Designing User Interface for Facilitating Live Editing in Streaming

Atima Tharatipyakul
Singapore University of Technology and Design
8 Somapah Road, Singapore 487372
atima_tharatipyakul@mymail.sutd.edu.sg

Pablo Cesar
Centrum Wiskunde and Informatica
Science Park 123
1089 XG, Amsterdam
The Netherlands
P.S.Cesar@cwi.nl

Jie Li
Centrum Wiskunde and Informatica
Science Park 123
1089 XG, Amsterdam
The Netherlands
Jie.Li@cwi.nl

ACM ISBN 978-1-4503-6819-3/20/04.
DOI: https://doi.org/10.1145/3334480.3383037

Abstract

Media assets, such as overlay graphics or comments, can make video streaming a unique and engaging experience. Appropriately managing media assets during the live streaming, however, is still difficult for streamers who work alone or in small groups. With the aim to ease the management of such assets, we analyzed existing live production tools and designed four low fidelity prototypes, which eventually led to two high fidelity ones, based on the feedback from users and designers. The results of a usability test, using fully interactive prototypes, suggested that a controller and predefined media object behavior were useful for managing objects. The findings from this preliminary work help us design a prototype that helps users to stream rich media presentations.

Author Keywords

User interface design; interaction design; streaming; live editing;

CSS Concepts

• **Human-centered computing~User interface design;** Web-based interaction; Usability testing;

Introduction

As technology advances, video streaming and television applications are continuously growing. Production tools for individuals or small teams become cheaper and more powerful, resulting in increasing popularity of live streaming as well as new challenges and opportunities. Media composition (mixture of graphics, live feeds, sound effect etc.) is one common practice that makes the streaming unique and engaging [6]. It has been a part of television broadcasting workflow, implemented in both commercial and experimental systems (e.g. [9]). While this feature is implemented and used by streamers [15], live media management, especially managing chat, is still challenging for them since they are already occupied with producing the main content (e.g., creative work [5]). As a result, it is difficult for live streamers to compose real-time media assets for live events. We aim to address this challenge and improve the quality of a live streaming.

Background and Related Work

Live streaming typically requires a set of hardware and software working together to prepare, capture, edit, and deliver live content to viewers, as described by various tutorials (e.g., [2,7,10,11,19]). Before a live streaming, streamers may craft a script and prepare *media objects* (e.g., video recordings and overlay graphics). Such preproduction stage may involve *streaming toolkits* such as Maelstream or Strexm for dynamic media objects (e.g., comment feed or subscriber alert) and various applications for different types of media object (e.g., Adobe Photoshop for a channel logo). The media objects, along with output from capturing device(s), are then put together in a *streaming application,* such as Open Broadcaster Software (OBS) Studio, Xsplit, Vmix, WireCast or Livestream studio. Some streamers also speed up their interaction with the application by using a third-party *controller*, such as Stream Deck or Touch Portal, for customized shortcuts. The application continuously encodes and uploads the content to *streaming platform(s)* such as YouTube live or Twitch, for viewers to see. These platforms usually come with additional features, such as a comment box or like button, to foster viewers engagement. The comment and other feedback from the viewers could be sent back to the toolkits to be streamed, as illustrated in Figure 1.

Though these tools support media composition and management during live streaming, the task is considered challenging for streamers. Such time-critical task is challenging even in professional television broadcasting [3,12], where a producer orchestrates a team to do the live streaming. So, each team member only gets specific tasks like creating and sending a limited number of media objects at a time (e.g., only a player name). However, live streamers usually manage

Related applications

https://www.maelstream.com
https://strexm.tv
https://obsproject.com
https://www.xsplit.com
https://www.vmix.com
https://www.telestream.net/wirecast/
https://livestream.com/studio
https://www.elgato.com/en/gaming/stream-deck
https://www.touch-portal.com
https://www.twitch.tv/

