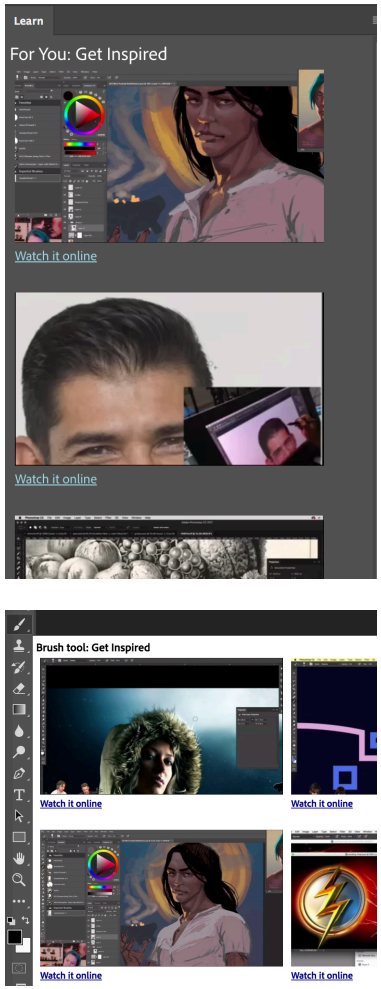

Figure 1: App-integrated clip recommendations in an image editing application. Short clips are selected from live-streamed videos using a mix of telemetry data and computer vision methods.

Software videos: Rich content and learning potential, but a challenge for sensemaking

C. Ailie Fraser
Design Lab, UC San Diego
La Jolla, CA, 92093, USA
cafraser@ucsd.edu

Mira Dontcheva
Adobe Research
Seattle, WA, 98103, USA
mirad@adobe.com

Scott Klemmer
Design Lab, UC San Diego
La Jolla, CA, 92093, USA
srk@ucsd.edu

Adobe Research
Seattle, WA, 98103, USA

Abstract
Video is a rapidly growing online medium; its rich content makes it both well-suited for communication and learning, and incredibly difficult to analyze and make sense of automatically. Our work explores ways to extract useful clips from online software videos and embed them as contextually-relevant demonstrations in software while the user works. We present methods for automatically extracting clips from live-streamed software videos and recommending clips based on user behaviour. To test these methods, we have implemented them in two systems as extensions to desktop software. We conclude by discussing future applications for these techniques to improve the ways videos are shared, searched, and interacted with.

Author Keywords
video; complex software; learning; recommendations

ACM Classification Keywords
H.5.1. Information interfaces and presentation (e.g., HCI): Multimedia Information Systems.

Introduction: Sensemaking with Software
People of all experience levels use complex software to complete tasks, and often turn to the web for help with

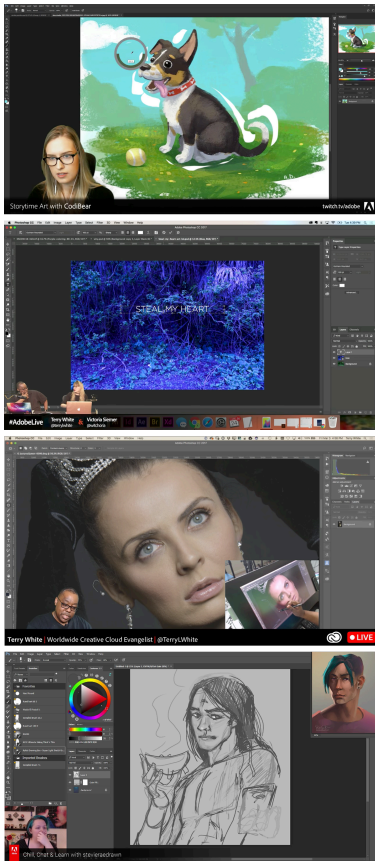

Figure 2: Examples of creative live-streams on Twitch and YouTube. Artists stream videos of themselves working in complex software on creative projects. (Sources: twitch.tv/videos/154575884, youtu.be/jP5fKeG1CkU, youtu.be/RtswnAYbrdk, twitch.tv/videos/152518965)

problems, learning new techniques, and finding inspiration – all of which are substantial sensemaking tasks. Prior work has demonstrated methods for mining & improving software tutorials (e.g. [3, 5, 9]) and recommending tools or actions in context [4, 10]. However, video demonstrations are often preferable for learning visual tasks [11], and videos remain difficult to parse and understand.

