LACES: Live Authoring through Compositing and Editing of Streaming Video

Dustin Freeman, Stephanie Santosa, Fanny Chevalier, Ravin Balakrishnan, Karan Singh

Department of Computer Science, University of Toronto
{dustin,ssantosa,fanny,ravin,karan}@dgp.toronto.edu

ABSTRACT

Video authoring activity typically consists of three phases: planning (pre-production), capture (production) and processing (post-production). The status quo is that these phases occur separately, with the latter two having a significant amount of "slack time", where the camera operator is watching the scene unfold during capture, and the editor is re-watching and navigating through recorded footage during post-production. While this process is well suited to creating polished or professional video, video clips produced by casual video makers as seen in online forums could benefit from some editing without the overhead of current authoring tools. We introduce LACES, a tablet-based system enabling simple video manipulations in the midst of filming. Seamless in-situ integration of video capture and manipulation forms a novel workflow, allowing for greater spontaneity and exploration in video creation.

Author Keywords

Video editing; video production; compositing.

ACM Classification Keywords

H.5.2 [Information Interfaces and Presentation]: User Interfaces – Graphical user interfaces.

INTRODUCTION

Ever since consumer camcorders were introduced to the mass market, video has become a powerful way of capturing and sharing personal experiences. The current pervasiveness of recording devices and social media sites such as YouTube and Facebook has further increased casual video creation and distribution. However, unlike still photography, for which there are reasonable tools for quick, convenient editing, video manipulation typically requires a tedious and cumbersome offline process with tools that are often too complex or unsuited for casual video. This mismatch between the ease at which video can be captured and the difficulty of making edits is evident in the rawness of much of the videos posted in online forums today.

Whether documenting an event, performing surveillance, or recording takes for a scene of a film, the ratio of captured video to useful content is always high. During editing, a great deal of off-line time is devoted to sorting through the corpus of video content to recall, analyze and make cuts. On the other hand, there is often a fair amount of down-time for the camera operator during capture while passively watching the scene unfold. We refer to these two periods of low user activity as "slack time", and explore how overlapping production with pre- and post-production can make video authoring a more spontaneous and less time-consuming experience. Applying this strategy, we propose the seamless integration of video capture and manipulation operations into the same phase of a fluid workflow to make the authoring of videos more spontaneous and accessible.

We start with a few motivating scenarios of modern casual video production which we feel are under-served by current techniques, and we follow with a related work review. We next examine limitations and issues in the traditional video production workflow, and discuss how the live workflow with LACES addresses these shortcomings, as well as the associated new challenges and opportunities it introduces. We then describe the LACES system and present several use cases brought forth by users in an informal evaluation.

MOTIVATING SCENARIOS

Below are three scenarios where we feel traditional video tools are lacking. These examples target casual video makers collecting footage to be post-processed off-line. This footage can have a range of uses from reviewing and extracting information to producing video creations, which extend beyond short single-clip segments. We aim to demonstrate that the realization of such scenarios should be possible with low overhead.

Scenario 1. Curating Content

Isa is a parkour artist visiting Toronto. At High Park, she encounters another parkour group with a unique style. Isa has a social media following, and wants to upload a highlights compilation of their stunts from her phone. She starts recording the group and collects a series of clips, each focusing on different members as they make numerous attempts at each trick. Even after catching a good stunt, Isa continues recording through breaks and failed attempts to avoid missing anything. She ends up with half an hour of footage which she will review and patch together offline.

CHI 2014, April 26 - May 01 2014, Toronto, ON, Canada
ACM 978-1-4503-2473-1/14/04$15.00
http://dx.doi.org/10.1145/2556288.2557304

Scenario 2. Annotating Content

Jane is a primatologist; Taz is her assistant. They are examining how chimpanzees grasp objects while completing puzzles. Jane is standing behind a camera, and Taz sits across from a chimpanzee. Taz presents the puzzles to the chimpanzee, and helps prompt them when they get stuck or distracted. Jane's goal is to get the timestamp of the beginning and end of each grasping activity, for further analysis by herself as well as others. This is difficult to do precisely – she writes down the timestamp that she thinks is nearest to the start of a grasping activity. She will have to review the video later to refine these timestamps.

Scenario 3. Coordinating Content

Jill and Pradeep are high school students working on a film. The film will have a shot of Jill, dressed as a gorilla, climbing Toronto's CN Tower. Pradeep goes out and films the tower from an arbitrary vantage point on a windy day. Later, they film Jill making climbing motions in a studio. These two videos get passed to an editor, Cheryl, and while compositing, she finds it too hard to line up Jill's motions with the tower. She also points out that it would be nice to have a shot of Jill grabbing the top of the tower from just the right angle. Pradeep is sent out to film the tower from a different angle, and Jill records a few more takes, hoping that Cheryl can make it work in editing.

In the above scenarios, we see opportunities to improve workflows by integrating video capture and post-processing. This would provide a means for on-the-fly adjustments and spontaneity in the video making process.

RELATED WORK

This section reviews related work in manipulation techniques aiming at improving video authoring, as well as video tools designed for live usage.

Video Authoring

A key component of video authoring lies in the diverse manipulations of raw footage, such as browsing, cutting, assembling and compositing input segments to create a refined output clip. Goldman [10] and Borgo [6] both provide comprehensive reviews of the space. In particular, the latter divides the space into *video manipulation*, the browsing, editing and compositing of video segments, and *video visualization,* the visual representation of the segments. We are primarily interested in ad hoc video authoring, and therefore manipulation techniques that are quick and easy to perform in the midst of capturing, as well as visualizations that facilitate understanding or further decision-making about the video at a glance.

Video Navigation

A large body of research has explored video navigation. Scrubbing provides a real-time update of the current frame of the video as the user moves the slider along the timeline [19, 22]. ZoomSlider [13] and PVSlider [26] explore different dynamics of playback sliders for finer control while scrubbing. Victor demonstrates a comic strip view of the video that may be scratched [38]. We later describe how we apply scratching within our live context. Several projects have explored video navigation through direct manipulation, where the user directly drags objects in the video rather than relying on the timeline [8, 10, 17].

