detections in the
         current frame, $D = \{d_j, j \leq M\}$. A chain of features for
         each track of length $L_i$, $F_i = \{f_k, k \leq L_i\}$, and a single
         feature associate with each detection $f_{d_j}$.
Output : A matrix consisting of costs for each pair of detection, $d_j$,
         and track, $t_i$.
$C = [c_{ij}, i < N \text{ and } j < M]$ Overall cost matrix.
$C^{\text{iou}} = [c^{\text{iou}}_{ij}, i < N \text{ and } j < M]$ IOU based cost matrix.
$C^{\text{feature}} = [c^{\text{feature}}_{ij}, i < N \text{ and } j < M]$ Appearance based cost matrix.
$C^{\text{avgfeature}} = [c^{\text{feature}}_{ij}, i < N \text{ and } j < M]$ Average appearance based
cost matrix.
foreach track $t_i \in T$ do
    overlapCount = 0 ;
    foreach detection $d_j \in D$ do
        $C^{\text{iou}}[i, j]$ = computeIOU($t_i$, $d_j$)
        $C^{\text{feature}}[i, j]$ = computeAppearanceCost($t_i$, $d_j$)
        $C^{\text{avgfeature}}[i, j]$ = computeAvgAppearanceCost($t_i$, $d_j$)
        if $c^{iou}_{ij} < \tau^{overlap}$ then
            overlapCount = overlapCount + 1
        end
    end
    if overlapCount < 2 and $t_i$ is not lost or being recovered then
        /* Only one detection is competing for this track
           spatially. Don't rely on appearance costs.   */
        foreach detection $d_j \in D$ do
            $C^{\text{feature}}[i, j] = 0$
            $C^{\text{avgfeature}}[i, j] = 0$
        end
    end
end
$C = C_{iou} + C_{feature} + C_{avgfeature}$
```

- (How many takes?) Just need a few more takes to find your feet and get it right; learn your movements and theirs.
- Skill? Probably prefer to get a crane operator, costs 2k per day. Similar price but less faff with a steadicam operator, but depends on the shot.
- I'd also think about using a drone on shots like these—just need someone who's licensed. But it's noisy, so no good if you're recording the dialog. Fine to try on a bigger production, and with no [low] wind.
- (LookOut useful? Current capabilities?) Need to trust it first. Prefer if saw other film-makers using it repeatedly. Was the same for Red cameras. Saw the same with steadicam—nobody wants to be first, because an expensive set is expensive because of so many people and their time costs money.
- (How many specialists, skill level?) Hire someone on the basis of their experience or their style, not really quantifiable.

Quotes from film-maker "F":

- To shoot a scene comparable to this? (video-A): myself + spotter, but still too rough a terrain. Need to rehearse. Could use a drone, but they're loud, a hassle, and then someone else is deciding.
- (video-A, How many takes?) Maybe just not attempt it, opt for simpler [shot], tripod.

Synopsis: A man and his conscience have an argument. The conscience decides to run away. We follow as the man chases his conscience. **Camera Style:** Smooth in transitions and actor framing. Long take.


Why can't you just do the right thing?
Chill, this is right. How would you know anyway?!
I'm your conscience.
Forget it. I can't do this anymore. I'm out of here.
Hey come back here!
Come on man, come back. It doesn't have to be this way.
How can I have a conscience without a conscience?

Conversation 1: Argument between both characters. Keep both in frame.

Blue Leaving: At argument climax, Red gets distracted. Blue takes this oppurtunity to run. Let Blue leave frame. Camera stays on Red.

Blue Running: Transition: Whip panning to Blue. Running along side. Slightly ahead.

Blue Cornered: Transition: camera moves from being slightly ahead to being in front of Blue as they hide behind a tree. Blue is framed on the bottom right, final camera position should be a low angle shot. Wait for Red to show up.

Red Appears: Red appears. Camera moves along indicated path with Red in center of frame. Stays low angle.

With Red: Camera ends up being behind Red, rises to eye level once behind tree. Red is still framed in the center and from an angle such that we only see what they see so that the left side of the tree obscured.

Blue Runs Off: Camera follows Red as they round the tree, Red still centered. By that point, Red realizes that Blue made a run for it from the other side of the tree.

Medium Shot Red: Camera rotates around Red to face him. Capture sad expression.

Fig. 3. Storyboard given to participants to film using LookOut during the user study.

- (video-B, special skill?): Probably not [needed] for most people; common to be riding on something driven by others.
- (video-B would do differntly?) I like the orbiting, I'd like more face. Tighter framing of guy.
- (video-C): Gimbal, big monitor to make framing easier.
- (video-C how many takes?): Practice first, if they move too quickly, or you trip... Be CLEAR in your direction first. DJI Ronan—majestic mode can't respond to quick movements.

