TOWARDS NEW FORMS OF NEWS GATHERING THROUGH CROWDSOURCED LIVE MOBILE STREAMING SYSTEMS

Ray van Brandenburg[1], Omar Niamut[1], Arjen Veenhuizen[1], Gert-Jaap Hoekman[2]

[1]TNO, The Netherlands; [2]NU.nl

ABSTRACT

Crowdsourced Live Mobile Streaming applications, such as Meerkat and Periscope, have seen an explosive growth in their popularity in the past few years. Whereas these applications provide great opportunities for crowdsourcers to directly share their experiences around live events, the unedited nature of these user-generated video streams makes them less suited for enriching news broadcasts or event reports. For such cases, the ability to select and edit streams as they come in, or to communicate with reporters in the field, are primary requirements for any editorial office or newsroom. This paper reports on the design of, and experimentation with, a crowdsourced live mobile streaming system and application for: requesting, receiving, filtering, directing, editing, and broadcasting live video streams from both consumers and professionals. This enables new forms of crowdsourced news gathering. The paper incorporates results from a number of technology validation tests and demonstrations, performed in collaboration with Dutch media partners.

INTRODUCTION

With the growing popularity of social media sites, online video services, and smartphones, content consumers are recording, editing, and broadcasting their own stories. Social media sites have evolved from text and photo streams to a rich medium for sharing audiovisual content. More and more mobile users are capturing and sharing video content, which can largely be attributed to the increased availability of video-recording capabilities on personal and mobile devices such as smartphones and their integration with online content-sharing platforms [1]. Amateur video capturing has also evolved from a personal hobby to prosumer usage for e.g. eSports broadcasting and citizen journalism. New crowdsourced live mobile streaming (CLMS) applications, such as Meerkat[1] and Periscope[2], have seen an explosive growth in popularity in the past few years [2]. In such systems, a large number of geo-distributed users publish live video streams from their mobile devices for an even larger audience to view. As a result, the role of these mobile Internet users during live events sees a shift, from taking part in a traditional passive audience, to acting as a content creator/participator, i.e., a crowdsourcer. In particular for live events, people from around the world are offered the ability to watch what is happening through the “eyes” of the crowdsourcer. CLMS systems offer event organizers a new and rapid way to distribute content to audiences; content that is unpolished, but genuine and real.

[1] http://meerkatapp.co
[2] www.periscope.tv

Most of the existing CMLS applications provide opportunities for crowdsourcers to directly share experiences around live events, but the unedited nature of user-generated video streams make them less suited for enriching news broadcasts or event reports. Broadcasters and media outlets are on the lookout for the best method to incorporate the potentially valuable source of content into their existing networks and workflow, for augmented TV broadcasts and citizen journalism in contemporary newsgathering. Curation of these crowdsourced streams, in combination with providing context and professional reporting, are crucial aspects of such a method. This comes with a set of functional (i), and technical (ii), challenges, e.g. (i) ensuring a low threshold for crowdsourcers with single-button streaming, filtering the incoming streams based on quality, allowing editors to 'direct' live streamers, making streamers feel appreciated and thereby more likely to continue streaming, and (ii) providing for reliable low-latency video streaming, a scalable backend and easy deployment, seamless switching between audio and video streams and catering for a huge diversity of sources and sensors.

This paper reports on the design of, and experimentation with, Cameraad, a CLMS system and application for requesting, receiving, filtering, directing, editing, and broadcasting live video streams from consumers as well as professionals, enabling new forms of crowdsourced news gathering. The system features a robust and modular system design and a simple user interface for on-the-fly editing and stream selection by the video editor. In particular, we (i) discuss the underlying design principles and use of open-source and web-based technologies such as WebRTC and GStreamer; (ii) show how we have derived and implemented a cloud-based architecture that allows for rapid deployment of the entire ingest, production and editing system; and (iii) present results from a number of technology validation tests and demonstrations, e.g. during the Grand Depart of the 2015 Tour de France in Utrecht, the 2015 Four Days March in Nijmegen, and a 2016 football game between PSV and Atlético Madrid, all taking place in The Netherlands.