Another approach to facilitate browsing exploits visual cues for video content. Most common are thumbnail visualizations, which provide an overview of the entire video stream [14], or summaries, where duplicate content has automatically been discarded [20, 36]. Enhanced timelines, which augment the seeking bar to make it content-aware, have also been explored to aid quick retrieval of content of interest [1, 25]. Videotater [7] and Swifter [23] are two examples of seamless combination of interaction and visualization, displaying near-context thumbnails during navigation.

All of the above navigation techniques have the common goal of facilitating quick access to segments of interest through enhanced interaction or visualizations that spare the user from passively watching the entire stream at normal speed. While they have primarily been designed for offline manipulation, similar approaches are suitable in the context of live manipulation.

Video Manipulation: Editing

We refer to video editing as the act of cutting and joining pieces of one or more sources together *in time* to make one edited movie [24].

Effectively collecting video segments of interest is made difficult by the current crude nature of the play/pause status of cameras—they can either be recording, or not. Vine [39] allows for a more fluid time control, where the video only records when the user is touching their finger on the screen, functioning as a quasimode [27]. In the status quo and with Vine, the captured videos are immutable in that recording applications usually do not support editing, which is therefore performed off-line.

Several techniques support fully automated video editing based on content analysis of the footage, such as the sound track [5, 31, 33], or by leveraging meta-data captured during recording, such as geographical location [35]. All of these works reinforce the idea that video editing is usually tedious and painstaking, especially when performed separately from and long after the actual capturing. Fully automating the editing process eliminates this problem but is lacking in flexibility and control.

In contrast to prior work, we propose a seamless integration of manual editing capabilities with video capture, enabling quick cutting and slicing of segments of footage on set, in the midst of recording the scene.

Video Manipulation: Compositing

In contrast to editing, video compositing refers to the assembling of video segments together in *space* to make one composite video, mainly by combining different areas of each source frame [24].

Live compositing has been addressed for still photographs, with projects such as Group Shot [21], that allow the combination of several pictures of the same scene by rubbing to remove parts of the photo that are undesired. In a similar vein, Cinemagram [9] allows users to create hybrid photo and video, instilling dynamics to still images, similar in the spirit of Cliplets [16], but more instantaneous.

Live compositing of several video inputs, however, remains an open problem. Some works have explored compositing the present and the near-past together, where a person can directly compose themselves with their own shadow, played with a few seconds delay [32]. Similar systems include Dancing with myself [3], DELEM [15] and Social Comic [18]. These approaches involve specific settings, including a video projector or a green screen, and are limited to a single, pre-determined compositing style. We propose to extend such approaches to any video stream, by supporting simple yet rich compositing capabilities of the live stream with recently recorded videos, while providing instant feedback of the result.

Live Video Authoring

Progress in digital technologies has yielded a new form of movie pre-production: virtual production enables filmmakers to interactively visualize and explore digital scenes using CGI pre-visualizations [2]. Such techniques afford a visually dynamic, non-linear workflow, blurring the barriers between planning, capturing and editing. We bear similar motivation to this process, but with live streaming video.

In the realm of live manipulation of videos, video jockey (or VJ) interfaces have captured researcher and artist attention [11, 12, 34]. However, these interfaces only allow for live editing of pre-recorded video segments, and do not support compositing operations. In contrast, systems for live broadcasting as traditionally used by television networks do support live capture and editing, but these are typically targeted to professionals and require extensive training, planning and human coordination. Dedicated camera operators and editors are typically working together in such cases, whereas we propose interactions that a single casual operator can perform.

Vaucelle and Ishii [37] demonstrate a video storytelling interface for children, where several toys have attached cameras. Video recording is toggled by physical gestures. The authors delineate separate video-making activities of rehearsing, recording, and playback. Ryokai et al. explored instantaneous re-use of captured video for artistic purposes in I/O Brush [28, 29]. A physical brush records images of real-world texture, and allows users to “paint” these textures onto a special canvas. These are targeted for production of abstract artworks and do not support video authoring in the traditional way.

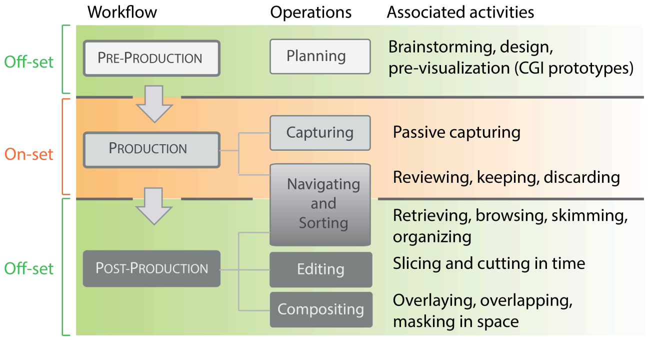
Workflow
Operations
Associated activities
Off-set
Pre-Production
Planning
Brainstorming, design, pre-visualization (CGI prototypes)
On-set
Production
Capturing
Passive capturing
Navigating and Sorting
Reviewing, keeping, discarding
Retrieving, browsing, skimming, organizing
Off-set
Post-Production
Editing
Slicing and cutting in time
Compositing
Overlaying, overlapping, masking in space

Figure 1. Traditional workflow

TRADITIONAL VIDEO PRODUCTION WORKFLOW

We first describe the traditional video production workflow and discuss the issues and challenges associated with it. We provide a broad workflow description here, encompassing both casual and professional video creation. We identify several limitations, and thus opportunities for exploration. In the subsequent section, we will introduce our proposed workflow within this unexplored space.

The traditional workflow for video production, shown in Figure 1, consists of 3 stages: pre-production, production, and post-production [2, 24]. Pre- and post-production typically take place separately and with different tools, enforcing a linear, forwards-only process and leaving little opportunities for back-and-forth iteration between the different stages.

Pre-production concerns the planning of the video. In professional environments, it includes activities such as brainstorming, designing, and pre-visualizing. The outcomes can be storyboards and scripts, to be used for the controlled, planned capturing of the video segments. In the context of casual or spontaneous usage, planning is not as thorough as in professional filming. An example of this is illustrated in Scenario 3 above, where desired shots are roughly planned out prior to execution.