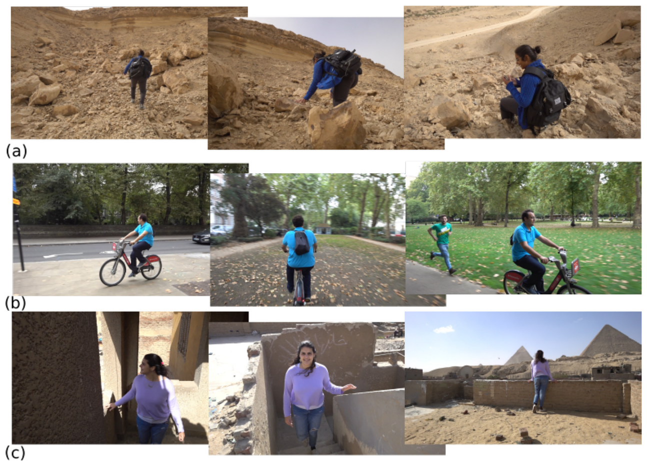


Fig. 4. Videos shown to senior film-makers. (a) Rocky escarpment: camera operator climbing on foot and with one hand free. (a) Bike ride: camera operator also riding a bike. (c) Pyramids: camera operator walking backwards on stairs.

- (Skill level? For all 3 videos:) If I really respected them, I'd tell them to get on with it. Vs. more novice, I'd give specific instructions. Not really detailed. Can always be a little different—so many variables.
- (What equipment?) Xion crane—but motor on back covers LCD screen! And then mirorless cameras, screen is too small. Usually Sony A7 series, manual for Zoom, autofocus, or remote focus wheel in another hand. But those become a nuisance when limited time. (Later, after seeing LookOut and understanding it) Big thing with this, that you wouldn't need to worry about it. Use it when you can't look at the screen, and need to pay attention to other things. Low or high angles, when you can't see the viewfinder.
- (On seeing Lookout) That would be so helpful! Especially in those Run & gun situations, documentary, travel, journalism. If you're filming something that won't happen again, you can focus on the other things.
- (Would you use it yourself?) Definite market for this; people could be funny about losing control [with motion control]; here purists could say it's part of the unique take.
- (Asked us if they could try it out "when we start selling them"—really?) [Well, I] can't see it in high-end commercial feature film. There, you have time on your side. ... Next time, let me know. I work with a lot of cinematographers. Lots of contacts who would like this. Should talk to DJI—probably most popular.
- (What's missing from LookOut?) Would you be able to control the zoom?

Quotes from film-maker "G":

- To shoot a scene comparable to this? (video-A): Use a gymbal camera with spotter to lead, move differently: smaller steps to minimize up and down; Or clear a path through boulders for walking, using a digger. Path needs to be out of shot obviously.
- (video-A equipment): Maybe 2-handed gymbal to reduce side-to-side; hold gymbal forward, at level of stomach.
- (video-B): Maybe Segway or cart driven by another person, so operator can focus on shooting.
- (Skill for video-B?) Should have understanding of camera work; need not be technical.
- (video-C): Definitely spotter leading me up the steps, on my shoulder, plus motorized single-handed gimbal, Easyrig [easyrig.se].
- (On seeing LookOut) That's amazing!
- (LookOut useful?) Does it know something about the scene? (answered him and explained system) I probably would be comfortable with a speech command.
- (Would you use it yourself?) It wouldn't take me very long to trust it to get the shot I needed, if I got to see it working a few times. I could be more creative once I got used to it. [On big budget projects] not many things [shots] that I can only get the first time.
- (Current capabilities?) Focus is a massive consideration, can have it pulled just right, but dynamics of scene change. Or auto-focus goes wrong: change focus from person A to B, or from very near to very far, while I pan up to see a mountain.
- (Other features?) Nothing right now. Is resistance predefined? Would like to adjust that, maybe dynamically: example, walk toward building and then look up, so need resistance. Sony FS7 with Ronin S, film a lot in slow motion. Start at 50 or 100fps then slow down.

REFERENCES

[1] B. Brown. 2016. *Cinematography: Theory and Practice: Image Making for Cinematographers and Directors.* Taylor & Francis.
[2] Luka Čehovin, Aleš Leonardis, and Matej Kristan. 2016. Visual object tracking performance measures revisited. *IEEE Trans. Image Process.* 25, 3 (2016), 1261–1274.
[3] David B. Christianson, Sean E. Anderson, Li-wei He, David H. Salesin, Daniel S. Weld, and Michael F Cohen. 1996. Declarative camera control for automatic cinematography. In *AAAI/IAAI, Vol. 1.* 148–155.
[4] Google. 2021. A JavaScript Library for Building Visual Programming Editors. Retrieved from https://developers.google.com/blockly.
[5] S. D. Katz. 2004. *Cinematic Motion: A Workshop for Staging Scenes.* Michael Wiese Productions.
[6] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. 2016. MOT16: A Benchmark for multi-object tracking. arXiv:1603.00831 [cs]. Retrieved from https://arxiv.org/abs/1603.00831.
[7] Picovoice. 2019. On-device Wake Word Detection Powered by Deep Learning. Retrieved from https://github.com/picovoice/porcupine.
[8] Remi Ronfard, Vineet Gandhi, and Laurent Boiron. 2015. The prose storyboard language: A tool for annotating and directing movies. arXiv:1508.07593. Retrieved from https://arxiv.org/abs/1508.07593.
[9] Hui-Yin Wu and Marc Christie. 2016. Analysing cinematography with embedded constrained patterns. In *Proceedings of the Eurographics Workshop on Intelligent Cinematography and Editing (WICED).*