One source of software demonstrations not yet explored in the literature is live-streams. Many experts share their process by live-streaming their screen while they work on creative tasks (Figure 2); these videos are a rich and rapidly growing source for expert demonstrations. However each is usually several hours long, and finding relevant moments from this large collection can be difficult, making live-streams an under-utilized data source.

Our current work focuses on extracting relevant clips from long live-streamed videos and recommending them to the user in the context of their own work. Our goal is to automate the tedious sensemaking task of searching and browsing through videos, so that users can focus their energy and time on their primary task in software. We present two systems that automatically extract and recommend video clips from live-streams:

1) *App-integrated, offline processing*: This system processes videos offline using visual information and usage data, and recommends clips based on tool use. It is implemented as an extension for Adobe Photoshop.

2) *OS-wide, online processing*: This system searches and analyzes videos online using metadata and captions, and recommends clips based on tool use. It is implemented as a MacOS panel that responds to user behaviour in any accessibility-enabled desktop app.

While our current work focuses on live-streams, it can easily be extended to any software videos, such as tutorials. We conclude by discussing future applications for our techniques to improve the way we navigate video online, and help users complete tasks efficiently.

Tutorial and Video Interaction

Tutorials, both online and in-application, are a popular resource for software users [3, 5, 9]. However, they take time to author, and they only show the information that the author explicitly decides to show. Seeing a live demonstration is often a crucial component of learning for visual tasks; video tutorials are therefore a very popular resource [11]. However, videos are hard to digest and are not well-suited for working at the same time; users must switch context back and forth to follow along with a video.

Existing methods improve video tutorials by pausing them in response to user actions so they are easier to follow along with [11], segmenting them into clips for each step [2], and sharing alternate demonstrations with the community [8]. However, these methods require either textual descriptions or usage data from the videos, as well as detailed knowledge about the software in question, and thus none have become pervasive in everyday software use. Making use of the huge amount of video content that already exists and is being uploaded every day remains an open problem.

Extracting & Recommending Clips

Our work aims to make the knowledge that is present in software live-stream videos more available to people

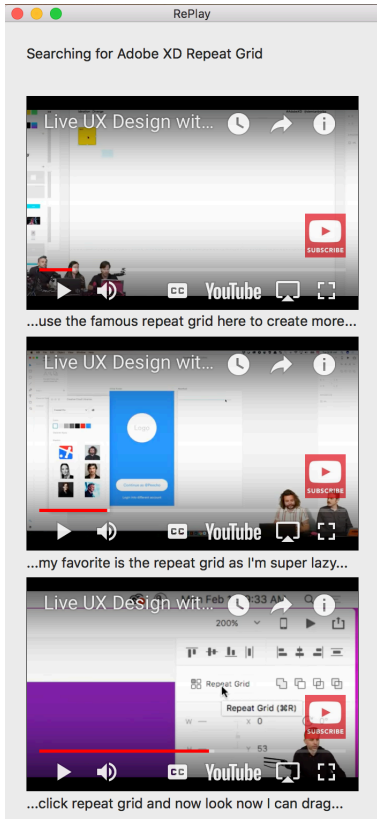

Figure 3: OS-wide clip recommendations based on the software currently being used. Short clips are selected from live-streams by searching YouTube in real time and identifying moments based on video captions. In this example the user has recently clicked on the "repeat grid" tool in Adobe XD.

while they work. This comprises 1) analyzing videos and extracting short clips, and 2) recommending the most relevant clips within the user's software. We have developed two methods for achieving both these goals, one app-specific method that processes videos offline and recommends clips in Adobe Photoshop, and one OS-wide method that processes videos online and recommends clips for any accessibility-enabled MacOS application.