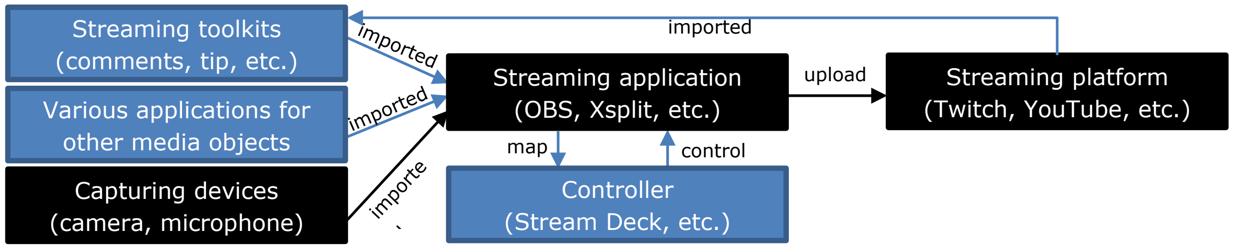
Streaming toolkits (comments, tip, etc.)
Various applications for other media objects
Capturing devices (camera, microphone)
imported
imported
importe
Streaming application (OBS, Xsplit, etc.)
map
control
Controller (Stream Deck, etc.)
upload
Streaming platform (Twitch, YouTube, etc.)
imported

Figure 1: Typical information and interaction flow between production tools. Blue boxes are optional.

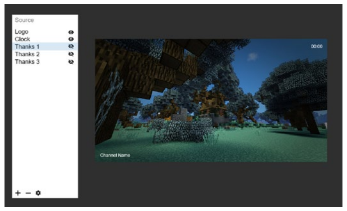

Figure 2: UI of the application in Design A (including the screenshot from *Minecraft*)

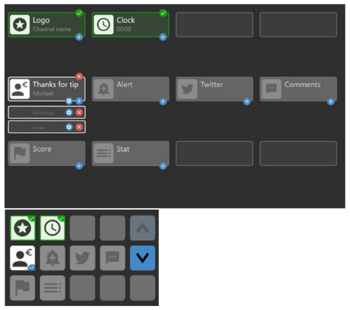

Figure 3: UI of the application and controller in Design B

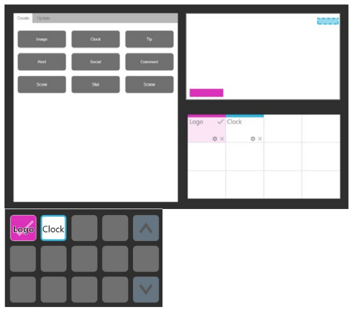

Figure 4: UI of the application and controller in Design C

the streaming alone [4]. They need to manage not only multiple streams but also real-time media objects. Some solutions suggest employing moderators [18] or bots [14] to assist live streaming. In this work, we focus on improving the user interface and interaction to facilitate live editing and streaming.

Initial Designs

We used iterative design process. Based on the insights from our previous work on production tools for television broadcasting [8,9,13], literature reviews of the streaming tutorials (e.g., [2,7,10,11,19]), and the on-going research in designing user interface for video applications [16,17]. We came up with four initial designs (A - D), which share some common features, and all have its own unique features (mainly about how to create, organize, and preview objects).

All designs consist of two parts: streaming *application*, where a streamer sees how media objects are put together on a PC or laptop, and *controller*, which consists of buttons to quickly control each media object from a mobile phone or Stream Deck. We automate some tasks and keep only relevant objects. In this way, users can easily select objects and assign each object with customizable behavior. **Fixed objects**, such as camera feed, remain throughout a live streaming. Tapping the buttons of the fixed objects on the controller will toggle their visibility. Users must manually remove them from the system when they are not needed. **Event objects**, such as comment feed or subscription messages, appear for an assigned time period. The event objects could be either (1) manually shown or (2) automatically shown on screen. Tapping the buttons of the event objects on the controller will remove the objects from the system. Media objects can be prepared before a live streaming. During the live streaming, a user may also create, duplicate, or import media objects using a template in the application. Event objects are automatically imported through a web service. Any change of objects is visible to viewers immediately or with delay (set by users). This feature allows users to verify and fix the object if there is something wrong before showing to viewers.

