mVideoCast: Mobile, real time ROI detection and streaming

Scott Carter, Laurent Denoue, John Adcock
FX Palo Alto Laboratory
3400 Hillview Ave.
Palo Alto, CA 94304 USA
(carter,denoue,adcock)@fxpal.com

Abstract
A variety of applications are emerging to support streaming video from mobile devices. However, many tasks can benefit from streaming specific content rather than the full video feed which may include irrelevant, private, or distracting content. We describe a system that allows users to capture and stream targeted video content captured with a mobile device. The application incorporates a variety of automatic and interactive techniques to identify and segment desired content, allowing the user to publish a more focused video stream.

Keywords
Mobile, multimedia, capture

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation: User interfaces

Submitted to CHI 2011 (Extended Abstracts), May 7–12, 2011. Vancouver, BC, Canada

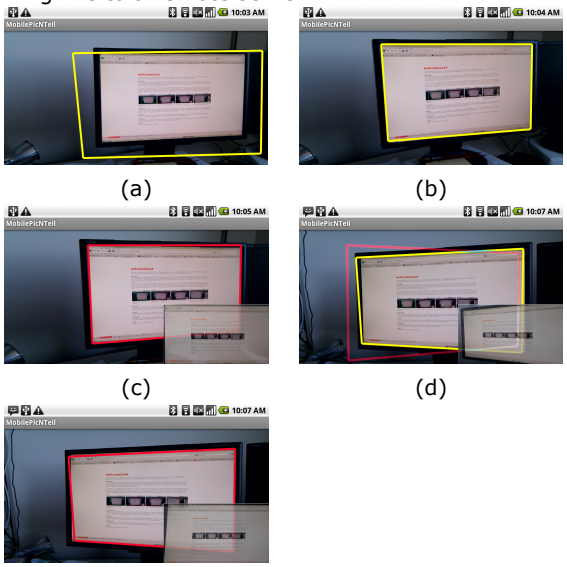
Introduction
As the processing power of mobile devices improves, they are being used for more computationally intensive tasks. Lately, some services have begun to offer live video streaming from mobile devices (e.g., Qik [10]), potentially allowing anyone to stream any event anywhere at anytime. However, as we have seen in the desktop world, unfiltered streaming, while useful, is not appropriate for every task. By filtering streamed content, presenters can better focus the audience's attention, improve bandwidth efficiency, and mitigate privacy concerns. Furthermore, mobile content capture faces challenges not present on desktops – for example the recording device may be off-axis from the desired content and may be handheld and therefore not very stable.

The system we present, mVideoCast, helps filter and correct video captured and streamed from a mobile device (see Figure 1). The application can detect, segment, and stream content shown on screens or boards, faces, or arbitrary regions. This can allow anyone to stream task-specific content without needing to develop hooks into external software (i.e., without installing a screen recorder software in the computer). In this paper we describe the system, algorithms we use to detect regions-of-interest (ROIs) in video frames, and our experiences using early prototypes of the application.

mVideoCast

mVideoCast is architected as a client-server system in which mobile clients publish content to a remote machine. The mobile application, implemented on the Android platform, can both record captured media on the device locally as well as stream content to a remote server in real time, allowing remote users to view live content. Users can control when the application is recording content locally and when content is streaming live to a remote server.

[t]



(a) (b) (c) (d) (e)

Figure 1: Screen detection. The application's first estimate shown in yellow (a) is inaccurate, but the second (b) correctly identifies the screen. When the user locks the ROI it changes color to red and the application displays a transparent image of the warped and cropped ROI in the lower-right, (c). As the user repositions the device, the application detects the motion and begins finding a new ROI, (d). The user can press a button to lock the ROI to the new estimate, (e).

Capturing and correcting the image frame

The mobile application extracts a quadrilateral ROI from each video frame, warping and cropping the frame to match a predefined output resolution and aspect ratio. The application runs three separate threads in order to save content locally, stream content to a remote server, and detect ROIs. The application first opens the camera and requests to receive preview frames. It stores received frames in a queue, and if it is in record mode it saves queued images locally. It then pushes frame events to the other two threads. The stream thread checks to see that it is in stream mode and that it has processed the last event. It then generates a compressed color image of the frame and posts it to a remote server. The detection thread similarly checks that detection is enabled and that it has processed the last frame. It then generates a grayscale image and runs the screen detection algorithm (methods for defining the ROI are described in the next section). Once a candidate quadrilateral is found, it is set in the main thread and shown on the display as an overlay drawn over the video preview (see Figure 1).

Audio

Because video frames undergo several processing steps and may not maintain a constant frame rate, audio is captured, saved, and streamed separately. While recording, captured audio is saved to a local file. While streaming, audio is Speex [12] encoded and forwarded to an Icecast [4] server running remotely. A web page associated with each user merges the audio with processed frames.

Selecting a region

Users can define a ROI using a variety of methods ranging from fully manual to fully automatic.

Manual selection

Users can set the ROI by tapping on the screen to set the four corners of the quadrilateral. The corner nearest the tapped

point is set to the tapped location. There are also shortcuts to set standard sizes more quickly; clicking on the upper-left and lower-right corners of the quadrilateral in rapid succession define a rectangle, and double-tapping anywhere on the screen sets the entire preview frame as the captured region.

Light tags

If the user has control of the display he wants to capture, he can switch the mobile application to a mode that detects light tags (e.g., LEDs) attached to the display. In this mode the mobile application detects the bright points corresponding to the light tags to determine the corners of the ROI.

