Investigating the Feasibility of Extracting Tool Demonstrations from In-Situ Video Content

Ben Lafreniere[*†], **Tovi Grossman**[*], **Justin Matejka**[*], **George Fitzmaurice**[*]

[*]Autodesk Research, Toronto, Ontario, Canada
{*tovi.grossman,justin.matejka,george.fitzmaurice*}@autodesk.com

[†]University of Waterloo, Ontario, Canada
bjlafren@cs.uwaterloo.ca

ABSTRACT

Short video demonstrations are effective resources for helping users to learn tools in feature-rich software. However manually creating demonstrations for the hundreds (or thousands) of individual features in these programs would be impractical. In this paper, we investigate the potential for identifying good tool demonstrations from within screen recordings of users performing real-world tasks. Using an instrumented image-editing application, we collected workflow video content and log data from actual end users. We then developed a heuristic for identifying demonstration clips, and had the quality of a sample set of clips evaluated by both domain experts and end users. This multi-step approach allowed us to characterize the quality of "naturally occurring" tool demonstrations, and to derive a list of good and bad features of these videos. Finally, we conducted an initial investigation into using machine learning techniques to distinguish between good and bad demonstrations.

Author Keywords

Help; learning; feature-rich software; video tooltips; toolclips; in-situ usage data.

ACM Classification Keywords

H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

Software applications for content design and creation can be difficult to learn and use [4, 14, 23]. One particular aspect of these programs that makes them difficult for users is their feature richness [21, 26]. These programs typically contain hundreds or even thousands of commands, each with their own particular usage dynamics, making it difficult to *understand* [14] how to use individual tools.

While many learning techniques have been proposed, recent work has found animated demonstrations to be quite useful in the context of graphical user interfaces [6, 15, 22]. In particular, ToolClips—short, 10–25 sec. video clips that demonstrate how to use a command or tool—have been shown to be a promising technique for communicating how to use tools in feature rich software [15].

Unfortunately video content is time consuming to author. As a result, it would be impractical for software vendors to provide video-based demonstrations for the entire set of a software's features. For example, Autodesk's AutoCAD includes ToolClips, but only for 35 of its 1000+ commands.

In contrast, free screen capture systems are making it easier for members of the public user community to record *in-situ* video content, while technologies such as Tutorial Builder [32], Pause-and-Play [30], Chronicle [16], and Waken [3] enable the tagging of workflows with command usage meta-data. This raises an interesting possibility: can short segments from collections of workflow videos be repurposed as contextual, tool-based demonstrations? Our key insight is that meta-data associated with videos (e.g. tool usage logs, or changes to canvas content), could be used to infer segments that could serve as good demonstration videos. This would allow software systems to provide an exhaustive collection of tool-based demonstration videos, without requiring the time-consuming and costly effort of manually creating demonstrations.

In this paper, we utilize a multi-stage methodology to answer the fundamental questions surrounding the feasibility of such an approach. Using an instrumented version of Pixlr, an online image-editing application, we collect workflow video content from end users, marked up with usage meta-data. We then extract candidate tool-based demonstrations, and have the quality of these videos assessed by both domain experts and end-users. This allows us to characterize the quality of "naturally occurring" tool use examples within in-situ video content, and derive a list of good and bad features for such videos. Finally, we investigate the potential of automatically distinguishing good and bad clips using machine learning techniques.

Our findings indicate that there is wide variation in the quality of clips extracted from in-situ video content, but high quality demonstration videos do occur. We also found that users evaluate the quality of such video clips based on a mix of technical aspects of the demonstration that can be detected programmatically, and subjective aspects, such as the nature of the image being edited. Finally, machine learning shows promise as a "first pass" method of filtering

CHI 2014, April 26 - May 01 2014, Toronto, ON, Canada
Copyright is held by the authors. Publication rights licensed to ACM.
ACM 978-1-4503-2473-1/14/04…$15.00.
http://dx.doi.org/10.1145/2556288.2557142

clips, but may need to be supplemented with collaborative filtering or other techniques to create an effective contextual help system. We close by discussing design implications and future directions.

RELATED WORK

Software Learnability

Early HCI research recognized the challenge of providing effective help systems for software applications [4], and established the benefits of minimalist and task-centric help resources [5] and user-centered designs for help systems [20, 31]. Since the conception of on-line help, there have been explorations into many types of learning aids, such as interactive tutorials, contextual assistance [1, 10] and video demonstrations [2, 28, 29, 30]. Below we review the systems most relevant to our own work.

Video Demonstrations

Software video demonstrations were proposed in early work by Palmiter, Elkerton [28, 29], and Harrison [17], but it was unclear if this new help modality would actually be more effective than static help methods [13, 18].

More recently, studies have better identified when animated assistance can be beneficial. A study of ToolClips, short (10–25 sec.) in-context video demonstrations of tools, found that they significantly improve task-completion rates compared to traditional static online help [15]. ToolClips are particularly useful as they are accessed in the context of the software application, making them available on-demand. Similarly, the MixT multimedia tutorial system showed that short video demonstrations are useful for demonstrating individual tutorial steps involving dynamic actions [6]. Additional techniques have been developed to address some of the shortcomings of video-based assistance, such as Pause-and-Play [30], which automatically synchronizes playback of a video to a user's workflow.

A limitation of these techniques is that they require the video content to be explicitly authored or recorded for the purpose of the demonstration. In contrast, we explore the feasibility of generating such content automatically from in-situ workflow recordings.

Workflow Capture

A number of efforts have looked at the possibility of capturing users' workflows. Automatic Photo Manipulation Tutorials [13] and the MixT system [6] allow users to perform a workflow, and have that workflow converted into a tutorial. MixT produces multi-media tutorials that include short video clips of individual steps. However, in both cases, users need to explicitly choose to author a tutorial.

