Jiku Live: A Live Zoomable Video Streaming System

Arash Shafiei, Ngo Quang Minh Khiem, Guntur Ravindra
Mukesh Saini, Cong Pang, Wei Tsang Ooi
Department of Computer Science
National University of Singapore

ABSTRACT

We present *Jiku Live*, a client-server system that supports zoom and pan operations in live video streaming from network cameras. The client is an Android mobile application that plays back live video from a selected camera and supports multi-touch zoom and pan interaction. The server acquires video streams from network cameras and transcodes the video feeds into one-second video segments at multiple resolutions. The transcoded video supports random access into any region-of-interest (RoI) within the video. Upon receiving zoom or pan requests, the server transmits the RoIs from the corresponding video segments to the client.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Video

General Terms

Design, Performance

Keywords

Zoomable Video, Live Video Streaming

1. INTRODUCTION

Many network cameras are capable of capturing and streaming high-definition videos for live viewing on remote clients. Such systems are useful in many contexts, such as video surveillance, e-learning, and event telecast. In these applications, it is common for a viewer to zoom into and pan around a video in order to view a region-of-interest (RoI) within the video in higher detail. Such *zoomable video* has been proposed in the literature [4, 3, 2]. To support zoomable video, video frames can be split into a grid of non-overlapping [3, 4] or overlapping [2] tiles. Tiles that overlap with the given RoI are sent to the client for decoding.

Existing work for zoomable video streaming is designed for pre-recorded video. In this paper, we present *Jiku Live*, a system that supports *live* zoomable video streaming from off-the-shelf network cameras. Jiku Live is a client-server system, where the server (called Jiku Video Server) is responsible for converting video streams from network cameras into tiles before streaming the RoI to the clients. The client (called Jiku Live Player) is a typical streaming video client with added support for zoom and pan operations. It communicates with and receives video streams from the Jiku Video Server.

MM'12, October 29–November 2, 2012, Nara, Japan.
ACM 978-1-4503-1089-5/12/10.

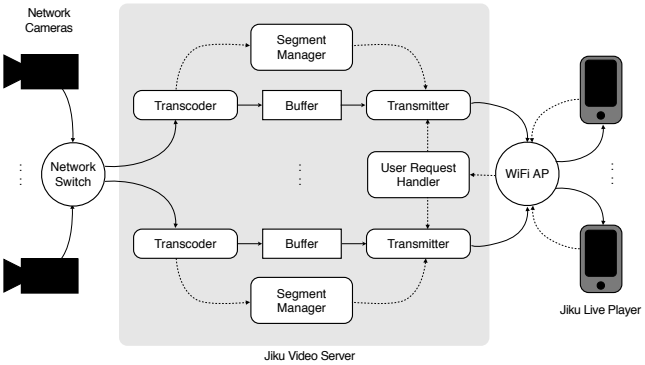

Figure 1: Jiku Architecture

Jiku Live Player also allows the user to switch between the available network cameras (See Figure 2).

To enable zoom and pan in live video, we use virtual tiling proposed by Feng et al. [1]. In this method, video frames are encoded with motion vectors limited to a tile. The encoded macroblocks are organized as slices, one per macroblock. Jiku Video Server parses an encoded stream to generate an index that has references to the tile offsets. Cropping of an RoI is translated to a random access operation, where tiles intersecting with the RoI are picked from the stream and composed as packets. As a result, each client can crop into a different RoI simultaneously. Zooming in and out is implemented with bitstream switching. The input video stream from one network camera is encoded at multiple frame dimensions. The index to tiles is maintained separately for each frame dimension and cropping is performed using the index structure for a specified zoom level.

2. ARCHITECTURE

We now elaborate on the architecture of the Jiku Video Server and describe the Jiku Live Player.

2.1 Jiku Video Server

Jiku Video Server has three main sub-systems: a transcoder, a segment manager, and a transmitter. These sub-systems are interlinked by a pipeline architecture, with a shared memory buffer serving as a common area to which encoded videos are written to and read from during streaming. The overview of Jiku Video Server is presented in Figure 1.

Transcoder: The transcoder runs as a multi-threaded process on the video server. It receives video streams from the network cameras, converts each video stream into video segments, each of

Figure 2: User interactions include (a) zooming, (b) panning, and (c) switching camera

one second duration, and encodes each segment into multiple frame resolutions with virtual tiling. Each tile is a 32×32 pixels region that is encoded in a way that limits the motion vectors to within the same tile. The encoded segments are then parsed to identify the byte offsets of each tile boundary. These offsets are sent to the transmitter over a control channel. The video segments are stored on a video buffer that can also be accessed by the transmitter. Once a segment of video has been encoded, the transcoder sends a signal to the segment manager.