App-Integrated Clip Recommendation

Inspired by prior work that extracts short instructional clips from long software video demonstrations [7], we built a system that analyzes live-streamed videos of Photoshop use using a mix of telemetry data from Photoshop and computer vision to extract short clips of various tools. Our initial dataset contained 8 videos (13 hours) from Twitch and YouTube, as well as telemetry data for each video, consisting of time-stamped event logs for tool selections and invocations.

Given a particular tool (e.g. brush) and each video's usage logs, we extract short clips using a heuristic approach that closely matches Lafreniere et al.'s [7]. We then crop each clip to the area of most visual change, calculated by taking the differences in pixel values between adjacent frames. Finally, we rank clips based on the amount of visual change (more is better), temporal location in the full video (closer to the end is better), and the user's behaviour in Photoshop (clips with the tools in common with the user are better).

We built an HTML Photoshop extension with three different interface modes: a persistent panel, an on-demand modal window, and tooltips that appear when mousing over a tool (Figure 1). It searches for clips from our dataset based on the user's recent tool use. Initial pilot feedback suggests that it is helpful to see expert demonstrations of the tools one is using while one works, but not all clips suggested are relevant to the task at hand, as each tool can be used for a wide variety of tasks. Careful consideration is required for developing improved heuristics for ranking clips.

OS-Wide Clip Recommendation

While offline processing allows detailed analysis of videos and enhanced display abilities such as smart cropping, it is time-consuming and usually requires knowledge about the software, either in the form of telemetry data [2, 7] or tool templates for vision methods [11]. To explore a more generalizable approach for video sensemaking, we have built a domain-general implementation that searches for and processes videos online in real-time. Our available dataset consists of all archived live-streams on YouTube that have a caption track (which most do, as YouTube automatically generates them by default).

This system searches for and extracts video clips using only the available online information from YouTube, that is, metadata and captions. We built a MacOS panel (Figure 3) that monitors the user's mouse clicks in any application that uses Apple's Accessibility API. To find relevant video clips, the system sends search requests to the YouTube API with queries containing the application's name and the name of the most recently clicked interface elements in the application (e.g., tool buttons). It retrieves the top videos that have captions and searches their caption tracks to find the specific moments in the video where the interface element(s) in question were mentioned. Finally, it ranks the retrieved clips and embeds the top clips in the panel as YouTube

embedded videos with specified start times. Currently, clip ranking is done by selecting the first occurrence of the keyword in each video, and users will have the option to view more clips from that video. As we gather long-term usage data, we will improve our clip ranking based on user behaviour.

An evaluation of this system is ongoing. Initial anecdotal findings and pilot test feedback indicate that this method is able to find reasonably relevant clips in a fraction of the time it would take a user to manually skim through the retrieved videos (which are on average about 1.5 hours long). Video streamers tend to describe what they are doing while they do it, as well as provide advice about making decisions (such as whether to use one tool over another), all of which are captured by the video captions.

The Future of Online Video

Contextual recommendation of video clips is just one potential application of the techniques we have presented. Another would be to improve the online browsing and searching process. Currently most video sites simply show a thumbnail and short description for each search result, which is rarely enough information scent for users to determine which video contains the information they are looking for. Enhanced video search interfaces could provide richer information such as key terms that are mentioned, tools or techniques that are used, and informative previews for key moments.

The literature on video summarization has presented techniques for reducing long videos to a selection of only the most important frames or clips (e.g. [12]), which can also help one explore a large collection of videos. We can exploit the properties of software videos in particular (i.e. that they are usually screencasts showing GUIs with buttons and menus) to produce video summaries that both shorten videos and highlight the key details that are needed to understand them.