RELATED WORK

Industry pioneer Twitch.tv[3] allows anyone to broadcast their content to large numbers of viewers, and the primary sources come from game players, from PCs or from other gaming consoles, e.g., Playstation 4 and Xbox One. According to Twitch's Retrospective Report 2013, in just three years, the number of viewers grew from 20 million to 45 million, while the number of unique broadcasters tripled to 900,000 [3]. Youtube Live and Facebook Live [4] are dedicated features of the respective well-known platforms that allow virtually anybody to easily broadcast themselves to a large audience. Pires and Simon [5] did an early study of YouTube Live and Twitch as emerging live streaming services. YouTube Live is functionally similar to Twitch, but with more general content and a variety of different channels. Their log data found that both services offered a choice of live content at all hours of the day, although they did exhibit diurnal and weekly patterns. They found that 30% of Twitch sessions lasted 60-120 minutes, which appeared to be largely driven by the length of the shared gaming activity. Meerkat and Periscope are among the most recent and popular CLMS applications. Periscope allows users to login with their Twitter account; as such, users are then provided with a list of public live-streams that others are currently broadcasting and one click gives one full access to what the "Broadcaster" is seeing in real-time. The live-stream is also accompanied by a viewer count indicating how many other people are co-watching the stream. In addition, viewers

[3] www.twitch.tv

can comment and 'heart' the stream for everyone to see. Streams with the most 'hearts' (or, as Periscope has dubbed them, the "Most Loved") end up with the highest rankings in the app, which attracts even more viewers. Periscope, now owned by Twitter, is set apart from Meerkat in two ways. First, it integrates deeply with Twitter, making it easy to create an account and Tweet a link out to the world. Second, Periscope offers a 'Replay' option where users who weren't around during the live stream can go back and watch.

BeFirst[4], developed by LiveU, leverages their live IP contribution technologies for citizen journalism. BeFirst has two parts: a software component, or "SDK" that plugs into an existing mobile application and that provides live video streaming, and the LiveU Central hosted back-end system that allows control of this stream via any web browser. The integration of BeFirst into existing mobile applications adds the ability to stream high-quality live video from any smartphone that has installed the application directly to a back office. In addition, it allows editors to determine which of their app-bearing users are in any specific geographical area, and message them directly. MakeTV[5] is a live video cloud solution for acquiring, curating and distributing incoming video streams from a variety of devices. Similar in its design to Cameraad, it features multiple stream inputs and outputs, communication with reporters, and newsroom and live production workflow integration. Both BeFirst and MakeTV primarily target professional mobile reporters, rather than untrained crowd reporters.

Within the EU FP7 project STEER [6], an application was developed augmenting a live event broadcast with live user-generated video originating from mobile devices. Event participants made live recordings with mobile devices, that could be streamed and watched fully synchronized with the live broadcasts by viewers at home. In addition, an analysis of social network messages on Twitter was performed to retrieve and show the most relevant posts in conjunction with the video. By providing additional event-related social and video content, viewers can enjoy an augmented view of live events.

CAMERAAD: CLMS FOR CITIZEN JOURNALISM

Cameraad was developed with the aim of supporting newsrooms to efficiently integrate live user-generated content (UGC) streams, captured and contributed live by citizen journalists. This required seamless integration of crowdsourced live streams into the newsroom workflow and editorial process. The system was co-developed by and tested with NU.nl[6], the largest Dutch online news provider. NU.nl reaches 2.7 million people on a daily basis, and has 4.2 million subscriptions to their breaking news services. Dedicated UGC editors oversee the insourcing, verification and curation of crowdsourced content, such as photos and offline videos, support integration of such content into their news feed, and allows their users to contribute and comment on news events. No less than 30% of their photo content in the domestic news section is crowdsourced, and UGC provides for an important

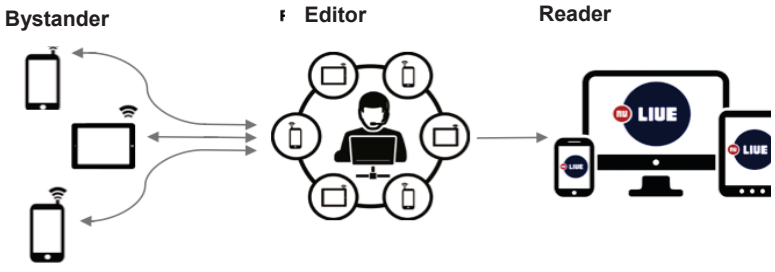

Figure 1 – Cameraad concept.