In contrast, systems like MeshFlow [8] and Chronicle [16] are used to continuously record a user's workflow *in-situ*, capturing meta-data, and in the case of Chronicle, a video screen capture as well. The authors of these systems envision that eventually such recording mechanisms could be running at all times, making such workflow histories available for any document. If this were to become a reality, there would be a potential of creating immense libraries of software workflow videos. In this paper, we look at how such libraries could be utilized as a source for video demonstrations.

Automatic Generation of Help Video Collections

Due to the time associated with authoring video, researchers have explored a number of strategies for expanding databases of video demonstration content.

The Ambient Help system [24] dynamically loads and displays potentially helpful video content. The system contains a large database of existing YouTube and professional tutorial videos, and attempts to retrieve sections of videos related to the user's current task. The Waken [3] system is able to recognize UI components and activities from screen capture videos with no prior knowledge of an application. Work on "learnersourcing" asks short questions of viewers of a video, and applies machine learning to the responses to automatically label the activities being performed in the video [19]. Community Enhanced Tutorials provide videos of individual tutorial steps, and the database of video demonstration increases by capturing the workflows of any user that completes the online tutorial [22].

The work above presents a number of plausible routes through which libraries of software workflow videos with meta-data might be created. However, it is unclear if segments of video content created during in-situ workflows could be reliably used within help components such as ToolClips, to assist end-users with learning the software.

METHODOLOGY

The goals of this paper are twofold: to characterize the quality of naturally occurring demonstrations in real-world usage data, and to produce a set of guidelines for what makes a good short video demonstration of a tool. To achieve these goals, we followed a five stage methodology. In this section we give an overview of each of the stages, which will be followed by a detailed section for each stage.

Data Collection: We started by collecting a corpus of real-world usage data from remote participants recruited online. Participants installed a custom screen recording utility and used an instrumented version of the Pixlr.com image editor to work on image editing tasks of their choice, and then submitted at least one hour of the workflow data to us.

Clip Segmentation: Next, we developed a method for selecting short video clips showing the use of individual commands. In total, we sampled a set of 277 clips for six commands, covering a range of different types of tools.

Internal Assessment: Three internal raters independently analyzed and rated the quality of the entire set of individual sample clips. This provided us with a sense of the distribution of quality that the sampling process would produce, as well as insights on the features and criteria that separate good and bad clips.

User Validation: To validate our internal ratings, 12 participants with experience using image editing software were recruited to rate a sampling of the clips. Participants were asked to think aloud as they rated the clips, to provide additional insights into their criteria for judging the clips.

Automatic Identification: Finally, we report on our initial efforts to use machine learning techniques as a method of automatically identifying good examples of tool demonstration videos from in-situ usage data.

The primary contributions of this paper are based on our methods and findings from the first four stages, which provide new data on the spectrum of quality of automatically extracted video demonstrations, and report on the features which high quality examples possess. The final stage serves as a secondary contribution, where we report on an initial attempt to automatically classify demonstration videos as good or bad.

DATA COLLECTION

In this section we describe our method for collecting in-situ workflow data. Our methods are guided by previous research that has captured usage logs of web browsing [27], software usage [21], and low-level input activities [9, 12].

In this work we do not consider recording or use of audio, because users would not typically narrate their workflows during in-situ use.

Task and Participants

To collect a corpus of real-world usage data, we recruited freelance graphic designers through online postings on oDesk and Reddit. Our ad asked participants to use an instrumented version of the web-based Pixlr image editor (www.pixlr.com/editor) for at least 1 hour, and submit the resulting usage data to us.

In a pre-screening process, we asked users to give us an idea of the types of jobs they would work on during the session. The intention here was to confirm that participants understood that they would need to choose their own work to do during the study. We did not screen users based on the tasks they proposed to perform during the study.

We hired a total of 21 participants, 17 from oDesk and 4 from Reddit. Contractors on oDesk were paid an average of $23.92 and participants recruited on Reddit were given a $25 Amazon.com gift card.

Screen Recording and Logging

Participants were asked to install a customized screen recording and logging utility on their computers. In addition to recording the screen, the utility hosted a local server on the participant's computer that received and recorded log events sent to it by the instrumented Pixlr editor (Figure 1).

Our goal was to be as exhaustive as possible with respect to the metadata which we collected. This would allow us to later investigate which, if any, features of the workflow would correlate to the quality of the demonstrations. The instrumented Pixlr editor gathered the following additional data, synchronized with the screen recording:

- Mouse movement and click logs
- Tool invocations (including start and end timestamps)
- Dialog open and close events
- All changes to the undo stack
- Document Snapshots after each change to the undo stack
- Changes to the selection state, and the selection region
- Changes to settings for the current tool
- Changes to the color palette

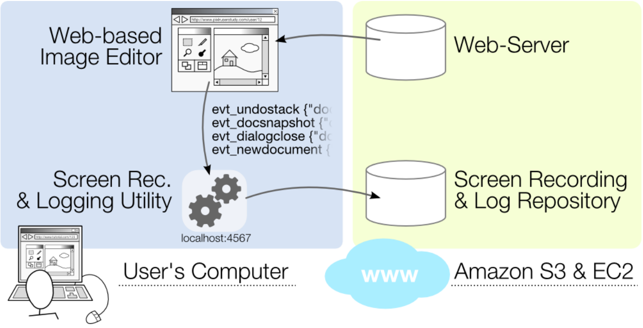

Figure 1. Architecture for remotely gathering synchronized screen recording and log data.

Collected Data

Across all participants, we gathered 25 hours, 23 minutes of usage data at an average cost of $22.80 per hour. Our data set included 11,553 tool invocations of 93 unique tools. The distribution of tool invocations was very uneven and drops off exponentially when considering commands ordered by the number of invocations (Figure 2). For example, the ten most frequent commands make up 75% of command invocations. This exponential distribution of command invocations is consistent with previous research characterizing the frequency of tool use in feature-rich software [21].