Design A (Figure 2) is a simplify version of OBS that can only create and toggle visibility of media objects. It serves as the baseline to other designs.

Design B (Figure 3) categorizes the media objects, and organizes them in queues, with preview on another screen. A user creates a media object by filling details in a popup form. The object button will then appear in a queue at predefined location. Only the first one in the queue is highlighted and can be triggered directly from the controller. The user could also work with other objects in the queue by clicking the text in the application or pressing arrow buttons on the controller for navigation. When an object is removed from the system, the next object in the queue will be shifted up. On-screen objects are highlighted with green border.

Design C (Figure 4) features media objects by their on-screen position, which are previewed as colored rectangles in the application. A user creates a new object by filling details in a tab. The object button will then appear in a grid, filling in the first available space. A user can scroll (in the application) or use arrow buttons (in the controller) to navigate through the set of objects. The color is used to distinguish the on-screen positions of media objects throughout the system. On-screen media objects are shown as solid

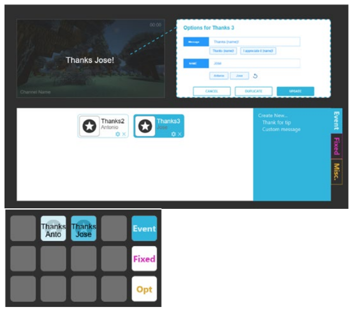

Figure 5: UI of the application and controller in Design D

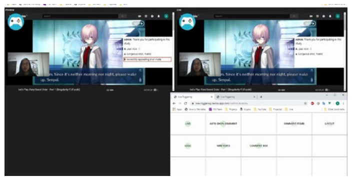

Figure 6: UI of the application and controller in Design A2, showing a game play from *Fate/Grand Order*

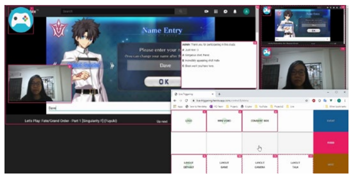

Figure 7: UI of the application and controller in Design B2, showing a game play from *Fate/Grand Order*

rectangles, while off-screen media objects are shown as semi-transparent rectangles in the preview panel and as white-background colored-border buttons.

Design D (Figure 5) features media objects by their behavior, organized by colored tabs, with preview on the top left of the application. A user creates a media object by filling details in a form that points to corresponding object location in the preview. The object button will then appear in a grid in next new space and will fill in empty used space only when all space is used. A user may not navigate in the controller, they can only scroll through object set in the application. On-screen media objects have saturated color, while off-screen media objects have light color. The selected object appears as 100% opacity, while other objects appear as 50% opacity in the preview panel.

The clickable prototype of all designs can be found in the supplementary material.

Design Feedback

We used the clickable prototypes to gather feedback from 2 game streamers (online, individually) and 3 designers (face-to-face, in group). We used Design A to demonstrate current practice of creating media objects and toggling their visibility. We then showed how the same tasks can be done with each design.

Both streamers agreed that all features should be pre-configured and come with templates, so beginners could start easily. Other comments included preference of a creation form that does not occlude the preview (Design D), fixed location of buttons on a controller for ease of remembering (Design B and D), and a request for hiding an object after a predefined duration.

In the collaborative design session with designers, they saw that manual object creation during time-critical live streaming is rare. There is no need to assign special space for this task. Identical buttons in the application and the controller waste the space on the application while icons on the controller require a user to refer to the screen. All designers agreed that highlighting new changes is important (Design C or D). Each designer preferred different ways to organize media objects.

The feedback suggested further study on how to organize media objects with a more interactive prototype, since opinions about this feature varied the most. Representations of media objects on the preview could be another interesting future consideration, but further studies on manual object creation may not be necessary at this point. Rather, the design should work on assumption that media objects could be dynamically created and imported through streaming toolkits.

