Visual Twittering Using Mobile Phones in Pervasive Environments

Michael Wittke, Sven Tomforde, Yaser Chaaban, and Jürgen Brehm
Leibniz Universität Hannover, Institut für Systems Engineering, SRA
{wittke, tomforde, chaaban, brehm}@sra.uni-hannover.de

Abstract: Research in the field of academic teaching and life focuses on the integration of recent technologies. Additionally, new trends within the current usage of e.g. the Internet can be observed which show interesting approaches. These approaches might be reasonable and beneficial to further improve lectures and interaction with students by enabling these processes themselves to be community-oriented and based on Web X.0 principles. This paper shows how to integrate new recent trends from Internet technologies into the campus. We propose a system where students are able to use their mobile devices to do some kind of visual twittering (e.g. capture short video sequences and enhance these with additional data like GPS position, time, etc.). A campus-wide community will be developed, which provides access to these files, allows for following fellow students and friends and automatically generates an overview of the most important distinct events during the course of day for a given sub-community (e.g. students from the department of Computer Science). Based upon the introduction of the general scenario, the paper presents basic algorithms and technology needed for the realisation and a first evaluation.

1 Introduction

The rapidly increasing performance of highly-mobile networks and devices provides new opportunities for different application scenarios in pervasive and academic environments. Context-driven usage of such mobile devices and interactions with a pervasive environment and neighbouring devices of other persons enable services to support the user. Especially, the physical area of a campus offers a broad range of possible application scenarios for upcoming technical development.

We assume that in future students will be able to use life-logging and community-based mobile systems on the entire campus. The mobile devices interact with a pervasive environment in the context of Pervasive Computing (PC - cf. [MNSH01]). As we aim at a highly robust and distributed application, methods and approaches of Organic Computing (OC - cf. [Sch05]) will be integrated in our system.

Our scenario is based on the usage of smart mobile devices (MD). Recent years have shown a dramatic increase in computation power and sensor equipment on these devices (cf. e.g. [Webor]). Due to the increasing number of sold units - as e.g. mentioned in [Maw08] - the probability that each student will possess a MD in the near future is very high. Additionally, we can presume that a highly-powerful infrastructure (e.g. WLAN,

etc.) is available on the entire campus area. Looking at current trends in social communities and at the research spent on technical enhancements in academic life (e.g. the *Notebook University* [BBMS+03] or the integration of teaching and recent technologies [HvHKT06]), we assume that current approaches like *life-logging* and *social communities* [Web09f] will be further investigated and extended for academic use.

Such a system will rely on the logging of visual events using the built-in video camera of the MDs. The students will also be able to comment the files by adding a short phrase. Additionally, the system will automatically add position data (GPS-based), time information, and other possible meta-attributes (like e.g. users status, etc.). Afterwards, these data are transfered to the campus' community server which is responsible for the storage of the data and managing of the community as well as providing the interface to access the information. This centralised component will also be able to extract the most important events during the day for a given sub-community (e.g. all Computer Science students) using our approach to determine distinct events [WHHMS08]. Based on a saliency measurement, the most important events are determined by identifying the most frequent scenes.

This paper presents a first attempt towards a visual twittering and life-logging system combined with a context-aware usage for groups of students. Section 2 gives a short overview of related work done in connected fields of research. Thereupon, the initial visual twittering system is introduced in Section 3 combined with a first evaluation of the basic techniques used for this approach. This is followed by an algorithm to detect common salient events and cover the absence of in-house positioning data like GPS in Section 4. Finally, the paper concludes with a short summary and names the further research to be done until a realisation of the whole system itself is feasible.

2 Related Work

This paper addresses different aspects from the fields of social communities, blogging and micro-blogging, eLearning and eTeaching, as well as mobile smart cameras. This section aims at presenting a short overview of related techniques and research having aspects in common with our proposed system. Therefore, we start with a short introduction to social communities and their varying characteristics, followed by the academic environment and an introduction to the upcoming research field of mobile smart cameras.

Recently, social communities are receiving an increasing attention. A lot of people all over the world build networks on-line, keep their friends up-to-date about their activities, and share pictures and videos. Even in universities and colleges, social communities are formed where students can help each other, share information, or just do conversation and look up administrative information. This is a first step into the direction of creating on-line social communities for academic life. As e.g. Cummings et al. claimed, such social communities have advantages, if they are used in combination with usual social interaction and not instead [CBK02], which leads to the idea of driving the research towards an integration of both aspects.