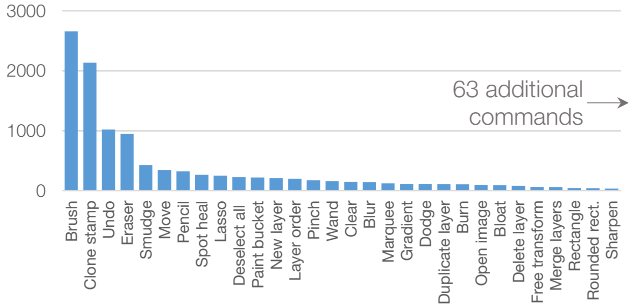

Figure 2. Invocation counts for the 30 most used tools.

CLIP SEGMENTATION

The next stage in the methodology was to create a set of candidate clips to serve as tool demonstration videos. This involved selecting a set of tools, and segmenting out clips of those tools.

Tool Selection

To focus our efforts we choose a set of six tools that were all invoked at least 20 times within the collected usage data.

The six tools we selected are shown in Table 1. These tools were chosen to get a perspective on a variety of different tool types, including direct manipulation tools which are primarily applied using brush strokes (DM-Brush), direct manipulation tools that are typically applied in one invocation that affects a large area (DM-Area), and dialog commands such as adjustments and filters (Dialog).

Tool Name	Type	Total Clips	Sample
Brush	DM-Brush	1311	60
Clone Stamp	DM-Brush	855	59
Fill	DM-Area	138	60
Gradient	DM-Area	80	46
Hue Saturation	Dialog	33	30
Levels	Dialog	22	22

Table 1. Summary of clips selected by our selection criteria.

Clip Segmentation

Next, we developed a method for segmenting out clips that demonstrate each of the six tools that we selected.

A naive method would be to simply cut the video up so that each clip shows a single tool invocation, with a short time before and after the invocation to provide some context on the operation being performed (Figure 3a). In a pilot implementation using this selection method, we found that direct manipulation tools were difficult to follow because users typically make a number of strokes in quick succession. This made for a large number of short and ineffective demonstration videos. As well, these clips seldom show the user selecting the tool from the toolbox, or making setting adjustments (e.g. setting the brush size before a stroke).

An alternative approach is to aggregate consecutive invocations of the same tool into a single clip, and adjust segmentation boundaries to include preceding operations such as selecting the tool [6] (Figure 3b). However, we found that aggregating all successive invocations of a tool resulted in clips that could be quite long. This makes them less appropriate for contextual videos, which should be in the range of 10–25 seconds [15].

Based on the above considerations, we developed a segmentation algorithm that balances clip time and includes tool selection and settings adjustments (Figure 3c).

For a given tool, we first find consecutive sequences where the tool is invoked. For each such sequence, we start by adding the first invocation to an empty segment *S*. Successive invocations of the tool are then appended to *S* if their starting timestamps are less than 5 seconds from start of the first invocation in *S* (this threshold was tuned through testing). When we reach an invocation that does not meet this criterion, segment *S* is reported and the current invocation is used to start a new segment. This process continues until the consecutive sequence of invocations is exhausted.

In addition, for modal tools, if the tool was selected less than 10 seconds before the first invocation in a segment, the segment boundary is adjusted to show the user selecting the tool. This 10 sec limit on pre-command time was chosen based on our log data, which showed that the median time between tool selection and first invocation is less than 10 seconds for all modal tools in Pixlr. Finally, a 2 second padding is added to the start and end of all sequences.

Though our algorithm is designed to keep demonstrations short it intentionally does not impose a hard limit on the duration of a segment, because our log data indicates that that the duration of an invocation for different tools can vary by orders of magnitude.

This algorithm generated a total of 2439 clips across our six selected tools. For each tool, we then randomly chose up to 30 clips with selection, and up to 30 clips without selection to be included in our quality assessments. A summary of the clip selection and sampling is provided in Table 1.

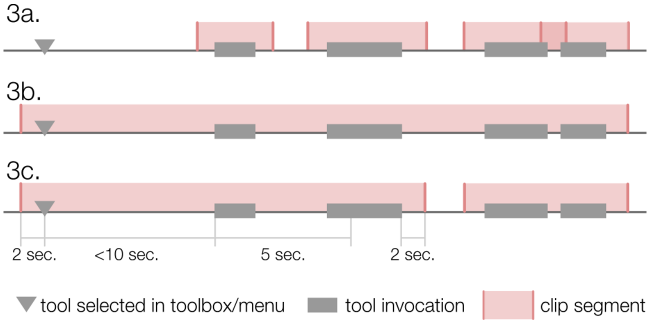

Figure 3. Three clip segmentation approaches. (a) Segment each invocation, (b) Include tool selection and merge contiguous invocations, (c) Our segmentation algorithm.

INTERNAL ASSESMENT OF CLIP QUALITY

While the clips generated by our selection algorithm may be of appropriate length, this alone is no guarantee that the content of the clips are appropriate for learning. In fact, we can imagine numerous reasons why clips might be of poor quality, including the author working too quickly, or making mistakes as they learn to use the software.

Method

To characterize the quality of naturally occurring tool use examples, three raters independently analyzed and rated the quality of each individual sample clip. All three of the raters were authors of this paper, and all had research experience in video-based help systems.

Each rater was instructed to view the 277 clips sampled in the previous section, and rate the quality of each on a seven-point scale for the question "*How good would this clip be for demonstrating how the [name] tool works to a novice user?*" (1=Very Poor, 7=Very Good). Each rater was also asked to record free-form notes on features of clips that influenced their ratings.

The order in which raters viewed the tools was fixed for all raters, and the order of clips within each group was randomized. It took each rater approximately 1.5 hours to rate the set of 277 clips. Once all clips were rated, the raters met to discuss clips with a large spread between the maximum and minimum ratings. Clips with a spread of 5, 4 and a selection of clips with spread 3 were discussed. In some cases, raters modified their ratings based on the discussion.