Segment Manager: On receiving a signal from the transcoder, the segment manager loads the index into an in-memory trie. The trie stores the frame numbers within the segment, and for each frame the list of tiles. Then for each tile, it stores a list of slices (and their byte offsets) in that tile. The root of the trie is held in a circular queue, which contains the trie of different video segments from the same network camera. There is one such queue per zoom level (frame dimension).

Transmitter: The transmitter streams video segments to the users over UDP. Users can specify the RoI they want to view from any camera feed at a chosen zoom level. These user inputs are provided to the transmitter. The transmitter accesses the corresponding trie in the segment manager, and looks up the byte offsets of the slices in the tiles corresponding to each user's RoI. The slices at these byte offset positions are then read, packetized, and transmitted.

2.2 Jiku Live Player

The Jiku Live Player is an Android mobile application implemented using ffmpeg libraries. We added an interface for zoom and pan based on multi-touch gestures. Users can zoom in/out by using two-finger pinch/unpinch, and pan by dragging on the screen. The player manages the display buffer, and translates the decoded RoI to the correct coordinate space for proper display. The player also allows users to cycle through the network cameras present in the system.

3. SYSTEM PERFORMANCE

We tested the Jiku system with two AXIS network cameras and three Samsung Galaxy SII Android phones connected via a 802.11n wireless network. The cameras, the wireless access point, and the Jiku server are networked via a Cisco switch. Ten virtual connections to the server were also used to load the streaming system. The virtual connections change the RoIs to random locations every five seconds. Jiku server runs on a 2.66 GHz Xeon Quad-Core machine with 8 GB RAM. The video received from each camera was encoded into three resolutions namely 384×288, 640×480, and 1280×960. The end-to-end delay of the system is three seconds out of which about one second is the delay in fetching data from camera, one second is the delay we introduce for buffering of the video segments, and one second is other processing overheads. The interaction delay is three seconds as well. Under our current compression settings (ffmpeg implementation of MPEG-4, I- and P-frames only, VBR with quantization scale of 4) with a typical scene in a lab, the bitrate between cameras and server at the resolution of 1280×960 ranges from about 450kbps to 4 Mbps. The RoI bitrate of the stream received by client ranges from 128 kbps to about 800 kbps, depending on the amount of motion in the scene.

4. THE DEMONSTRATION

During the demonstration, we will show the Jiku Video Server and Jiku Live Player. The server will receive video streams from AXIS P1347 network cameras and wirelessly transmit the transcoded video to Samsung Galaxy SII phones running the Jiku Live Player. Users can view the video streams, select a camera view of choice, zoom and pan into regions of their interest. The cameras will be capturing the scene at the conference where the demonstration will be conducted. Attendees will be able to judge interaction latencies, encoding delays, practical issues involving computation complexity and limitations of wireless infrastructure.

Acknowledgement

This research is conducted under the NExT Search Center, supported by the Singapore National Research Foundation and the Interactive Digital Media R&D Program Office of Media Development Authority under research grant WBS:R-252-300-001-490.

5. REFERENCES

[1] W.-C. Feng, T. Dang, J. Kassebaum, and T. Bauman. Supporting region-of-interest cropping through constrained compression. *ACM Transactions on Multimedia Computing, Communications and Applications*, 7(3):17:1–17:16, Aug. 2011.

[2] S. Halawa, D. Pang, N.-M. Cheung, and B. Girod. ClassX: an open source interactive lecture streaming system. In *Proceedings of the 19th ACM International Conference on Multimedia*, pages 719–722, Scottsdale, Arizona, USA, 2011.

[3] M. Inoue, H. Kimata, K. Fukazawa, and N. Matsuura. Interactive panoramic video streaming system over restricted bandwidth network. In *Proceedings of the 18th ACM International Conference on Multimedia*, pages 1191–1194, Firenze, Italy, 2010.

[4] N. Q. M. Khiem, R. Guntur, A. Carlier, and W. T. Ooi. Supporting zoomable video streams with dynamic region-of-interest cropping. In *Proceedings of ACM Multimedia Systems*, pages 259–270, Scottsdale, Arizona, USA, 2010.