[4] http://www.liveu.tv/befirst
[5] https://www.make.tv/
[6] http://www.nu.nl/

source in developing news stories. The Cameraad CLMS offers NU.nl an extension to their crowdsourced inputs, beyond images and offline video, to increase the timeliness of their news reports. In particular, they sought to maximize both the quality and quantity of incoming streams, to filter and select among the incoming streams, and to provision the interaction between crowdsourcers and newsroom editors. Figure 1 depicts the Cameraad concept, implemented as part of the NUlive service.

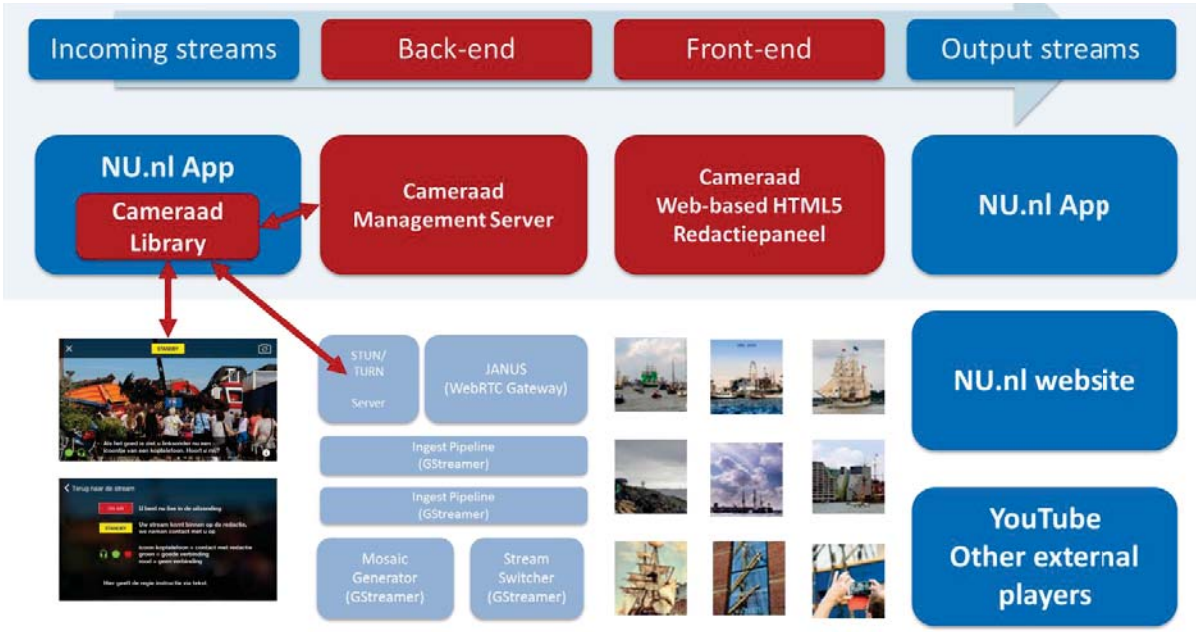

Figure 2 – Cameraad system overview.

System overview

The Cameraad system, shown in Figure 2, has a modular setup; on the client side, the Cameraad library can be integrated with an existing mobile application for live video streaming functionality; on the server side, a management server and editor web panel allow for service control and news production, with outlets to websites, online video portals such as YouTube and mobile applications. The underlying platform ensures the ingest and routing of incoming video streams.