Results

For the purposes of calculating inter-rater reliability, we consider a rating of 5 or higher to indicate that a clip is "good", and compute Fleiss' Kappa on raters' assessments of clips as good or not. Before the meeting to discuss the clips, the inter-rater reliability was 0.49, and it increased to 0.57 after the meeting. Both of these values indicate moderate agreement between the raters.

We assigned each clip an overall score based on the median of its 3 ratings. The average score across all clips was 3.3 (SD 1.7). Figure 4 shows the distribution of clip scores.

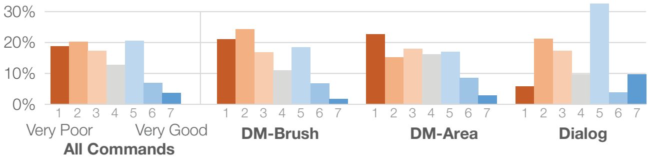

Figure 4. Distribution of median ratings for all six tools (left), and for each of the tool types (right).

We highlight two findings. First, while only 3.6% of the clips had a score of 7, almost a third of the clips (31%) had a score of 5 or greater. This suggests that good tool demonstrations do occur within in-situ workflow recordings.

Second, more than half (56.3%) of clips had a score of less than 4, which suggests that poor quality clips also occur in in-situ workflow recordings. This suggests that appropriate clip filtering mechanisms will also be required to identify good clips; an effective clip selection algorithm is not sufficient on its own.

We also examined the distribution of scores by tool type (Figure 4). Though the data is noisy, there appears to be a similarity between the distributions of scores for the two direct manipulation tool classes, with clips for the Dialog tools receiving relatively higher scores.

In addition to assessing the quality of clips, this phase of the methodology was used to obtain initial insights into the features that distinguish good and poor demonstrations. The first author of the paper used an open coding approach to analyze the list of features noted by the raters. The most commonly mentioned features were:

- Extent of change to the image in the clip
- How well the clip demonstrates the dynamics of the tool
- The pace of the clip, or how smoothly actions are performed
- Whether the clip shows tool settings being adjusted
- The presence of unrelated or distracting actions in the clip
- Qualities of the image being edited

In the next section we derive a more nuanced look at features impacting clip quality, based on the results of a think aloud protocol with users of image editing software.

USER VALIDATION

The results of our internal assessment indicate that:

1. *Good tool demonstration clips exist within in-situ workflow recordings.*
2. *The quality of naturally occurring tool demonstrations ranges widely.*

These findings represent the judgment of domain experts in the area of video-based help systems. However, a caveat to these results is that they are also the opinions of the authors of this paper, who are familiar with the goals of the project. Thus, we felt it necessary to obtain external validation for these findings from an unbiased group.

In this section, we report on a laboratory study we conducted with users of image editing software to validate the internal ratings, and to evaluate the quality of the clips as a help resource.

Method

We recruited 12 participants (5 male, 7 female), with a mean age of 27.5 (SD 7.2, min 20, max 46) via email and mailing list postings. Participants were screened to ensure they had some image editing experience by asking them to describe their experience using image editing software, and their familiarity with image editing concepts (e.g. selections, filters, and layers). Participants were given a $25 gift certificate for an online retailer for participating.

For each of the six tools from the previous section, we selected nine clips, three each with *Good*, *Neutral*, and *Poor* median ratings from the internal assessment. For the *Good* and *Poor* rating classes, we selected the top and bottom rated three clips respectively. The *Good* clips had an average score of 6.6 (SD 0.5) and the *Bad* clips had an average score of 1.4 (SD 1.0). For the *Neutral* clips, we selected the three clips with median ratings closest to 4 for each tool. These clips had an average score of 4.1 (SD 0.2).

For each of the six commands, the study included a *viewing stage*, in which participants viewed the nine clips for the command, followed by a *rating stage* where they viewed each clip again and rated their agreement to the statement *"This is a good video clip for demonstrating the [name] tool."* (1=Strongly Disagree, 7=Strongly Agree).

The order of each command block was counterbalanced across participants in a balanced Latin square design. Within each block, clip order was randomized but was consistent across the viewing and rating stage for each participant. We asked participants to "think aloud" about their rationale for rating each clip as they were rating. We intentionally did not specify to participants where the video clips came from, to avoid introducing a bias that might impact their ratings.

An experimenter was present to take notes, and the sessions were video recorded so they could be later reviewed. Occasionally, the experimenter would ask the participant for clarifications of their rationales, or remind the participant to state their rationale aloud while rating.

A qualitative coding scheme was developed using open coding to record the attributes of the video clips that were mentioned by participants. The experimenter additionally assigned each code an affect (positive or negative) depending on whether the participant cited an attribute as contributing positively or negatively to their rating of a clip.

Results – Ratings

The ratings assigned by participants, broken down by the category of each clip based on the expert ratings can be seen in Figure 5.

Participants assigned *Good* clips an average rating of 5.6 (SD 1.4), *Neutral* clips 4.9 (SD 1.7), and *Poor* clips 2.8 (SD 1.8). A one-way repeated measure ANOVA found a significant main effect of expert-rating category on average ratings ($F_{2,22}$=101.4, p < .01, η^2=0.72), and post-hoc analysis using paired t-tests with Bonferroni correction revealed significant differences between all pairs of expert-rating categories (p < .01 in all cases).

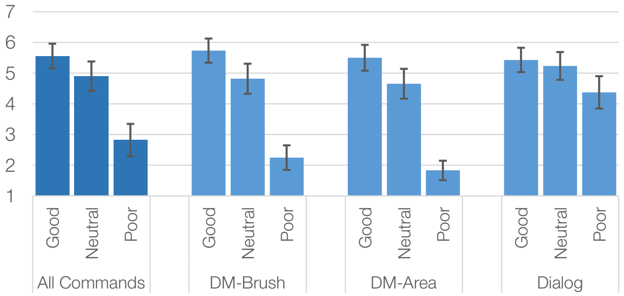

Figure 5. Average clip ratings for each of the expert-rating categories. Error bars show standard error.