Production concerns capturing the raw files, which are either informed by planning, as in Scenario 3 or ad hoc recording of live moments, as in Scenario 1. User interaction during capture is mainly passive, where the sole task of the operator simply is to observe the video as it is being shot. If a specific shot is needed, there may be many takes of a single scene before an adequate one is captured, possibly involving re-watching for verification

Post-production concerns manipulation of the raw files to generate ready-to-share movies. As with planning, this stage may be bypassed completely for casual users that consider the raw camera data the final product. The level of interaction can then range from minimal clip alteration to extensive, professional production. Actions during this stage include navigation, editing, and compositing.

Video navigation is performed to review footage and locate key moments, in support of editing and compositing. We refer to video editing as manipulating video segments *tem-*

porally. This involves slicing and removal of clips to isolate segments of interest, and assembling clips together. In contrast, video compositing concerns the manipulation of video segments *spatially*, where all or portions of the image from one clip are mixed or layered with those of other clips.

Editing and compositing are typically off-line processes, in both casual and professional cases. Tool support for video editing can vary in complexity from simple trimming of the start and end of each clip to joining multiple clips together with different transition effects. Video compositing tends to be an advanced practice, and tools supporting such actions are not generally targeted towards casual users. However, as shown in all three motivating scenarios, there can be a wide range of situations which require off-line processing.

Limitations and Opportunities

There are several limitations we perceive with the current workflow that highlight problem areas we seek to address in the design of our workflow.

Slack Time: In the traditional workflow, we observe that both production and post-production have "slack time"—periods of low-intensity user activity. Overlapping operations from production and post-production stages allows slack time during capture to be used more efficiently for performing editing and compositing. This makes video authoring a more spontaneous, flexible, and less time-consuming experience. For example, we see in Scenario 1 that there is opportunity for Isa to cut out footage of failed stunts while the group is not doing anything of interest.

Precision Timing: Desired editing operations are often reactive to precise events in video. For example, a user may want to add a bookmark to a clip when an exciting event occurs, but would not know to do this until the event has passed. Jane from Scenario 2 needs to review footage in order to achieve precision in event time markers. In another case, an interesting event may start before the user is able to turn the camera on—the operator intending to be parsimonious about recording to avoid having to sort through excess video content later.

Workflow Phase Separation: Since the toolsets for each phase of the traditional workflow are separate, a clear choice must be made when transitioning from one phase to another. In Scenario 3 for example, all footage must be captured on site first, and an off-line compositing process follows elsewhere. Since going back to shoot more footage when you have moved on to the post-production phase can be frustrating and costly [24], the ability to visualize raw video with some rough editing and compositing effects would be beneficial.

Coordination: In the final outcome of a traditional workflow, several different clips may contribute to a scene, whether alternating in time or composited together as with the CN tower scene in Scenario 3. While large production studios can afford to use multiple cameras, with one camera a scene must be recorded twice, inevitably with different timing due to real-world variation. The changes in timing must be later fixed by an editor.

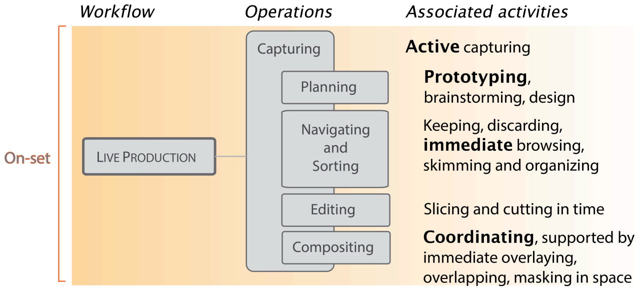
Workflow
Operations
Associated activities
On-set
LIVE PRODUCTION
Capturing
Planning
Navigating and Sorting
Editing
Compositing
Active capturing
Prototyping, brainstorming, design
Keeping, discarding, immediate browsing, skimming and organizing
Slicing and cutting in time
Coordinating, supported by immediate overlaying, overlapping, masking in space

Figure 2. Our proposed live authoring workflow

LACES: A FLUID WORKFLOW

We propose a workflow which blurs the boundaries of the traditional phases of video production, by allowing the co-located, simultaneous and seamless operation of planning, capturing, navigating, and manipulating in the same, fluid, flexible workflow (Figure 2). Our fundamental question is: How can a user interact with a live video stream? The live workflow we propose addresses many issues we find with traditional tools. We will now present the new challenges it presents, as well as design goals for a system supporting it.

Challenges in working with a live stream

Performing manipulation on a live stream while new video content is coming in presents several challenges.

Conflicting operations: Capturing and manipulating the same video are interfering operations *a priori*, and traditional video manipulation tools have expected video to be immutably stored in a file. While manipulating requires freeform temporal navigation, capturing will continually extend the file of interest—the meaning of "present time" will be constantly changing.

Catching up: Bezerianos et al. [4] discuss the difficulty of interacting with changing content, and techniques to "catch up" when the user did not observe a visual change. Silva et al. [30] present the Hold and Speed Up technique so users may apply annotations to the current frame of a live video. For a user to be able to edit while also paying attention to capture, the system must allow these two activities to share focus effectively—neither one can consume the full attention of the user.

No preview mode: While the saved video output from the system could conceivably be sent through a traditional post-production workflow, for the purposes of our exploration, the output is live. This means that there will be decreased opportunity to fine-tune a manipulation before it is applied to the live video. However, we see this as a trade-off: the traditional workflow applies video manipulations separate and off-line, whereas our workflow aims to apply these manipulations at the time of filming.