Client module and editor panel

The client library has been developed as a standalone library for easy integration in mobile applications. The library incorporates the Google WebRTC implementation to stream Opus audio [7] and VP8 video [8], at a typical total bitrate of 800-1024kbps. WebRTC[7] was developed by the Google Hangouts team to support audio and video technology in browsers, without the need for dedicated plugins. The library further provides for communication with the management server via Websockets for initial session setup. A separate peer-to-peer WebRTC-based audio link allows for communication between a streamer and an editor via the editor web panel. Finally, the WebRTC library offers ICE Negotiation with a STUN/TURN server (defined below) for NAT traversal [9]; a crucial

[7] https://webrtc.org/

feature for reliable video streaming over mobile networks. ICE (Interactive Connectivity

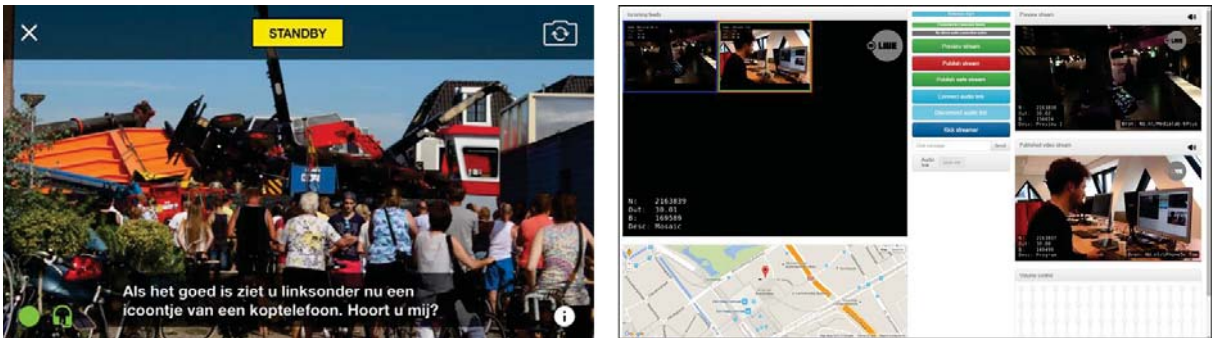

Figure 3 – Cameraad client module (left) and editor panel (right) user interfaces.

Establishment) is a framework to provide reliable IP set-up and media transport.

The editor panel provides an easy-to-use web-based interface (JavaScript, CSS, HTML5) for editors to monitor incoming streams, and to select and route streams to a live newsfeed in an interactive mosaic-style user interface. Separate mosaic and switcher streams are received via WebRTC from the backend, allowing for simple low-latency streaming to the web admin panel, and Websocket-based communication is maintained with the management server. Editors can setup a separate peer-to-peer audio or text chat link for communication with live streamers. Google Maps API integration provides for signaling the current location of crowdsourcers, allowing editors to perform position-based filtering of streams. Figure 3 above shows the client module and editor panel user interfaces.

Management and backend server

The management server handles the video room for each streamer: monitors the streams that join and leave or experience connection issues, and it spawns stream-specific ingest processes for each client, depending on the stream type the client is sending. It is aware of incoming stream details, ongoing interactions with streamers, and controls stream switching and routing, by signaling the mosaic structure (e.g. size and sources), and which stream to output to which destination (e.g. to the web admin panel or a YouTube Live channel for large-scale distribution). All Cameraad front-end components run on a cloud-based backend platform (see Figure 4), deployed on an Amazon EC2 instance. The backend can scale with the number of incoming streams and has a modular setup, including the following components:

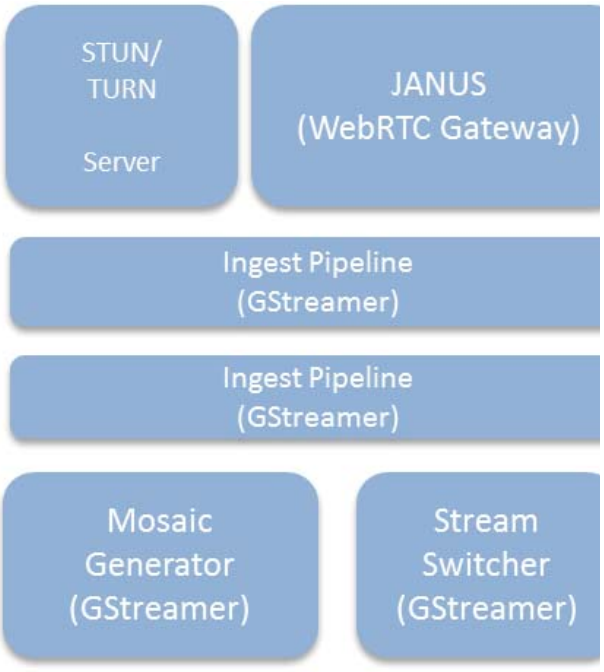

Figure 4 – Cameraad backend.