These results provide evidence to validate that demonstrations of a wide range of different levels of quality occur in in-situ workflow recordings.

Examining the median ratings of clips by participants, we found that 3 clips (5.6% of those rated) were assigned a median rating of 7, and 15 clips (27.8%) were assigned a median rating of 6 or higher. This provides encouraging evidence that good quality clips do exist in in-situ workflow data, as these clips consistently received high ratings from a majority of the 12 study participants.

We also found evidence to suggest that tool type impacts the distribution of quality of demonstrations. As can be seen in Figure 5, the ratings for the *Poor* clips for Dialog tools is significantly higher than for the two DM- tool types (Welch's two-sample t-test, $t_{18.73}$ = 8.55, p < .01, Cohen's d = 3.22). This is consistent with our observations from the internal assessment, and suggests that it may be easier to find good clips for dialog tools.

In summary, the quantitative results of our study validate our finding that tool demonstrations of a range of distinct levels of quality exist in in-situ workflows, including good demonstrations. In the next section we examine the attributes that participants used when evaluating demonstrations.

Results – Think Aloud Rationales

A summary of attributes cited by participants while thinking aloud is shown in Figure 6. The majority of attributes were mentioned with a particular affect (positive or negative), however some attributes were mentioned as both positives and negatives. These attributes are shown with both positive and negative bars in Figure 6.

In this section, we discuss some of the common themes that emerged from participants' stated rationales, with an eye toward how these features could be automatically detected.

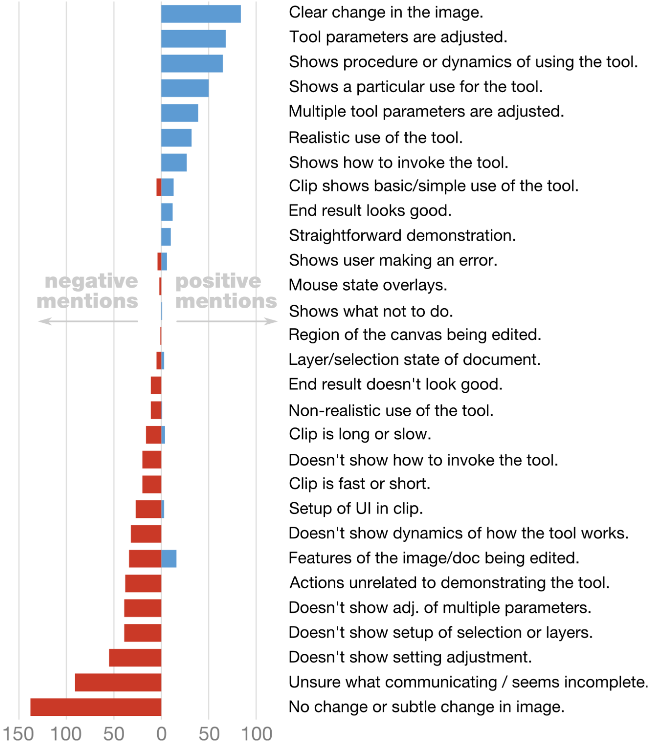

Figure 6. Participants' rationales for their clip ratings. Blue bars on the right indicate attributes cited as positive. Red bars on the left indicate attributes cited as negative.

Change in the Image

The most often cited positive rationale was that the clip showed a clear change in the image. Likewise, the most often cited negative rationale was when the clip showed no change in the document, or only a subtle change. It is clear why this is an important feature: if no changes are visible, a user couldn't be expected to understand how the tool works.

We examined clips cited with these attributes to better understand the cause of this rating, and found a number of reasons why clips drawn from in-situ workflows might not show a large change in a document. First, users sometime make mistakes, such as performing an operation with the wrong layer selected, so a change made to the document is not visible. Second, users often make several experimental applications of a tool before settling on a final result, and some of these experimental applications have less than the intended effect. Third, this may be intentional; it's not always desirable to make a bold change to an image.

Fortunately, changes in an image are easy to detect algorithmically. We explicitly logged snapshots of users' documents before and after each operation, but it's conceivable that this data could be extracted from screen recording videos as well.

Unclear Intention

The second most common negative attribute was that the clip appeared to be incomplete, or the participant was not sure what the clip was trying to communicate.

We observed two frequent causes for this confusion. The first was when there was only a subtle or undetectable change shown in the clip, as discussed above. Second, some clips included actions unrelated to the tool being demonstrated, during the padding time at the start or end of the clip. For example, in one clip an author applies the Gradient tool to an image. After the tool is applied, but before the clip ends, the author is shown to adjust the Gradient tool's opacity setting and undo the Gradient they just created, presumably preparing to apply the Gradient tool again.

For users who are new to a tool, these extraneous actions can lead to an incorrect mental model of how a tool works. One participant misinterpreted the Gradient clip discussed above as demonstrating that a gradient can be adjusted after it's been applied, which is not true in Pixlr:

> *So this one, I think goes to show that you can also alter the opacity after you've applied the gradient. So, yeah, a good kind of indication of what you can do, that it's more versatile than just applying [the] gradient and then you're finished with it.* (P10)

These observations suggest that we should refine our clip selection algorithm to crop out additional actions from the start and end of the clips. Another option would be to give the viewer the ability to access additional video around a clip on demand (e.g. a "Play next 30 seconds" button.)

Settings Adjustments

Participants frequently mentioned that showing adjustment of a tool's settings was positive, and conversely cited not showing setting adjustments as a negative.

This provides some validation of our clip selection heuristic for direct manipulation tools, which was designed to include the time between a tool's selection and its invocation, which is often spent adjusting the tool settings (e.g. selecting a brush to use.)