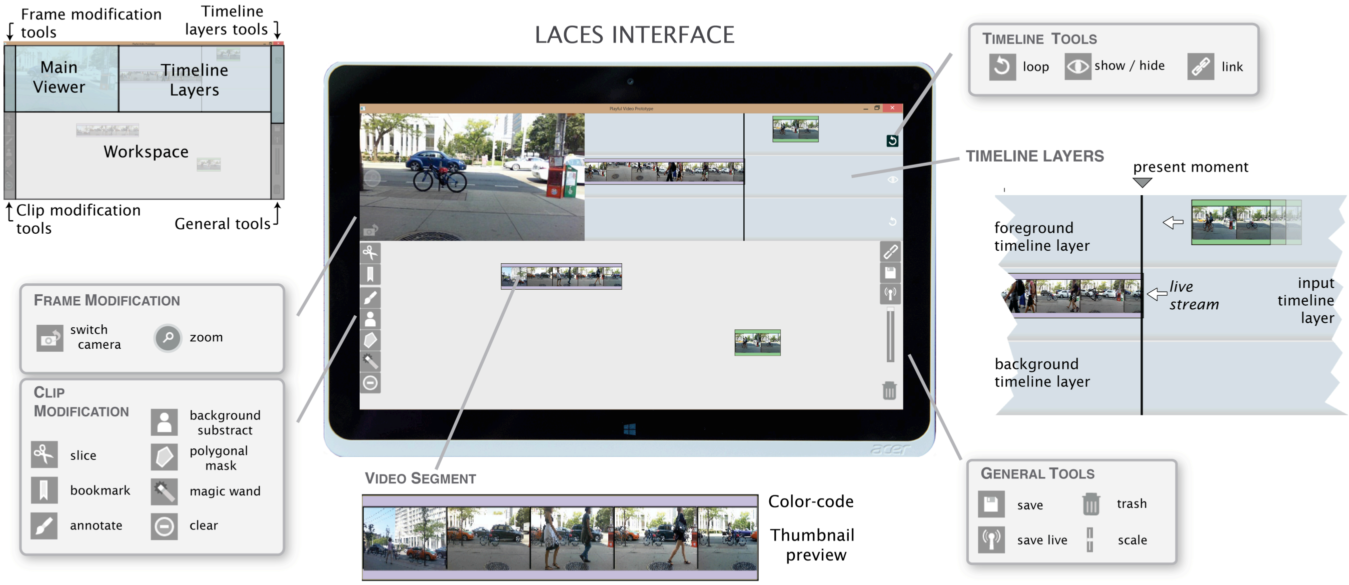

Figure 3. The LACES user interface, comprising the interactive main viewer (top left panel), timeline layers (top right) and workspace (bottom). Clip modifications can be performed through bi-manual interaction on the layers and side toolbar

THE LACES SYSTEM

We implemented LACES, a live video editing and compositing system supporting our fluid, live workflow (Figure 3). The interface contains three main interactive components: the *main viewer*, displaying the current video frame, the *timeline layers*, a stack of timelines featuring the captured live stream and additional layers for compositing, and the *workspace*, a library area where the user can save and arrange assorted video clips. We provide a set of tool palettes on both sides of the interface, for easy access while holding the tablet with two hands [40].

Overview

LACES is characterized by the continuous recording of the input video from the tablet. As the user launches the application, the recording of the live stream automatically starts. As time progresses, the associated movie strip progressively builds up in the *input timeline layer* (Figure 4). Videos clips on LACES timelines move right to left. In the middle of the timelines is a black vertical marker indicating the "present". The main viewer shows the real-time view of the camera. When passively capturing, the user can use LACES as a traditional recording device.

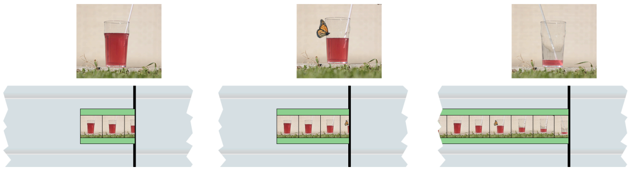

Figure 4. Capturing the live stream as seen in LACES. The main viewer (top picture) shows the real-time view of the camera. The recorded clip is visualized as a comic strip, progressively building up as time passes

At any moment, the user can interact with the live stream and assorted clips in the workspace. Video clips can be dragged to and from the workspace or the timeline layers. There are 3 timeline layers, ordered from top-to-bottom as *foreground*, *input,* and *background* (see Figure 3). The foreground and background timeline layers start empty, while the system continually adds new frames from the camera to a clip on the input timeline layer.

Video can be sliced at the input and dragged to the workspace or to other layers, in which case the viewer outputs the blended view of all clips including the input overlapping at the present marker. The real-time input stream can be blended with other camera footage, displaced it into the past, or even hidden from view to place focus on a prerecorded video clip in one of the other layers.

LACES provides several features to users, with interactions that can be performed while capturing: clip control, frame editing, clip transform control, layer control and saving.

Clip Control

Slicing and Bookmarking

A typical feature of video editing is the chunking of video into clips that may be sliced and arranged into sequence. We provide *slice* and *bookmark* buttons. The slice button separates the current clip into two clips on either side of the slice, and the bookmark button adds a marker to that frame within the clip. By default, if the user presses the slice or bookmark button, the corresponding operation is applied to the current real-time frame in the input layer clip—cutting it into two segments, or placing a frame mark for future reference. Slices are colored to provide additional visual cues for recall and organization.

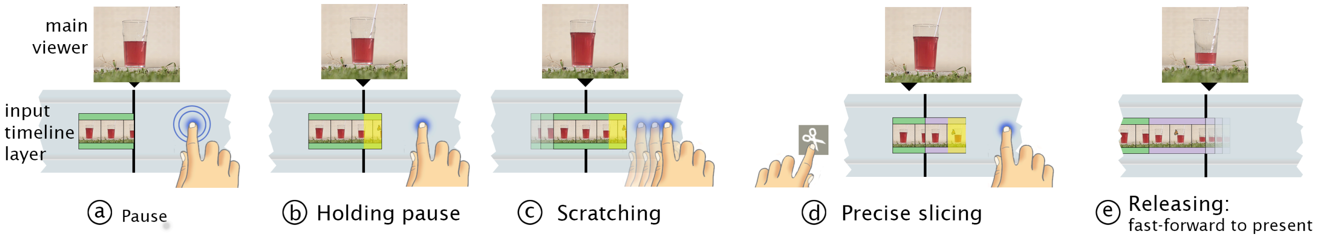

Figure 5. Manipulating clips on the input timeline layer. The user places a finger on the input timeline layer, which freezes the current frame (a); portions of the clip the user has not seen yet are shaded yellow (b). She scratches the clip into the past (c) then slices it at the desired frame (d). When she releases her finger, the input clip plays back to the present at an accelerated rate

Sorting

Clips can be dragged from the input to the workspace. We think of the input timeline layer as a queue of clips to be sorted. If not dealt with by dragging to the workspace, they move off the timeline to the left. However, removing a clip from the input closes the gap between its neighbouring clips, so all clips may eventually be retrieved.