Screen detection

Users can enable a special screen detection mode that will automatically attempt to determine screen regions within frames. We make use of the JJIL toolkit [5] in order to detect screens in a frame as follows: 1) Run a Canny edge detector over a grayscale version of the frame; 2) Remove all but the top 5% most significant edges from the result; 3) Divide the remaining points into four regions representing the top-half, bottom-half, left-half, and right-half of the image; 4) Send the subregions to a Hough line fit algorithm to find the dominant line in each subregion; 5) Construct a quadrilateral from the resulting lines.

This approach is typically not immediately accurate, so while in screen detection mode the application presents the current ROI estimate every few seconds (see Figure 1). When the user is satisfied with a result, he can use a hardware button to "lock" the current ROI coordinates in place. When a user locks an ROI, the application displays a thumbnailed view of the warped and cropped region in the lower-right of the screen.

The mobile application continuously monitors the mobile device's onboard accelerometer and compass to detect a change in position. When it determines that the device has moved, it begins generating new potential ROIs. While it generates new estimates it also maintains the past ROI. When the user decides that a new estimate is a better match, he can click the hardware button to set it as the current ROI.

The user can also adjust the corners of the quadrilateral at any time using the manual controls described above.

Face detection

In another mode the application uses Android's face detector libraries to set the ROI by automatically updating the ROI with the location of the most salient face detected in each frame. In this mode the application does not adjust the output aspect ratio and only crops the frame.

Scenarios

mVideoCast can be used for a variety of tasks.

Reporting: A news reporter is interviewing a subject on the street. By using the face detection mode, mVideoCast allows the reporter to easily stream a focused view of the interviewee, reducing distractors in the video and possibly preserving the privacy of bystanders.

Remote demonstrations: A business user wants to present a demonstration of a software application while sitting at a cafe. He starts mVideoCast on his phone and can stream only his screen to the remote participants; the system will crop regions out of the screen such as the cafe surroundings and his croissant.

Troubleshooting: Alice is trying to send a FAX internationally using the machine in her office, but the machine stops unexpectedly while processing her paper. She decides to call support and puts her phone on loud-speaker, launches mVideoCast, and points her device at the printer's LCD screen. The software automatically detects the boundaries of the LCD screen, un-warps its image and sends it to support who can view it and guide her to a solution.

Related work

While other research projects have explored video retargeting, or automatically selecting salient subregions of a video for redisplay on smaller screens such as mobile devices [7], mVideoCast uniquely allows users to stream specific ROIs from a mobile device.

There have been a variety of applications that have used ROIs in non-mobile video. Researchers have investigated user- and group-defined ROIs to control cameras for remote collaboration tasks [8, 11]. Similarly, the Diver system allows users to create videos from cropped clips of a prerecorded, panoramic video [9]. Other tools have explored automated solutions. El-Alfy et al. investigated automatically cropping surveillance videos to salient events [2]. Finally, a variety of systems have been developed to redact individuals from video recordings or video conference streams (such as [1]).

Conclusion and future work

In the past remote communication suffered primarily from a lack of bandwidth. Today networked, mobile multimedia devices are ubiquitous, and the core challenge has less to do with how to transmit *more* information than with how to communicate the *right* information. mVideoCast is a small but important step toward this goal.

In future work we plan to integrate optical motion compensation to maintain the alignment of the ROI between explicit detections. The step of locking coordinates provides an anchor to which subsequent frames can be registered. That is, if the ROI is locked at frame 0, subsequent frames 1 ... N can be aligned to this reference frame 0 with well known automatic image registration techniques. For instance, we can compute a transformation given a set of corresponding image coordinates, determined by matching image features (see Figure 2). In this way, the initial lock can be used without losing the position due to camera motion.

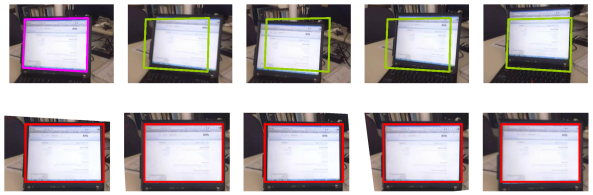


Figure 2: Using image stabilization to improve screen detection. We are experimenting with optical flow techniques that can make minor adjustments to the original frames (top) in order to keep the ROI registered without user intervention (bottom).

Extras

A video of mVideoCast is available at http://bit.ly/c0Hvc2.

References

[1] D. Chen et al. Tools for protecting the privacy of specific individuals in video. *EURASIP J. Appl. Signal Process.*, 2007(1):107–107, 2007.

[2] H. El-Alfy et al. Multi-scale video cropping. In *MULTIMEDIA '07: Proceedings of the 15th international conference on Multimedia*, pages 97–106, New York, NY, USA, 2007. ACM.

[3] G. Hua et al. Automatic business card scanning with a camera. In *International Conference on Image Processing (ICIP)*, Los Alamitos, CA, USA, 2006. IEEE.

[4] Icecast.org. http://www.icecast.org.

[5] Jon's Java Imaging Library, for mobile image processing. http://code.google.com/p/jjil/, 2010.

[6] JotNot. http://www.jotnot.com.

[7] F. Liu and M. Gleicher. Video retargeting: automating pan and scan. In *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, pages 241–250, New York, NY, USA, 2006. ACM.

[8] Q. Liu et al. Flyspec: a multi-user video camera system with hybrid human and automatic control. In *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*, pages 484–492, New York, NY, USA, 2002. ACM.

[9] R. Pea et al. The diver project: Interactive digital video repurposing. *IEEE MultiMedia*, 11(1):54–61, 2004.

[10] Qik. http://qik.com/, 2010.

[11] D. Song, A. et al. Exact and distributed algorithms for collaborative camera control. In *In The Workshop on Algorithmic Foundations of Robotics*, pages 167–183, Berlin, Germany, 2002. Springer-Verlag.

[12] Speex: A free codec for free speech. http://www.speex.org.