For Dialog commands, participants particularly cared about whether the *majority* of available settings were adjusted in a tool demonstration. For example, the Hue Saturation command in Pixlr includes three parameters (Hue, Saturation, and Lightness), and a Colorize option. Participants frequently cited showing the effect of multiple or all of these settings as a positive attribute.

Usage Scenarios

In some clips, the user had set up layers or selections before applying a command, but due to our segmentation heuristic, the setup operations were not shown in the clip. Some participants immediately recognized what was going on in these clips, and commented that they were useful to demonstrate the tool in a particular scenario. Other participants did not realize that setup operations had occurred, or cited these clips as problematic because novice users might not understand how to recreate the context shown in the clip.

This suggests that it may be beneficial to choose a variety of clips showing a tool used in different contexts. Some participants explicitly expressed a desire for variety:

> *I'd like to see an example where you have an extremely bright image, and you do the opposite, you darken the image. Because I think most of [the clips], they're darker and you lighten them. I'd like to see the opposite.* (P3)

From a design perspective, this motivates the idea of providing a range of short demonstrations showing tools used in different contexts, or geared toward users with particular knowledge or expertise.

Difficult to Detect Features

While the features mentioned up to this point can be detected from video or log data, participants also mentioned a number of features of clips that are difficult to detect, or which have a subjective component to them.

Some participants expressed a preference for clips that showed a basic use of a tool, as in the following comment for a Levels clip that showed the tool on a document consisting of one solid color:

> *This one I really liked. I put [it] as my favorite [rates as 7/7], because it's very simple. It shows how [the tool] was selected, and shows exactly what's going on when you change each of the settings.* (P11)

Other participants commented that this clip was too simple. On other clips, participants expressed a preference for demonstrations on realistic images, as in this comment:

> *I like this video because it shows how you actually use this. [...] It's something that people would actually use [the Clone Stamp tool for] in real life.* (P12)

The aesthetic quality of the result in a demonstration was also mentioned by participants. For example, one clip clearly shows how the Clone tool operates, but several participants mentioned that the end result didn't look good:

> *It does kind of screw up the picture as it went on with the Clone brush… Which does happen when you use Clone brush a little too much.* (P2)

Subjective or aesthetic features pose a challenge because different users will react to them in different ways, and they are difficult to detect algorithmically. However, these features were in the minority of those mentioned by participants. As well, it is possible that demonstrations of tools with imperfect results can still effectively communicate the dynamics of how a tool works. For example, P2 assigned the Clone tool clip discussed above a 6/7, suggesting that, for him, the imperfect end result did not significantly detract from the demonstration's value.

AUTOMATICALLY IDENTIFYING GOOD CLIPS

In the previous section we discussed some of the most commonly cited positive and negative features of clips. Next, we conduct an experiment to see if low-level features can be used to automatically distinguish good clips.

We processed the meta-data associated with 274 clips rated in our initial evaluation to add a set of automatically extractable features (Table 2). Many of these features were suggested by our qualitative findings, including features that quantify changes in the image, changes in settings, the pace and duration of the clip, and the presence of extraneous command invocations in the clip.

Image Features	% of Pixels Changed, Start Entropy, End Entropy, Entropy Change, Pixels in Viewport, Median Zoom Level, % of Image Selected, Entropy of Selection Mask.
Mouse Features	Idle Time, Mean Velocity, Mean Acceleration, Mouse-Down time, # of Clicks, Mouse Bound Area
Behavior Features	Command ID, Timestamp, Total Clip Duration, % Pre-Command Time, % Command Time, # of Command Invocations, # of Other Commands, # of Settings Changed

Table 2. Features used to train the SVM.

We assigned all clips with a median rating of 5 or higher from the initial evaluation a label as *Good*, and the remaining clips as *Poor*. We then trained a Support Vector Machine (SVM) classifier [7] to distinguish between *Good* and *Poor* clips. SVMs are a widely used supervised learning method for classification, supported by many machine learning libraries (we used the scikit-learn Python library).

The *Cost* and *Gamma* parameters of the SVM classifier were tuned using grid search, with the $F_{0.5}$ score as an objective function. This objective function was chosen to bias the classifier toward higher precision (fewer false positives) at the cost of lower recall (more false negatives), with the rationale that it is better to reject some good clips than to classify poor clips as good.

True +	37	*Accuracy*	77.4%
True −	175	*Precision*	0.71
False +	15	*Recall*	0.44
False −	47	*F_1 Score*	0.54

Table 3. Results of SVM classification of tool clips.

We evaluated our classifier using 10-fold cross validation. The results are shown in Table 3. The overall accuracy of our approach was 77.4%, with a precision of 0.71. As expected from our decision to bias toward higher precision, the recall (0.44) and F_1 score (0.54) were lower. In interpreting the results we note that only 31% of the original clips were *Good*, so a randomly selected sample of clips would have an expected precision of 0.31.

The average score for clips classified as *Good* (4.67, SD 1.7) was significantly higher than the average score for clips classified as *Poor* (2.98, SD 1.6) (Welch's t-test, $t_{74.91} = 6.68$, $p < .01$, Cohen's *d*=1.05). This indicates that the classifier was generally successful in separating good and bad clips.

These results provide promising evidence that lower-level features of clips extracted from in-situ workflow data can be used to identify good demonstrations. However, the low recall and F_1 score indicate room for improvement.

It may be that additional training data, or more sophisticated machine learning techniques, could allow us to improve on these results. However, there is likely an upper limit on the accuracy that can be achieved by automatic methods; recall that we found only moderate agreement in our internal ratings, and subjective features played a role in evaluations of clip quality. Even so, the accuracies reported above make this approach suitable as a "first-pass" method of indicating potentially good clips in a large set produced by a selection heuristic. These could then be refined using other techniques, such as collaborative filtering, as we discuss in the next section.