Re-using

Clips can be dragged from the workspace, or even directly from the input timeline layer, to the foreground or background layer. The display of several clips underneath the present time marker is, by default, an alpha blend in the viewer. The meaning of foreground or background only comes into play with different blending styles, which we will discuss later. We support snapping when the ends or bookmarks of clips are dragged near those of other clips.

Navigating

Inspired by "scratching" as it appears in disc jockey (DJ) performance, the user can move both the background and foreground timeline layers left or right using their finger. If the user simply holds their finger still on a layer, it stops that layer from playing forwards and freezes it on a specific frame (Figure 5a-b). We refer to scratching as moving the medium while the play marker stays fixed, as in our interface or with a vinyl record and a needle. The term "scrubbing" is used when a user moves a player marker while the medium remains fixed, such as with most digital video player interfaces. This is a subtle but important difference—especially if multiple media are playing and one marker indicates the present-time position in all of them, as with our interface.

To support the typical video editing activities of slicing and bookmarking described earlier during capture, but it is very difficult to anticipate the correct time to slice or bookmark a live video stream until after the key event has gone past. To this end, we support *scratching the input*.

To apply a slice or bookmark operation to a specific time in the recent past, the user places their finger on the input layer. By holding it still, the viewer's display will be frozen at the exact time the user began the scratch. The user can scratch the input layer right to left to tune the exact timing of the desired slice or bookmark, and then press the desired operation's button to apply it to the frame underneath the present time marker. Since LACES is a live interface, the camera is still recording while the user is performing this operation. We provide a visualization of how new, unviewed video builds up during this operation (Figure 5b). When the user releases their hold on the input timeline layer at the end of any operations they want to perform, the input layer plays at an accelerated rate to catch up to the real time display of video (Figure 5e). The high-speed play is a useful alternative to a sudden transition to real time, giving the user a summary of the video they missed while focusing on their editing operation [4]. The user can slice or bookmark previously-recorded clips using a similar technique, with the operation button applying to whatever layer the user is currently touching, or to the input if none are currently touched. We find it possible to perform this bimanual operation without an obvious disruption in the filmed video [10].

When a user drags a clip from the workspace to a layer, it is possible to lose that clip when it progresses off the timeline. Thus, references each clip are maintained the workspace. Excess clips can be removed from the workspace by dragging them to the *recycle bin*. Frames in each clip are visualized as a comic strip, with a default frequency of one frame every three seconds. With longer clips, this scale can get cumbersome, so a scale slider is provided to increase the amount of time each displayed frame represents.

Examining our first two motivating scenarios, these simple clip controls would allow Isa to cut discard uninteresting parts and, during such times, recombine shots of cool stunts, which she can add bookmarks for, into a final highlights reel. Similarly, Jane would be able to review footage during capture by scrubbing back to determine precise timestamps corresponding the chimp grasping activites, eliminating the need to review all footage offline.

Frame Editing

Here we discuss options to control the input coming from the camera. First, note that we have a *camera flip* button that controls whether we use the tablet's back or front camera. Other frame edit controls are applied to frames as they come in from the input: *ink annotations*, and a variety of methods for creating. Annotations persist on the frame they are drawn over and on subsequent frames. *Masking operations* are also supported. The results from multiple different masks are merged into a final mask. If a clip has a mask and is blended with another layer, the mask replaces the default alpha blend with a direct pixel overwrite.

To perform *background subtraction*, the user must ensure the camera's scene will be stable and clear of any foreground objects. As the user taps the button, LACES captures a background frame and immediately begins masking out the background from any foreground. We used a basic hue-based discriminant on a Gaussian-blurred image, followed by an erosion and dilation pass. While this occasionally worked sufficiently in a carefully prepared environment, it did not in real world examples.

To create a user-defined *polygonal mask*, the user taps the polygonal mask button, and then sketches the mask outline directly on the main viewer. All pixels not belonging to the mask are hidden. This is useful to grab specific portion of a video that is not likely to move significantly.

The *magic wand* removes all pixels that closely match a given hue. To do this, the user taps the magic wand button and then taps a location on the viewer. The hue in a small region around the tap is averaged, and subsequent pixels that closely match that hue are masked out. This worked well with solid colors. In contrast to the user-defined polygonal mask, this suits objects that will change shape significantly, such as moving limbs and hands.

Multiple masks can be combined—for instance, a magic wand to remove all elements in the frame with a certain colour, and then a user-define polygonal mask to remove the remainder. All annotations and masks can be removed from subsequent frames using the *clear* button.

Clip Transform Control

We provide traditional *panning and zooming* capabilities. This is particularly useful when we have multiple video clips playing simultaneously, and want to create a spatial relationship between them for compositing.

Pan and zoom operations are performed directly on the viewer. Similar to the slice and bookmark operations, pan and zoom transformations are applied to the input timeline layer by default, or any currently scratched timeline layer. These operations are keyed to the frame they are performed on, allowing the user to "act out" a pan and zoom sequence as they scratch through a video clip, and then replay the transformations at regular speed as many times as desired.

The user applies a pan by dragging their finger in either dimension on the viewer. While zoom is typically performed with two fingers on touch interfaces, in most of our scenarios the user is holding the interface with both hands and can only operate it with their thumbs [40]. Thus, we provide a zoom handle on the left side of the viewer. Pulling this handle upwards or downwards increases or decreases the zoom of the current video clip. A tap on the handle resets the pan and zoom.

Layer Control

Each timeline layer can hold a collection of clips. In the input layer, incoming frames will always be added to the last clip. We provide a few simple layer controls.

The foreground and background layers have a *loop* toggle button. When the button is activated and the present marker reaches the end of a clip on that timeline layer, the timeline layer will shift back to the start of that clip.