The community presented in this paper has some similarities with existing systems from

the Internet, as we aim at using video data, which is also done by platforms like Youtube [Web09g] or Sevenload [Web09d] (or others like Flixawagon). In contrast to our system all of them do not support life-logging using video files or automatically adding position data and further meta-data to the files. Also, no building and managing of sub-communities is possible.

As we aim at combining a social community approach with automatic visual-based micro-blogging, the most important approaches related to our system are those dealing with life-logging. Until 2007 several persons tried to log their daily life using video cameras. During the 1990s, Steve Mann at the Massachusetts Institute of Technology (MIT) used a head-mounted camera to record his daily life on video tape [Man97]. In 2000, Gordon Bell at Microsoft Research experimented with a similar approach called MyLifeBits, where he documented every aspect of his work life [Webll]. During the time of the experiment, Bell wore a special camera around his neck (the *SenseCam*) which reacted on the body heat of other persons next to Bell and automatically took photos of them. Alternatively, it took pictures of a place, if it detected a change in light. Another approach by Ellis at the Columbia University focused on audio logging [Webwe], while Pentland at MIT Media Lab built a *socioscope* which continuously tracked the location of 81 volunteers and their communication patterns for 9 months.

Even more famous than these academic examples are on-line platforms available on the Internet. Most important are the well established social communities like Facebook [Web09b] or the German pendant StudiVz [Web09e]. Here, users are able to share personal information (e.g. date of birth, address, etc.) with their friends, upload photos, and keep friends up-to-date about their activities. The German website StudiVz started in 2006 with a focus on students and their activities. Some remains of this focus can still be observed as users can add lectures and tutorials to their profile and discuss the contents with other members of that course. A system which has its focus more on the logging aspect is Twitter [Web09f]. Here, users can upload short phrases called *tweets* (up to 140 symbols per phrase). Other users can become *followers* (observers of the specific person) and subscribe to these updates.

The proposed system is based on some aspects presented before and relies on the integration into the academic context. Here, research has been focusing on the integration of new media and recent technologies into eLearning and eTeaching approaches for a couple of years (cf. e.g. [HvHKT06]). Lectures are augmented with interactive parts using network-connected tablet-PCs. Additionally, platforms like Stud.IP [Web09a] are used to provide lecture content and allow for discussions and surveys for all participants.

Technically, our system relies on the usage of mobile devices (MD) equipped with smart cameras. As within our system each MD will be able to perform some calculations on its own and take sensor information (e.g. GPS sensor, movement sensors, etc.) into account, we assume the MD itself to be *smart*. Smart cameras have become part of active research since a few years. Smart cameras (SC) can be either mobile or static. For non-mobile scenarios Wolf et al. [WOL02] are working on enabling SCs to allow for the calculation of vision algorithms in real-time. Another approach by Hoffmann et al. deals with the self-organisation of distributed SC systems [HWHMS08].

As we use mobile SCs, restrictions of embedded systems are of special interest. Rinner et al. presented a SC as a fully embedded system, focusing on power consumption, QoS management and limited resources [BDM+06]. In contrast to our research - as presented in e.g. [WHHMS08] - they do not concentrate on out of the box MDs like mobile phones with networking capabilities. Another approach by Bollinger et al. proposes an architecture for mobile devices equipped with vision sensors [BKR07]. They focus on MDs using techniques like Bluetooth for the communication. Since the basic technical implementation is based on the usage of J2ME and Symbian, they are not able to use sophisticated computer vision algorithms due to performance issues. In contrast, our system aims at using standard image processing libraries being able to do image processing on the MDs with less restrictions. This background is the basis for the proposed system.

3 Mobile Visual Twittering

Twitter is a free social networking and micro-blogging service that enables its users to send and read other users' updates known as *tweets* (see [Web09f]). Tweets are text-based messages of up to 140 characters in length which are displayed on the user's profile page and delivered to other users who have subscribed to them due to interest in their activities (known as *followers*), e.g. friends. We introduce a new kind of tweets that are automatically fed by sensor input of the user's mobile phone. Based upon this sensor input, *visual tweets* are defined as consisting of visual data from the mobile phone's built-in camera and the user's position delivered by GPS sensors (and possible further meta-data).