DESIGN IMPLICATIONS

In this section, we discuss two main implications of our findings. First, we propose design guidelines for demonstration videos. Second, we present a design sketch of an interface for delivering automatically extracted video clips.

Design Guidelines for Demonstration videos

The input we elicited from study participants and domain experts provide us with a clear picture of the attributes of a good tool-based demonstration video. Despite the recent popularity of video-based assistance, we are unaware of any such existing set of attributes. Our insights can be distilled into the following five guidelines:

1. Be short and concise, 15–25 sec.
2. Show the tool making a clear change to the document.
3. Demonstrate how the various parameters or settings of the tool affect the dynamics of how the tool works.
4. Demonstrate an example that is representative of how the tool would typically be used.
5. Avoid including extraneous actions not directly related to the tool being demonstrated.

This set of guidelines for individual clips contains some tensions. For example, the Brush tool includes eight settings dictating the dimensions of the brush and the dynamics of how the brush affects the document; an example showing all of these would not be concise. As such, we can include a set of additional guidelines relating to multiple examples:

1. Provide multiple clips to demonstrate the range of settings for a given tool.
2. Provide clips tailored to a range of users with different backgrounds and knowledge of the system.

In the next section, we provide a mockup of what the interface for such a system might look like.

Delivery Interface

The results of our machine learning experiment suggest that the low-level features of demonstration clips could be used to filter out obviously poor clips. This smaller set of clips could be presented to the community of users with a voting mechanism for flagging good or poor clips (e.g. Figure 7). This would allow the community to refine the set of clips over time, and at the same time provide additional training data for machine learning algorithms.

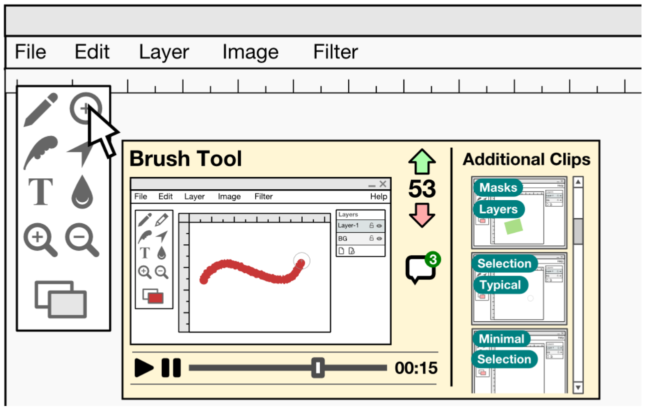

Figure 7. Design sketch showing how tool clips could be collaboratively filtered, and how multiple tool clips could be presented to users.

This approach would also allow the system to tailor the clips it presents to individual users, using techniques similar to those explored for command recommender systems [25]. The voting behavior of users could be combined with metrics on how long the voter has used the software, or the set of tools they use most frequently, to create models of which clips are good or bad for individual users. For example, if a user that has never used a tool before votes a clip up, this may indicate the clip is good for novice users of the tool.

This connects with another theme that emerged from our user evaluation, that it may be useful to make multiple clips available for each tool. These could show a tool used in different contexts, or for users with varying levels of expertise. Figure 7 demonstrates this idea as well, with an area where users can view additional clips for a given tool. Clips could be tagged with badges indicating fundamental skills (e.g. knowledge of Selections, Layers, or Masks) associated with each clip. This would help the user to select clips that fit their personal skill level and knowledge of these fundamental concepts. Similarly, clips could be tagged with the setting parameters that are adjusted in the clip, or grouped to show a complementary set of settings or adjustments that together give a full idea of how the tool operates.

DISCUSSION AND FUTURE WORK

Our initial assessment and validation study have shown that in-situ workflows include tool demonstrations of a range of levels of quality, including demonstrations that are suitable for use as help resources. Our investigation also revealed attributes that impact the quality of demonstrations, and proposed a set of corresponding guidelines for demonstration videos. Finally, we experimented with using machine learning to distinguish good demonstrations. In doing so, we've shown that low-level features extracted from log data can be used to identify potentially good clips from in-situ workflows. These results are encouraging, and we can envision several avenues for extending this work.

First, we could investigate the problem of identifying good demonstrations of higher-level tasks, which may involve use of multiple tools. This is a more challenging problem than automatically identifying tool demonstrations because it requires a model of the tools and procedures involved in higher-level tasks. However, recent work provides some potential sources for this information. Fourney et al. have presented a technique for identifying command-task relationships using web search query logs [11], and work on "learnersourcing" suggests the potential of crowdsourced tagging of activities in video clips [19]. Finally, our own work on Community Enhanced Tutorials has investigated a modified web tutorial design that could gather in-situ workflow recordings of individual tutorial steps [22].

Second, we could consider other sources of video content. In this work, we examined workflow data from in-situ use. It would be interesting to contrast our results with video data gathered from existing libraries of online video tutorials (e.g. on YouTube). These videos provide longer and more extensive instruction than the short demonstrations of tools that we seek to provide. However, it is possible that these videos include a higher occurrence of good quality demonstrations than in-situ workflow recordings, because they are created with the intent to communicate with an audience. They also typically include audio narration, which adds another dimension to the problem of clip segmentation. Some of the issues with segmenting clips with audio have been identified in past work on multimedia tutorials [6], but this a largely unexplored area.

Finally, the approach that we have presented could be generalized or repeated. A first step toward generalizing our results is to consider other feature-rich applications that include concepts such as modal tools (e.g. MS Office). The main challenge to extend our approach to a new application or domain is in identifying the features that indicate where to cut clips, and that impact the quality of demonstrations. It would be interesting to see if similar features to those we identified for image editing apply in other domains as well.

Extending our approach to drastically different domains, such as video lectures for online courses, is another possibility. For example, it would be interesting to investigate whether demonstrations of mathematical operations (e.g. matrix diagonalization) could be automatically identified.