Prototypes

As Design D got the most positive feedback, we decided to refine it to Design B2. Minor suggestions were put in Design A2 that mostly based on the current practice. Both designs place new event objects at the end in circular order, showing recent 12 objects. They feature different ways to organize and preview objects.

In **Design A2** (Figure 6), media objects can be organized into (user-predefined) folders. Design A2 application consists of two panels: preview and live. Any new media object or change will be highlighted in red and visible in the preview first, then three seconds later, it will appear in the live streaming. The three-second countdown on the controller assist the streamer to immediately remove an object before it goes live.

	Experience of Live
P0 (31-35)	(Mostly puzzle) game (~4 days per week, for 4 years)
P1* (31-35)	Game and coding (~once every two months, for 1 year)
P2 (31-35)	First person shooting game (~once every two days, for 2 years)
P3* (36-40)	Language teaching (~several times a week, for 3 years)
P4 (21-25)	Battle royal game (when he wants, for 1 year)

Table 1: Participant profile with their age range, type of live, typical frequency of live, and when you started broadcasting. P1 and P3 have design background.

	Design A2	**Design B2**
P0	87.5	**100.0**
P1	72.5	**75.0**
P2	**80.0**	45.0
P3	55.0	**57.5**
P4	**70.0**	62.5
Avg	**73.0**	68.0

Table 2: System Usability Scale of each design

In **Design B2** (Figure 7), media objects are organized based on their (user-predefined) behavior. Design B2 application has a main panel where minor changes will happen in-place. Major changes, such as layout, are displayed side by side. Instead of hiding a media object, dashed lines around the object indicate that it is visible to the streamer but not to viewers. We display an object identification number on the top-right corner of each object on the controller and the application, to help map objects across devices.

We implemented fully interactive prototypes at https://github.com/atima/live-triggering (see https://youtu.be/VrbYwSsnuKo for a demo video).

Usability Testing

We conducted a usability test with 5 male streamers (P0 - P4) with different levels of live streaming experience (Table 1). P0 participated in the previous iteration of interview and feedback while P1 – P4 was introduced to the designs for the first time. A researcher either sat with the participant (P1) or video called the participants (P0, P2-P4). For each design, we introduced main features and demonstrated how to toggle object visibility (see the demo video). Then, we let them freely try the design with their laptop (for the application) and mobile phone (for the controller). Since the application requires a webcam, it was infeasible for us to ask for another webcam and directly observe the online participant's interaction. Instead, we asked them to think aloud. We also encouraged the participants to ask questions, suggest features, and share their current practice during the trial. When the participants completed the trials, we asked them to remove a particular comment, simulating the common moderating task in a live streaming. The participant then filled in a questionnaire containing 5-point scale of overall ease of completing the task, System Usability Scale (SUS) [1], and usefulness of unique design features. After finishing all designs in counterbalanced manner, the participant filled in a final questionnaire containing 5-point scale of common design features usefulness and a question about their preference. The researcher translated and filled in the questionnaires for the Thai participants. Voice was recorded for later analysis. Each session took 60 – 90 minutes.

From average SUS score (Table 2) and ease of task completion rating (Figure 8), it seems that Design A2 was slightly preferred. However, individual SUS score and rating suggest otherwise. While we cannot derive statistical findings from 5 participants, increasing the number of participants may still be not useful at this point since each participant tends to have their own way of streaming. Also, each design has its own strengths and weaknesses, as reflected in participants' comments and usefulness rating of each design feature (Figure 9). Therefore, we focus on analysis of each design feature in both designs.

Design A2's strength is its straightforwardness. Side by side preview is quite common in professional live streaming setups, and highlighting could help a user easily identify new changes. Main weakness of this design is the delayed action feature, which could cause confusion for first-time users (P2). Also, the delay may not be appropriate for all media objects. Changing logos, for instance, works well with delay while changing layouts should require confirmation from the users (P2). New comments should be displayed without delay, so the comment owners could immediately see the comments, which increases the engagement (P1).