The input timeline layer has a *hide* toggle button. The user can use this to view previously-recorded video on the foreground or background layers, without it blending with the input layer. Input is still recorded even if it is hidden.

We provide a layer *link* toggle button. By default, scratching any layer moves it independently of the other layers. When the layer link button is activated, a scratch on any of the layers scratches all layers together, preserve intra-layer timing relationships.

In Scenario 3, Frame Editing, Clip Transforms and Layer control as described above could be used to isolate Jill as she performs her climbing motions, scale her as required while filming the CN tower, and layer her climbing clip over this background footage.

Saving

We provide two modes to save the resulting video: *saving out*, and *live saving*. Saving out renders all video clips on the timeline, similar to a traditional nonlinear video editor. Live saving records the stream as seen on the viewer. In contrast to saving out, this preserves any live performance components, such as scratching back and forth.

Device and Platform Information

We implemented LACES on a Microsoft Surface Pro tablet, which provides high performance and memory capacity in a tablet that could be comfortably held in two hands, or one for short periods of time. For video processing, we use Emgu, a C# wrapper of OpenCV. We capture frames from the camera at 30 fps with 424x240 resolution.

INFORMAL EVALUATION

Designing a suitable evaluation was difficult - we could either assess quality of video outcomes from traditional versus LACES workflows, or we could perform a usability study of specific areas of the system (i.e. simultaneous scratching and cutting). The former is difficult, as our motivation for LACES was to enable editing where it did not occur before, and the latter would miss evaluation of the concept of capturing and editing in one interface.

We gave the LACES system to 4 people to use in a self-determined manner for extended periods (2 hours to several days each) in their own environments. Our goal was to observe their use cases and creations without imposing specific use scenarios on them, and to observe whether the interaction techniques we presented would be effective in fostering creativity in opening up new options for expression with video.

We found that our informal evaluation participants invented many interesting uses. One collected all clips from a comedy show of a performer laughing, intending to create a long track of her continually laughing. Another prototyped a two-view guitar lesson one could imagine broadcasted on social media. Another notable use case we observed is tracing; tracing is typically done over a still image, and if done over a moving image with a longer duration, it is called rotoscoping. However, our participant noted that tracing over a short, looping clip of a few seconds could lead to sketches that capture the movement sequence.

Participants also saw value of LACES as an improvement over traditional video capture—one referred to it as "*organized taping*" and noted that *"[what] I hate most about [video capture] is getting rid of the chaff because at the end of taping I have all this video I have to go through*."

DISCUSSION

We have presented an instance of a fluid video workflow that seamlessly integrates video authoring tasks usually performed at separate phases with different tools. We discuss the performance of this relative to our motivating scenarios, the challenges we identified in working with a live stream, and the potential for issues with cognitive load.

Our scenarios motivated a tool that enabled in-situ *Curating*, *Annotating* and *Coordinating* of video content. The LACES workspace, being adjacent to the live timeline layers, affords collecting and arranging video clips without having them disappear, and without committing them to a location on the timeline. LACES' compositing on multiple layers supports coordinating.

The input timeline layer clip was effective in providing access to live footage. One of the subjects described the movement as "*initially stressful*", but relaxed after recognizing he could collect clips in the workspace instead of having to use them immediately. In terms of the anticipated design challenges of *conflicting operations* and *catching up*, we have addressed these with scratching and accelerated playback. For the design challenge of *no output preview*, the lack of an additional viewer to preview changes is a trade-off to maintaining focus on the live action.

There is potential for the cognitive load of performing capture and editing together to be overwhelming if the user is required to divide focus on both operations at once. LACES accounts for this by providing flexibility in the timing of editing. Interactions such as scratching input and workspace clip pooling allows for editing to be performed when appropriate. We found users naturally edited during slack time–moments of low interest, during or immediately following capture: ideal opportunities where low cognitive load can be capitalized on to perform quick edits.

USE CASES

We present a few use cases, some developed by users in our informal evaluation, to illustrate the design principles and features in this interface, starting with the simple case of collecting interesting sub-clips for later use during capture, and building to complex compositing scenarios. Our accompanying video demonstrates some of these use cases, as well as other interesting examples.

Editing during Capture

Karim turns on music at home and his two children start dancing to it enthusiastically. He wants to film this to share with friends. Traditionally, he would take out his smartphone and record the entire moment. This results in a large video file, tens of minutes long. Karim could edit this video down to key moments, but this is very time consuming, as after transferring it to his computer he will need to re-watch it to retrieve the interesting segments. If he shares the entire unedited video, it can be boring to watch and cumbersome to upload. Karim needs to be always recording video, but have an ability to mark key moments, and remove uninteresting sections. This ability to edit while recording is what our interface is designed to support.

Karim holds the tablet running LACES with both hands and films his children. He is able to see the live view in the viewer, but he can also see a comic strip view of previous frames. As the music changes in style, Karim can press the slice button to segment the input video into separate clips for each style. He drags these to the workspace and arranges them based on the different styles of music.

At one point, Karim turns to talk to his wife, and shortly after, both of his children sit down to take a break. Karim notices his children stopped dancing and wants to discard this part of the video. He scratches the input timeline layer back to the beginning of the break and slices it, and then scratches to the end of the break and slices again. He drags the sliced clip containing the break from the input timeline layer to the recycle bin. As the input video quickly plays back the live events that occurred while he was editing, he sees that one of his children jumps in a funny way. He scratches the input again to this exact point and sets a bookmark—this would be a good still to send in an email.

Storytelling With Props

Derek is a big fan of the Star Wars movies. His favourite sequence is the floating air speeders navigating through the forest moon Endor in *Star Wars VI*. This was filmed by compositing a green-screen scene on vehicle models in a studio with footage of a camera moving through a forest.

Derek takes a tablet with LACES and films a "flying" view by walking through wooded area in his local park. He slices and drags the scene clip to the workspace. He then drags the forest clip to the background layer, which now is blended with the live view of the tablet's rear camera. He takes out the speeder toy he brought and puts it in front of the rear camera (Figure 6).