STUN/TURN server; STUN (Session Traversal Utilities for NAT) helps connect IP end-points, by discovering whether they are behind a NAT/firewall, and if so, to determine the public IP address and type of the firewall. STUN uses this information to assist in establishing peer-to-peer IP connectivity. TURN (Traversal Using Relay NAT) addresses this by providing a fallback NAT traversal technique using a

media relay server to facilitate media transport between end-points in corporate networks, where STUN is ineffective.

JANUS WebRTC Gateway; provides the means to set up a WebRTC media communication with an application, exchanging messages with it, and relaying RTP/RTCP and messages between applications and the server-side application logic.

All media processing in the Cameraad backend is handled by GStreamer, an open source media framework[8]. The following process can be discerned:

- *Ingest Pipeline*; a custom GStreamer pipeline tailored to receiving a specific type of audio/video feed. Its plugin architecture ensures minimal effort to create a plugin to receive a WebRTC stream, a live feed from a drone or a professional broadcast camera. The ingest pipeline output (both audio and video) is decoded in a standard intermediate format which is subsequently used by the *Mosaic Generator* and *Stream Switcher*. Since stream arrivals and departures are ad-hoc, and since the connectivity can vary heavily, the ingest pipeline makes sure that any lost packets and/or broken audio/video frames are replaced and/or duplicated.
- *Mosaic Generator;* dynamically creates a mosaic of zero or more incoming feeds. All tiles in the mosaic have equal dimensions. The output of the mosaic is encoded in VP8 and does not contain any audio feed.
- *Stream Switcher;* facilitates seamless switching between available streams. This is achieved by switching in the uncompressed audio and video domain. The audio/video output is then encoded with appropriate encoders. For WebRTC output, for example, VP8 and OPUS are used. For YouTube, H264 and AAC are selected.

EXPERIMENTS AND TRIALS

The Cameraad CLMS platform and application was tested during several large-scale events. Initial tests during development were performed during Liberation Day, on May 5th, 2015 and an emerging news story around a fire in a government building. In the first test, editors of Nu.nl reported live on several festival locations around the country, with the resulting live stream visible on the Nu.nl main website. In the second test, the unexpected live report rapidly attracted up to 15.000 viewers, and the live newsfeed was embedded on several other large Dutch news websites.

2015 Grand Depart Tour de France in Utrecht

A first large-scale live test was performed during the Grand Depart, the start of the Tour de France 2015, which took place in Utrecht, The Netherlands. Participants had to pre-register by mail, resulting in 40 semi-friendly crowdsourcers. No professional reporters were included. Incoming and selected streams were incorporated in the newsfeed, produced in a dedicated broadcast studio, and commented on by a professional reporting

Figure 5 - First large-scale test during Grand Depart Tour de France in Utrecht.

[8] https://gstreamer.freedesktop.org/

team. Having the geo-location of streamers available helped in briefing and preparing live streamers prior to their live contribution to the newsfeed. With close to 250,000 bystanders, the mobile network was severely overloaded, and the WebRTC Gateway was unstable. We noted that the value of UGC, to enrich an event that is already very well covered by professional registration , was limited, and that creating a storyline for the aggregated UGC proved difficult. Still, up to 10,000 viewers viewed the live stream. See Figure 5 for an impression of the dedicated broadcast studio and the incoming streams.

2015 Four Days March

A second large-scale test was performed during the Four Days March. During a period of four days, a dedicated Nu.nl reporter walked among the participants. 50 pre-registered crowdsourcers among the March participants and the audience provided several hours of live streams. Since the March took place close to the Dutch-German border, limited 3G/4G connectivity was experienced at several times. Still, up to 10,000 viewers returned to the live newsfeed during all four days. On the fourth day, the Cameraad system saw a major overhaul, with a new audio engine for: audio mixing and volume control, picture-in-picture feature, stream storage and real-time platform monitoring. Also, a fully automated system deployment approach was incorporated. See Figure 6 for an impression of the incoming streams in the editor web panel.