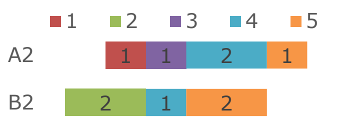

Figure 8: Ease of task completion of each design (1 – very difficult, 5 – very easy)

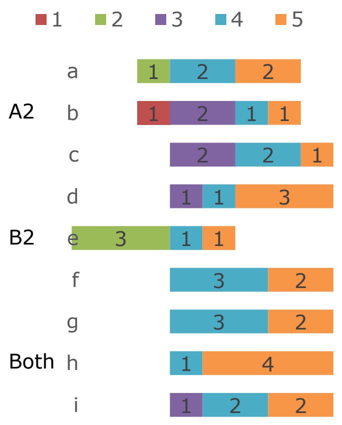

Figure 9: Usefulness of a) change highlighting, b) delayed action, c) side-by-side preview, d) major changes in side panels, e) visible/invisible objects for viewers, f) organizing media objects according to behavior, g) web-based application, h) controller, and j) predefined object behavior (1 – strongly disagree, 5 – strongly agree)

P0, P2, and P3, however, still liked the delay, which allows them to, for instance, remove a problematic comment before it appears to viewers (P0).

Design B2's strength is its ability to preview major changes in side panels. In contrast with Design A2 that requires streamers to recall layouts, Design B2 is easy to visually inspect them (P0). The participants also thought organizing media objects according to their behavior (all) and assigning object identification number (P0, P1, P2) are useful for them to quickly find the buttons. The visibility indicator (i.e. dash and solid border) in Design B2, however, could cause confusion. Though P1 liked this feature because he can see all hidden media objects at a glance, P2 and P4 suggested that the object should be hidden to the streamer too, so the invisibility is more obvious. Instead of visibility indicator, P0, P1, P2 also suggested that the application should display the selected layout at the center.

Common features were rated well by all participants. The concept of controller with predefined object behavior is new to the participants. Though some learning is required (P3), all participants demonstrated their interest in using it in practice. The controller removes the need for hotkeys, which may conflict with hotkeys for games (P1), and speeds up the interaction. However, they suggested a few improvements for the controller. First, the current design of controller buttons should be less abstract. Position of fixed objects may be remembered with some practice. A comment button, however, should include a few key words from the comment or show the user avatar to minimize the effort to double check in the application (P0, P1, P3). Second, there should be a mechanism to handle removal mistakes since accidentally pressing a wrong button is quite common (P0, P2, P3). Last, there could be additional filter for event objects (P1, P2, P3). For instance, the controller should highlight comments that require extra attention (P1). Other requests and suggestions include ability to control objects in the application as well (especially ability to move objects around – P1, P4), bookmark comments (P3), select screens to be shared during live (P4), support scrollable list in the controller (P1), and support multiple actions in the controller (e.g. by swiping left and right – P1).

When asked for the most preferred designs, participants gave various opinions. Design A2 was preferred for the side-by-side preview (P1) and delay feature (P2, P3). Design B2 was preferred for the side-by-side major changes (P0, P2, P4) and visibility indicator (P1). P1 and P2 suggested merging good points of each design together for the final design.

Conclusion and Future Work

This paper presents the design process of streaming tools that facilitate live object management. We proposed and tested several features, including how to create, organize, and preview media objects, especially when the media objects are continuously changing due to live events. The usability testing with 5 streamers demonstrated the potential of using a controller with predefined behavior to quickly manage media objects. Based on the findings, we will remove visibility indicator and refine designs B2 into a final design, then conduct another usability testing in more realistic scenario, where streamers play games while using the tool. We believe the tool could help improve the quality of livestreaming in the future.

References

[1] John Brooke. 1996. SUS-A quick and dirty usability scale. Usability evaluation in industry 189, 194: 4–7.

[2] Charlie Deets. 2018. Beginner's Guide to Streaming on Twitch. Retrieved April 10, 2019 from https://medium.com/@charliedeets/beginners-guide-to-streaming-on-twitch-dc2a7108fbd7.

[3] Arvid Engström, Oskar Juhlin, Mark Perry, and Mathias Broth. 2010. Temporal Hybridity: Mixing Live Video Footage with Instant Replay in Real Time. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 1495–1504.