He then moves it around so it appears to be steering to avoid trees. Since the forest clip was filmed at a walking pace, he scratches the layer to increase the playback speed, making the flight appear much faster. As Derek is moving

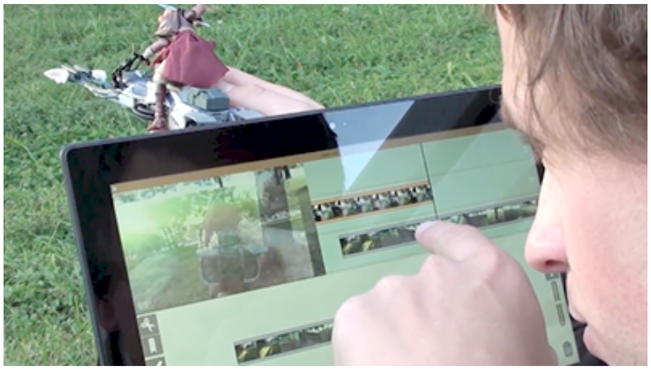

Figure 6. Storytelling with props, mimicking a speeder run from Star Wars VI. Demonstrates blending a live and recently-recorded video. The user scratches the previously-recorded video so it runs at a higher speed.

the speeder toy around live, blended with the forest clip, the movement of the speeder toy is itself being recorded. He can slice this clip and overlay it on the live view for yet another walkthrough of the forest. The ability to keep one source of a compositing constant while adjusting the other is very compelling as a prototyping tool.

Overlaying Faces

With a user-sketched polygonal mask, a user can overlay parts of their face on other objects or people in fun and interesting ways. This is seen in popular online videos as *The Annoying Orange* or the Quebecois *Têtes à claques*.

When Flint is at a fruit market and comes across a stall of oranges, he is reminded of the funny Annoying Orange videos, and is inspired to use LACES to put his face on an orange. He starts off by using the front camera and makes a few funny faces in anticipation of placing them on a fruit.

Flint presses the polygonal mask button and draws a mask carefully around the boundaries of his eyes and mouth. Once finished, everything outside the mask is blacked out, and he only sees his eyes and mouth. He makes funny faces for a few seconds, looking left and right and wiggling his tongue. He needs to capture a good section of his performance to use as an overlay on the fruit, so he scratches the input layer to the start of his performance, slices it, then scratches it to the end and slices it again. He drags the face-making video clip onto the workspace, then removes the polygonal mask from the input.

Flint presses the "flip camera" button so the viewer shows the stall of oranges in front of him. Flint drags the face-making clip from the workspace to the foreground layer. He presses the loop button on the foreground layer so that this short clip will loop continuously. The masked part of his funny faces clip is overlaid on to the live view. However, it is not aligned with any fruit in particular, so Flint uses pan and zoom controls to align it spatially. Flint holds his finger on the foreground layer to indicate operations will be applied to it. First, he uses the zoom handle to scale his face down so it will fit inside a fruit. Next, he pans his face so it appears on a fruit by direct dragging on the viewer.

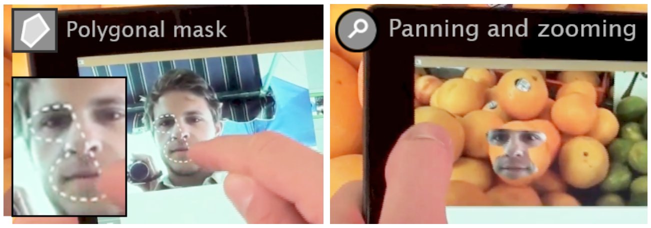

Figure 7. Overlaying face on objects. Demonstrates the use of a user-defined polygonal mask.

Fighting with Yourself

Patrick and his friend Felicity want to play a game where Patrick fights a video version of himself. They find a large wall with a uniform pink colour and Patrick stands in front of it. While Felicity films Patrick, she selects the magic wand and taps a pink part on the wall in the viewer. This masks out any pink pixels in subsequent frames.

Patrick turns to the side and makes a punch, pauses, and then makes a kick. Felicity uses our input scratching technique to isolate this clip form the input layer, and drags it to the workspace then clears the wand. The clip of Patrick now consists of a first part where he punches, a brief neutral rest in the middle, and a second part where he kicks.

Next, Felicity turns the tablet around, reversing the camera at the same time so Patrick can see himself. She drags the punch and kick clip to the foreground layer and holds it so the middle of the clip, where Patrick is neutral, straddles the present-time marker. This freezes Patrick's previous video in time. Real-time Patrick takes up a position opposite his pre-recorded self. Felicity can choose to play a punch or a kick by scratching the video in one direction or the other - forwards to play the punch section, or backwards to play the kick section, always returning to the middle (Figure 8).

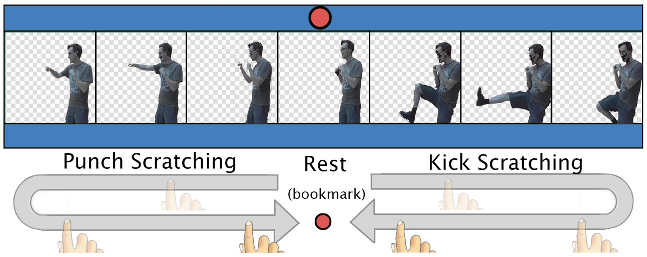

Figure 8. Self-fighting scenario: From a neutral frame in the centre, scratching right and back produces a kick; scratching left and back produces punch.

CONCLUSION AND FUTURE WORK

We have presented LACES, a system that combines video capture, editing and compositing to make video production more accessible to everyone. We have presented techniques to perform operations on live, moving media, methods for compositing live action with recently-recorded video, and several compelling use cases for LACES. We believe that that ability to operate on and annotate live, incoming data is important, whether video or not, as it closes the gap between capture and usage of media for any application.

While we have presented several short video authoring use cases, we have not explored scenarios of extended video activities, such as using the tool to help document a live event or produce a finished film, both of which expect a traditional video file as output. Our interactions can be extended for such cases, and we leave this to future work, and limit our exploration of live video manipulation to a smaller scale here.