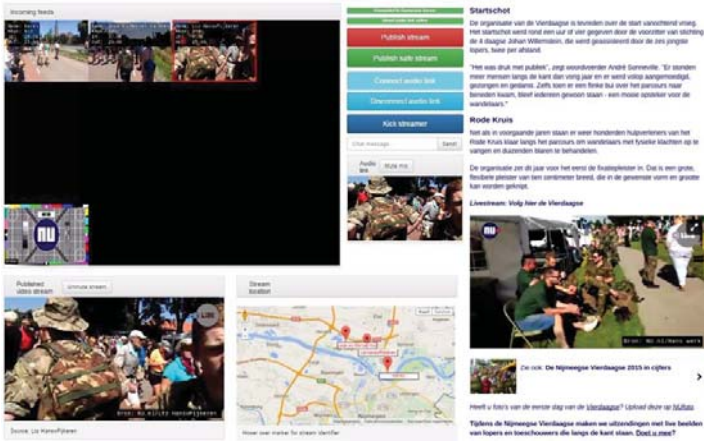

Figure 6 – Second large-scale test during Four Days March in Nijmegen.

2016 UEFA Champions League game: PSV vs. Atlético Madrid

A third large-scale test was performed during the UEFA Champions League game between PSV and Atlético Madrid, in February 2016. Close to 60 participants contributed up to 4 hours of video content in 260 individual streams. In addition, two bars in Eindhoven offered continuous live streams to incorporate in newsfeeds. The underlying system saw improvements in the export and storage of streams, with full configuration enabled via the editor web panel. A/V synchronisation of streams was improved to cope with bad connectivity. See Figure 7 for an impression.

Figure 7 – Final large-scale test during PSV vs Atlético Madrid.

SUMMARY AND OUTLOOK

The development and subsequent testing of the Cameraad CLMS provided us with several valuable insights; first, we noted that great value lies in the ability to cover unscheduled events, leading to a larger number of views. The integration of the Cameraad client library in an already-installed mobile application (in combination with geo-location features), allows a news provider to reach out rapidly to potential crowdsourcers in the neighbourhood of events. The ability to communicate with live stream contributors is highly appreciated by both editors and crowdsourcers, and increases the endurability of the service, i.e. ensuring people return to contribute. In particular for large-scale high-profile events, this communication is a dedicated editorial task. In our subsequent developments, we shall focus on: improving the stream selection process with respect to quality and content, alleviating the process of communication with streamers, annotating stored content for searching, controlling server-side audio settings, and supporting a full H.264 video codec for higher audiovisual quality. We also plan to investigate the application of the Cameraad CLMS in other domains, e.g. video-based surveillance and remote assistance.

REFERENCES

1. O. Juhlin, A.Engström, and E. Reponen. 2010. Mobile broadcasting: The whats and hows of live video as a social medium. Proceedings of MobileHCI 2010, 35-44.
2. Scott Kleinberg. 2015. Live streaming: The next big thing in social media. Chicago Tribune. Retrieved April 1, 2015 from http://www.chicagotribune.com/lifestyles/ct-so-sociallivestreaming-meerkat-periscope-20150401-column.html
3. William Hamilton, Oliver Garretson, and Andruid Kerne. 2014. Streaming on Twitch: Fostering Participatory Communities of Play within Live Mixed Media, Proceedings of CHI 2014, 1315-1324.
4. Josh Constine. 2015. Facebook Confirms Live Broadcasting Will Soon Open To Journalists And Verified Profiles, (August 12, 2015). Retrieved December 8, 2015 from http://techcrunch.com/2015/08/12/facebook-live-livestreaming/
5. Karine Pires and Gwendal Simon. 2015. YouTube Live and Twitch: A Tour of User-Generated Live Streaming Systems. Proceedings of MMSys 2015, 225-230.
6. STEER: Exploring the dynamic relationship between social information and networked media through experimentation. Retrieved April 7, 2016 from http://fp7-steer.eu/experiments/world-cup-rowing/
7. IETF RFC 6716, Definition of the Opus Audio Codec
8. IETF RFC 6386, VP8 Data Format and Decoding Guide
9. IETF RFC 5245, Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the Stimuleringsfonds voor de Journalistiek.