Regarding the repeatability of this work, while we instrumented our test application directly, the log data we gather is standard (e.g. command timings, mouse movement), and could potentially be gathered using publicly-available accessibility APIs.

In conclusion, we feel that the automatic extraction of help resources from in-situ usage data has great potential, and the work presented here opens important opportunities for future efforts in this area.

REFERENCES

1. Ames, A.L. Just what they need, just when they need it: an introduction to embedded assistance. *ACM SIGDOC '01*, (2001), 111–115.
2. Baecker, R. Showing instead of telling. *ACM SIGDOC '02*, (2002), 10–16.
3. Banovic, N., Grossman, T., Matejka, J., and Fitzmaurice, G. Waken: reverse engineering usage information and interface structure from software videos. *ACM UIST '12*, (2012), 83–92.
4. Carroll, J.M. and Rosson, M.B. Paradox of the active user. In *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*. MIT Press, 1987, 80–111.
5. Carroll, J.M. *The Nurnberg funnel: designing minimalist instruction for practical computer skill*. MIT Press, 1990.
6. Chi, P.-Y., Ahn, S., Ren, A., Dontcheva, M., Li, W., and Hartmann, B. MixT: Automatic generation of step-by-step mixed media tutorials. *ACM UIST '12*, (2012), 93–102.
7. Cortes, C. and Vapnik, V. Support-vector networks. *Machine Learning 20*, 3 (1995), 273–297.
8. Denning, J.D., Kerr, W.B., and Pellacini, F. MeshFlow: Interactive visualization of mesh construction sequences. *ACM Trans. Graph. 30*, 4 (2011), 66:1–66:8.
9. Evans, A. and Wobbrock, J. Taming wild behavior: the input observer for obtaining text entry and mouse pointing measures from everyday computer use. *ACM CHI '12*, (2012), 1947–1956.
10. Farkas, D.K. The role of balloon help. *ACM SIGDOC Asterisk J. Comput. Doc. 17*, 2 (1993), 3–19.
11. Fourney, A., Mann, R., and Terry, M. Query-feature graphs: bridging user vocabulary and system functionality. *ACM UIST '11*, (2011), 207–216.
12. Gajos, K., Reinecke, K., and Herrmann, C. Accurate measurements of pointing performance from in situ observations. *ACM CHI '12*, (2012), 3157–3166.
13. Grabler, F., Agrawala, M., Li, W., Dontcheva, M., and Igarashi, T. Generating photo manipulation tutorials by demonstration. *ACM Trans. Graph. 28*, 3 (2009), 66:1–66:9.
14. Grossman, T., Fitzmaurice, G., and Attar, R. A survey of software learnability: metrics, methodologies and guidelines. *ACM CHI '09*, (2009), 649–658.
15. Grossman, T. and Fitzmaurice, G. ToolClips: An investigation of contextual video assistance for functionality understanding. *ACM CHI '10*, (2010), 1515–1524.
16. Grossman, T., Matejka, J., and Fitzmaurice, G. Chronicle: Capture, exploration, and playback of document workflow histories. *ACM UIST '10*, 143–152.
17. Harrison, S.M. A comparison of still, animated, or nonillustrated on-line help with written or spoken instructions in a graphical user interface. *ACM CHI '95*, (1995), 82–89.
18. Kelleher, C. and Pausch, R. Stencils-based tutorials: Design and evaluation. *ACM CHI '05*, (2005), 541–550.
19. Kim, J., Miller, R.C., and Gajos, K.Z. Learnersourcing subgoal labeling to support learning from how-to videos. *ACM CHI EA '13*, (2013), 685–690.
20. Knabe, K. Apple guide: a case study in user-aided design of online help. *ACM CHI '95*, (1995), 286–287.
21. Lafreniere, B., Bunt, A., Whissell, J., Clarke, C.L.A., and Terry, M. Characterizing large-scale use of a direct manipulation application in the wild. *GI '10*, (2010), 11–18.
22. Lafreniere, B., Grossman, T., and Fitzmaurice, G. Community enhanced tutorials: Improving tutorials with multiple demonstrations. *ACM CHI '13*, (2013), 1779–1788.
23. Lazar, J., Jones, A., and Shneiderman, B. Workplace user frustration with computers: an exploratory investigation of the causes and severity. *Behaviour and Information Technology 25*, 3 (2006), 239–251.
24. Matejka, J., Grossman, T., and Fitzmaurice, G. Ambient help. *ACM CHI '11*, (2011), 2751–2760.
25. Matejka, J., Li, W., Grossman, T., and Fitzmaurice, G. CommunityCommands: Command recommendations for software applications. *ACM UIST '09*, 193–202.
26. McGrenere, J. and Moore, G. Are we all in the same "bloat"? *GI '00*, (2000), 187–196.
27. Obendorf, H., Weinreich, H., Herder, E., and Mayer, M. Web page revisitation revisited: implications of a long-term click-stream study of browser usage. *ACM CHI '07*, (2007), 597–606.
28. Palmiter, S., Elkerton, J., and Baggett, P. Animated demonstrations vs written instructions for learning procedural tasks: a preliminary investigation. *Int. J. Man-Mach. Stud. 34*, 5 (1991), 687–701.
29. Palmiter, S. and Elkerton, J. Animated Demonstrations for Learning Procedural Computer-Based Tasks. *Human–Computer Interaction 8*, 3 (1993), 193–216.
30. Pongnumkul, S., Dontcheva, M., Li, W., et al. Pause-and-play: automatically linking screencast video tutorials with applications. *ACM UIST '11*, (2011), 135–144.
31. Sellen, A. and Nicol, A. Building user-centered on-line help. In R.M. Baecker, J. Grudin, W.A.S. Buxton and S. Greenberg, eds., *Human-Computer Interaction*. Morgan Kaufmann Publishers Inc., 1995, 718–723.
32. Adobe Labs Tutorial Builder. 2012. http://labs.adobe.com/technologies/tutorialbuilder/.