After presenting the requirements and the technical setting of our system in Section 3.1, we introduce the concept of points of interest in Section 3.2. In Section 3.3 the system's backend is described and evaluated.

3.1 Requirements and Technical Setting

Visual tweets comprise visual information and location-dependent data such as GPS, time or other space-time relevant data. In our scenario of a campus-wide community consisting of students using visual twittering to protocol their day, the visual information is delivered by cameras of the students' mobile phones. The mobile phone's camera could be head-mounted or attached to the user's clothes having free field of view. The location-dependent sensor data is supported by a GPS device.

The GPS sensor can be connected directly to the student's mobile phone as it is commonly done in augmented reality scenarios such as described in [Webet]. Today's mobile phones are more and more equipped by sophisticated interfaces such as USB (Universal Serial Bus) [Webes]. This allows for connecting USB-based GPS sensors. If the GPS sensor is directly connected to the user's mobile phone, the sensor data can be synchronised on the device in real-time and sent to the backend server, where the visual tweets are stored. Likewise, the sensors delivering the user's position can be carried independently from the

camera and synchronised using timestamps. In this case, the GPS data may be delivered by external GPS devices such as tracksticks (see e.g. [Webom]). A trackstick is a small GPS device that continuously records the user's position for later download through a built-in USB connector. The visual data is uploaded in real-time to the backend server and post-synchronised with the trackstick's GPS data. Recorded data of the trackstick is encoded by the GPX format (GPS eXchange Format), which is an XML schema designed for describing GPS data. It includes date, time, location, speed, heading, altitude, and the exact length and location of any stops.

Another important trend in the mobile phone industry is the steadily rising availability of bandwidth. In our on-campus based scenario, the mobile phones use high-bandwidth communication methods such as 802.11 WLAN to upload all data to the backend server. In the case of extending the scenario to off-campus locations, high-bandwidth technologies are not available in all locations. Nevertheless, future technologies like LTE (Long Term Evolution) promise bandwidth increases up to 100 Mbps on the downlink, and up to 50 Mbps on the uplink (cf. e.g. [DEF+06]). Following this, mobile phones will be able to upload high-load media data from any places in future.

The backend server for storing visual tweets is an off-the-shelf webserver running Linux and offering a REST (Representational state transfer) interface (similar to [Fie00]). Using REST, additional communication overhead is avoided compared to other communication interfaces such as SOAP, which is crucial for mobile devices. The visual tweets are stored in a MYSQL database. As programming language PHP is used.

The visual tweets' visual information may be delivered by JPEG pictures or AVI media files. The backend server extracts the date, image size (width and height), file size and - in case of an AVI file - quality assurance parameters such as the video's length in frames and frames per second. To segment the AVI file into single JPEG pictures, the tool *FFmpeg* is used. *FFmpeg* as presented in [Webhu] is a tool to record and convert multimedia files.

3.2 Expanded Points of Interest

Points of interest (POI) are locations, which are defined by users following other users. Using this concept, followers can define regions of interest. If the track of a followed user approaches or enters this region, an alert message is sent to the following user. For instance, the university's campus can be defined as a POI (or special buildings on campus like refectory and examination office) as well as other locations such as a the city hall or the airport in off-campus scenarios. Since a POI may be a widespread region (e.g. the campus comprises several buildings), it cannot be defined by a single pair of GPS coordinates. Due to this restriction, we introduce the concept of *expanded points of interest*. Expanded points of interest are characterised hierarchically as follows: (1) a single pair of coordinates (e.g. a statue), (2) a polygon (e.g. one building), or (3) a set of polygons (e.g. a block of buildings). Using this concept, any POIs can be formed. Within the visual twitter backend this concept is encapsulated in a GoogleMaps mashup.