Finding the right moment(s) *within* a given video is also a challenge; users can skim through the timeline and see small previews, but these low-fidelity thumbnails only give a very general picture. Some work has explored interfaces for improving the browsing experience of video tutorials (e.g. [6]). Finding ways to more closely link this browsing experience to the software task at hand would be valuable.

Future work should also consider how videos can integrate with other data sources such as text and images, all of which are useful sources for sensemaking. Some websites provide limited interaction between tutorial steps and corresponding sections in videos (e.g. lynda.com), but there may be ways to deepen the connection between them, such as automatically matching up mentions of tools or techniques across the text, images, and video.

Text is quickly becoming a way of the past when it comes to learning and communicating visual information. As technology continues to improve, higher-fidelity media such as video will become the preferred types of consumed and created media. Videos (especially live-streams) are easier to make than webpages, and short clips are more entertaining to watch [1]. In order to keep up with this rising trend, the sensemaking community must work to understand and improve the ways people navigate and experience online videos, not just for software, but across all domains.

References

1. Saeideh Bakhshi, David A. Shamma, Lyndon Kennedy, Yale Song, Paloma de Juan, and Joseph "Jofish" Kaye. 2016. Fast, Cheap, and Good: Why Animated GIFs Engage Us. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, ACM Press, 575–586. http://doi.org/10.1145/2858036.2858532
2. Pei-Yu Chi, Sally Ahn, Amanda Ren, Mira Dontcheva, Wilmot Li, and Björn Hartmann. 2012. MixT: Automatic generation of step-by-step mixed media tutorials. *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*, ACM Press, 93. http://doi.org/10.1145/2380116.2380130
3. Adam Fourney and Michael Terry. 2014. Mining Online Software Tutorials: Challenges and Open Problems. *Proceedings of the extended abstracts of the 32nd annual ACM conference on Human factors in computing systems - CHI EA '14*, ACM Press, 653–664. http://doi.org/10.1145/2559206.2578862
4. C. Ailie Fraser, Mira Dontcheva, Holger Winnemoeller, Sheryl Ehrlich, and Scott R. Klemmer. 2016. DiscoverySpace: Suggesting Actions in Complex Software. *DIS '16: Proceedings of the 2016 conference on Designing Interactive Systems*.
5. Caitlin Kelleher and Randy Pausch. 2005. Stencils-Based Tutorials: Design and Evaluation. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '05*, ACM Press, 541. http://doi.org/10.1145/1054972.1055047
6. Juho Kim and Juho. 2013. Toolscape: enhancing the learning experience of how-to videos. *CHI '13 Extended Abstracts on Human Factors in Computing Systems on - CHI EA '13*, ACM Press, 2707. http://doi.org/10.1145/2468356.2479497
7. Ben Lafreniere, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2014. Investigating the feasibility of extracting tool demonstrations from in-situ video content. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, ACM Press, 4007–4016. http://doi.org/10.1145/2556288.2557142
8. Benjamin Lafreniere, Tovi Grossman, and George Fitzmaurice. 2013. Community enhanced tutorials: Improving tutorials with multiple demonstrations. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, ACM Press, 1779. http://doi.org/10.1145/2470654.2466235
9. Gierad Laput, Eytan Adar, Mira Dontcheva, and Wilmot Li. 2012. Tutorial-based interfaces for cloud-enabled applications. *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*, ACM Press, 113. http://doi.org/10.1145/2380116.2380132
10. Justin Matejka, Wei Li, Tovi Grossman, and George Fitzmaurice. 2009. CommunityCommands: Command Recommendations for Software Applications. *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*, ACM Press, 193. http://doi.org/10.1145/1622176.1622214
11. Suporn Pongnumkul, Mira Dontcheva, Wilmot Li, et al. 2011. Pause-and-Play: Automatically Linking Screencast Video Tutorials with Applications. *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, ACM Press, 135. http://doi.org/10.1145/2047196.2047213
12. Ba Tu Truong and Svetha Venkatesh. 2007. Video abstraction: A systematic review and classification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 3, 1: 3. http://doi.org/10.1145/1198302.1198305