REFERENCES

1. Alexander, J., Cockburn, A., Fitchett, S., Gutwin, C., and Greenberg, S. (2009). Revisiting read wear: analysis, design, and evaluation of footprints scrollbar. *ACM CHI*. 1665-1674.
2. Autodesk (2009). *The new art of virtual moviemaking*. Whitepaper.
3. Bartneck, C., Funk, M., and ten Bhömer, M. (2009) Dancing with myself: the interactive visual canon platform. *ACM CHI EA*. 3501-3502.
4. Bezerianos, A., Dragicevic, R., and Balakrishnan, R. (2006) Mnemonic rendering: an image-based approach for exposing hidden changes in dynamic displays. *ACM UIST*. 159-168.
5. Boiman, O., et al. Magisto. http://www.magisto.com/
6. Borgo, R., Chen, M., Daubney, B., Grundy, E., Heidemann, G., Höferlin, B., Höferlin, M., Leitte, H., Weiskopf, D., and Xie, X. (2012) State of the Art Report on Video-Based Graphics and Video Visualization. *Comp. Graph. Forum*. 31(8):2450-2477.
7. Diakopoulos, N. and Essa, I. (2006) Videotater: an approach for pen-based digital video segmentation and tagging. *ACM UIST*. 221-224.
8. Dragicevic, P., Ramos, G., Bibliowitcz, J., Nowrouzezahrai, D., Balakrishnan, R., and Singh, K. (2008) Video browsing by direct manipulation. *ACM CHI*. 237-246
9. Factyle. Cinemagram. http://cinemagr.am
10. Goldman, D., Curless, B., Salesin, D., and Seitz, S. (2006) Schematic storyboarding for video visualization and editing. *ACM Trans. Graph.* 25(3):862–871.
11. Hook, J., Green, D., and Olivier, P. (2009) A short film about vjs: using documentary film to engage performers in design. *ACM CHI EA*. 3491-3492.
12. Hook, J. and Olivier, P. (2010) Waves: multi-touch vj interface. *ACM ITS*. 305-305.
13. Hürst, W. (2006). Interactive Audio-Visual Video Browsing. *ACM MM*. 675-678.
14. Hürst, W. and Darzentas, D. (2012). Quantity versus quality: The role of layout and interaction complexity in thumbnail-based video retrieval interfaces. *ICMR*. 45.
15. Jimenez, G., Sanmartín, F., and Mazza, E. (2005) "DELEM - Delayed Mirror". *ACM ACE*. 196-199.
16. Joshi, N, Mehta, S., Drucker, S., Stollnitz, E., Hoppe, H., Uyttendaele, M., and Cohen, M. (2012) Cliplets: juxtaposing still and dynamic imagery. *ACM UIST*. 251.
17. Karrer, T., Wittenhagen, M., and Borchers, J. (2012) Draglocks: handling temporal ambiguities in direct manipulation video navigation. *ACM CHI*. 623-626.
18. Lapides, P, Sharlin, E, and Sousa, M.C. (2011) Social comics: a casual authoring game. *BCS-HCI*. 259-268.
19. Li, F.C., Gupta, A., Sanocki, E., He, L., and Rui, Y. (2000) Browsing digital video. *ACM CHI*. 169-176.
20. Ma, Y., Lu, L., Zhang, H., Li, M. (2002) A User Attention Model for Video Summarization. *ACM MULTIMEDIA*. 533-542.
21. Macadamia Apps. Group shot. www.groupshot.com/
22. Matjeka, J., Grossman, T. and Fitzmaurice, G. (2012) Swift: Reducing the effects of latency in online video scrubbing. *ACM CHI*. 637-646.
23. Matjeka, J., Grossman, T. and Fitzmaurice, G. (2013) Swifter: Improved Online Video Scrubbing. *ACM CHI*. 1159-1168.
24. Okun, J. A., and Zwerman, S. (Eds.). (2010). The VES handbook of visual effects: industry standard VFX practices and procedures. Taylor & Francis US.
25. Pongnumkul, S., Wang, J., Ramos, G., and Cohen, M. (2010) Content-aware dynamic timeline for video browsing. *ACM UIST*. 139-142.
26. Ramos, G. and Balakrishnan, R. (2003) Fluid interaction techniques for the control and annotation of digital video. *ACM UIST*. 105-114.
27. Raskin, J. (2000). *The humane interface: new directions for designing interactive systems*. Addison-Wesley Pro.
28. Ryokai, K., Marti, S., and Ishii, H. (2004) I/O brush: drawing with everyday objects as ink. *ACM CHI*. 303.
29. Ryokai, K., Marti, S., and Ishii, H. (2005) Designing the world as your palette. *ACM CHI*. 1037-1049.
30. Silva, J., Cabral, D., Fernandes, C. and Correia, N. (2012) Real-time annotation of video objects on tablet computers. *ACM MUM*. Article 19.
31. Shrestha, P, de With, P., Weda, H., Barbieri, M., and Aarts, E. (2010) Automatic mashup generation from multiple-camera concert recordings. *ACM MULTIMEDIA*. 541-550.
32. Snibbe, S. (2003) Body, screen and shadow. *San Francisco Media Arts Council (SMAC) Journal*.
33. Sumner, J. Vyclone. http://vyclone.com
34. Taylor, S., Izadi, S., Kirk, D., Harper, R., and Garcia-Mendoza, A. (2009) Turning the tables: an interactive surface for vjing. *ACM CHI*. 1251-1254.
35. Tompkin, J., Kim, K.I., Kautz, J., and Theobalt, C. (2012) Videoscapes: exploring sparse, unstructured video collections. *ACM Trans. Graph.* 31(4):68:1–68:12
36. Truong, B.T. and Venkatesh, S. (2007) Video abstraction: A systematic review and classification. *ACM Trans, Multimedia Comp, Appl.*, 3(1). Article 3.
37. Vaucelle, C. and Ishii, H. (2009) Play-it-by-eye! Collect movies and improvise perspectives with tangible video objects. *AIEDAM*. 23:305-316.
38. Victor, Bret. Inventing on Principle. Talk, 2012. URL: http://worrydream.com/#!/InventingOnPrinciple
39. Vine Labs. Vine. http://vine.co/
40. Wagner, J., Huot, S. and Mackay, W. (2012) BiTouch and BiPad: Designing Bimanual Interaction for Handheld Tablets. *ACM CHI*. 2317-232