[4] Eric Foote, Peter Carr, Patrick Lucey, Yaser Sheikh, and Iain Matthews. 2013. One-Man-Band: A Touch Screen Interface for Producing Live Multi-Camera Sports Broadcasts. Proceedings of the 21st ACM International Conference on Multimedia, Association for Computing Machinery, 163–172.

[5] C Ailie Fraser, Joy O Kim, Alison Thornsberry, Scott Klemmer, and Mira Dontcheva. 2019. Sharing the Studio: How Creative Livestreaming Can Inspire, Educate, and Engage. Proceedings of the 2019 on Creativity and Cognition, Association for Computing Machinery, 144–155.

[6] William A Hamilton, Oliver Garretson, and Andruid Kerne. 2014. Streaming on Twitch: Fostering Participatory Communities of Play within Live Mixed Media. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Association for Computing Machinery, 1315–1324.

[7] IvoryTV. 2018. Elgato Stream Deck Tutorial - Ideas, Tricks and Setup to Become a Pro! Retrieved April 10, 2019 from https://www.youtube.com/watch?v=3kPpI9BMUUs.

[8] Jie Li, Thomas Röggla, Maxine Glancy, Jack Jansen, and Pablo Cesar. 2018. A New Production Platform for Authoring Object-based Multiscreen TV Viewing Experiences. Proceedings of the 2018 ACM International Conference on Interactive Experiences for TV and Online Video, ACM, 115–126.

[9] Jie Li, Zhiyuan Zheng, Britta Meixner, Thomas Röggla, Maxine Glancy, and Pablo Cesar. 2018. Designing an Object-based Preproduction Tool for Multiscreen TV Viewing. Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems, ACM, LBW600:1--LBW600:6.

[10] Nick Nimmin. 2018. OBS Studio Tutorial 2018 (Make Your Streams Look Pro). Retrieved April 10, 2019 from https://www.youtube.com/watch?v=j2HzbY8E4yQ.

[11] Nick Nimmin. 2018. Live Streaming Remote Control for Streamlabs OBS! Retrieved December 8, 2019 from https://www.youtube.com/watch?v=CihXy5UVP64.

[12] Mark Perry, Mathias Broth, Arvid Engström, and Oskar Juhlin. 2019. Visual Narrative and Temporal Relevance: Segueing Instant Replay into Live Broadcast TV. Symbolic Interaction 42, 1: 98–126.

[13] Thomas Röggla, Jie Li, Stefan Fjellsten, et al. 2019. From the Lab to the OB Truck: Object-Based Broadcasting at the FA Cup in Wembley Stadium. Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, ACM, CS16:1--CS16:8.

[14] Joseph Seering, Juan Pablo Flores, Saiph Savage, and Jessica Hammer. 2018. The Social Roles of Bots: Evaluating Impact of Bots on

Discussions in Online Communities. Proc. ACM Hum.-Comput. Interact. 2, CSCW.

[15] Max Sjöblom, Maria Törhönen, Juho Hamari, and Joseph Macey. 2019. The ingredients of Twitch streaming: Affordances of game streams. Computers in Human Behavior 92: 20–28.

[16] Atima Tharatipyakul and Hyowon Lee. 2018. Towards a Better Video Comparison: Comparison as a Way of Browsing the Video Contents. Proceedings of the 30th Australian Conference on Computer-Human Interaction - OzCHI '18, ACM Press.

[17] Atima Tharatipyakul and Hyowon Lee. 2019. Towards a Better Video Comparison: Case Study. Proceedings of the Asian HCI Symposium'19 on Emerging Research Collection, ACM.

[18] Donghee Yvette Wohn. 2019. Volunteer Moderators in Twitch Micro Communities: How They Get Involved, the Roles They Play, and the Emotional Labor They Experience. Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, ACM, 160:1--160:13.

[19] Asayhi channel. Retrieved from https://www.youtube.com/channel/UCVPC3j5J9hu9LBMZGFj61Rg.