Automatic Mobile Video Director

Alexander Egurnov
University of Mannheim
aegurnov@mail.uni-mannheim.de

Thilo Weigold
University of Mannheim
tweigold@mail.uni-mannheim.de

Jon Pettersen
University of Oslo
jonup@student.matnat.uio.no

Alf-André Walla
University of Oslo
alfandrw@ifi.uio.no

***Abstract*—The abstract goes here.**

I. Introduction

In the last years the Internet has changed rapidly. Users not only retrieve information of simple websites anymore, rather they are sharing information, videos and pictures all over through networks. With the "Web 2.0", user-generated content has become an important part of the Internet we know nowadays. Well-known platforms like, YouTube, Facebook and Twitter make it easy for regular people to distribute data and media.

With the changing role of mobile phones from simple communication devices to devices with a multiplicity of functionalities, user-generated content has even seen a faster growth in recent years. Due to more precise sensors, integrated high-resolution cameras and of course faster mobile network technologies, it has never been so simple to contribute content when ever and where ever you want. This fact leads to complete new opportunities for web services using data and user information, in order to create dynamic content.

Let us imagine an interesting, public event like a political speech, concert or any kind of sport event, which is not filmed and streamed professionally by TV channels. Fortunately it has become common of spectators nowadays to capture parts of ongoing events by their phone camera. The potentials are tremendous. Why not using these user-generated media to provide a real-time stream or list of videos of the ongoing event, which not present people could therefore follow anyhow. A high quality mash up of an event requires an complex selection of all potential videos available, in order to cover the whole event with the best quality provided and being aware of bandwidth constraints of the mobile network.

In the following paper we want to deal with this kind of situations and introduce our implemented approach of an Automatic Mobile Video Director. Thereby we focus on the interaction between multi clients and one central server, our automatic video director, while addressing bandwidth and complexity problems, already mentioned above. Furthermore we do not focus on retrieving sensor information, which is done only in a rudimentary and simple way.

II. Related work

Describe articles and how our work differs from theirs. Throw in some references [1] so bibliography does not look empty. [2]

III. Methodology

A. System Overview

BRAINSTORMING: goal of our application? Client-Server architecture? RESTful Web Service.

B. Client application

BRAINSTORMING: archtitecture? android sdk, camera, mediamanager, sensor, http client, service class, sqlite.

C. Server application

Server general description goes here. Video storage, database connection, server framework description. RESTful Web Service

D. Client-server interaction

BRAINSTROMING: HTTP protocol,

1) Protocols: Our Automatic Mobile Video Director server implementation provides a general interface to applications which wish to interact with it. It is implemented through HTTP requests to certain server locations result.

GET /events
: Lists all events (including videos) in JSON.

GET /event/*id*
: Returns Event (including videos) in JSON.

POST /event/new
: Create new event from JSON. Expects request body to be a JSON string containing attribute *name*.

POST /event/*id*
: Upload JSON metadata about a video for Event with given *id*.

PUT /video/*video_id*
: Upload video *video_id* from Event *id*. Expects request body to be a file stream containing a full video file.

GET /video/*video_id*
: Retrieve video *video_id* from Event *id*.

GET /selected
: Retrieve a list of selected but not yet uploaded videos in JSON.

E. Metadata format

JSON vs XML arguments here As a final result we should state that metadata is transferred in JSON format.

id
: Client-side unique identification of the video.

filename
: File name in client's local file system.

timestamp
: Video creation time.

duration
: Video duration in frames.

width
: Video frame width in pixels.

height
: Video frame height in pixels.

shaking
: Amount of shaking detected by sensors.

status
: Video status. Indicates video life cycle phase.

serverId
: Server-side unique identification of the video. Needed for coordination of all clients.

IV. Video Director Algorithm

A. Video life cycle

B. Selection algorithm

V. Evaluation

How good/bad it is.

A. Data Traffic

Thilo can test it. I will have a look to this. If someone has experience please let me know.

B. Battry consumptio

Who wants to test it?

C. Selection criteria

VI. Future Work

Put down all the awesome ideas we have.

VII. Conclusion

The conclusion goes here.

References

[1] P. Shrestha, P. H. de With, H. Weda, M. Barbieri, and E. H. Aarts, "Automatic mashup generation from multiple-camera concert recordings," in *Proceedings of the International Conference on Multimedia*, ser. MM '10. New York, NY, USA: ACM, 2010, pp. 541–550. [Online]. Available: http://doi.acm.org/10.1145/1873951.1874023

[2] P. Seshadri, M. Chan, W. Ooi, and J. Chiam, "On demand retrieval of CrowdSourced mobile video," *IEEE Sensors Journal*, vol. Early Access Online, 2014.

TABLE I
TASK DISTRIBUTION

Part	Task	Subtask	Responsible
Android application	Video Capture		Thilo Weigold
	Sensor data collection		Thilo Weigold
	Metadata class		Thilo Weigold
	Http Client	Post Method	Thilo Weigold
		Cookies, Callbacks	Alexander Egurnov
		Get & Update methods	Alexander Egurnov
	Background upload service		Thilo Weigold
	SQLite database connection		Thilo Weigold
	Preferences		Alexander Egurnov
	GUI		Thilo Weigold, Alexander Egurnov
	Client-server data exchange		Alexander Egurnov
Server application	RESTful Server application	Client authorization	Alf-André Walla
		Request processing	Alf-André Walla
		Video and Event logic	Alf-André Walla
	MySQL database connection		Jon Pettersen
	Video Director		Alf-André Walla
	Client-server data exchange debug		Alexander Egurnov
	Video upload		Alf-André Walla, Alexander Egurnov
Web Server	MySQL Administration		Alexander Egurnov
	Nginx setup for streaming video		Alexander Egurnov
Documentation	Basic template formatting		Alexander Egurnov