3.3 Visual Twitter Backend

A basic part of our concept is a central server - the Visual Twitter Backend (VTB). This is used to store the tweets, allow for the access to the tweets and communities, and manage the system. Additionally, we rely on the existence of a pervasive environment which is responsible for the communication with the user, a pre-processing of user data, and further augmented services in future (refine and pre-process information for the particular user taking his status into account). The pervasive environment can e.g. provide screens that can be automatically reused for a more detailed search (using the POI mechanism) with a higher resolution than the display of the mobile phone. Research in integrating the pervasive environment and the proposed community system are part of the future work.

Therefore, the VTB has to provide a web-interface for the following tasks: (1) Provide access the the campus-community and the particular sub-communities. (2) Administrate the community system. (3) Allow for the user-specific definition of POIs. (4) Provide possibility to manage the user's settings (following of users, profile, etc.).

The currently investigated feature is the *following-mode*. Therefore, the VTB has to provide an opportunity to define POIs. This is realised with a map-based approach. Currently, the map contains just the region of interest (the campus), but it can easily be extended to an off-campus view as it relies on the usage of GoogleMaps [Web09c]. The user defines expanded POIs as defined before using an interactive interface.

In the context of this paper, the sub-communities and the following-mode are of special interest and will be introduced in the remainder of this sub-section.

3.3.1 Sub Communities

The system relies on the possibility to form groups of students in order to build sub-communities out of the set of persons related to the campus. These sub-communities will have a common access page, through which the participants have access to the group functions (board, message service, find other persons, etc.). Additionally, a future version of our system will provide an automatically generated overview of the most salient events for this sub-community for a given period of time (usually the last day). Therefore, the VTB has to compare all received visual tweets, to determine the same events in different video files, and to calculate the most famous events. This has not been realised yet, but a first approach, which is able to locate users with the same field of view in in-house scenarios, is presented in Section 4. This localisation algorithm makes way for the future detection of global salient events.

The solution needs a possibility to define sets of participants belonging to a sub-community. As these groups are identified via a common attribute (e.g. members of the Computer Science department), this can be easily realised. The main feature of the groups is, that participants will be directly navigated to their group site and will be able to receive automatically generated messages (e.g. e-mail or SMS) if another person he is *following*, is next to him on campus. This information is determined by the distance calculation as presented in the next subsection.

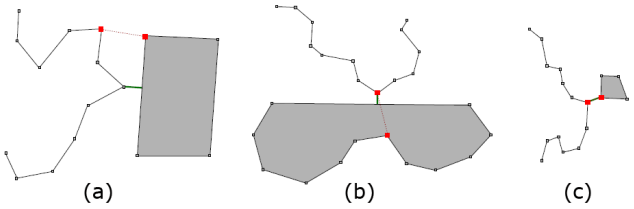

Figure 1: Rapid distance determination between routes and POIs (pre-selection step)

3.3.2 Following Mode

The most important functionality of the VTB is the user's following-mode. For following another user, the user's tracks have to be determined and compared to other tracks continuously. Due to this automatic calculation, the system can determine, if the user approaches or enters a POI defined by the follower. This is performed by a process consisting of two steps: Firstly, the tracks are tested by an imprecise but fast pre-filter for a pre-selection of POIs. Secondly, the distance between the POIs and the particular tracks is examined precisely (examination step). The pre-selection step is implemented by a simple database query. Those tracks are selected, whose minimal distance of one of its track points to one of the corner points (δ_{pre}) is lower than a pre-defined threshold t_1:

$$\delta_{pre} = min(\left\| poi_{i,j} - trac_{n,m} \right\|) \tag{1}$$

$poi_{i,j}$ determines the POI i and the element j of this set. Corresponding to this, $trac_{n,m}$ indicates track n and element m of this set. δ_{pre} is determined by computing the minimal distance between each of the points contained in the set poi_i and $trac_n$. If there are h points of interest and g tracks, i and n are defined as $i = (1, ..., h)$ and $n = (1, ..., g)$. Tracks matching the criteria $\delta_{pre} < t_1$ are added to the set of tracks, which are examined more precisely within the examination step. If the result of the examination step (δ_{ex}) is greater than t_2, the corresponding track is deleted from the set of tracks (i.e. $t_1 > t_2$). Nevertheless, the pre-selection step is a fast but imprecise heuristic approach, since δ_{pre} may deviate sharply from δ_{ex}. δ_{pre} is an upper bound, which is depicted in Figure 1: three POIs (gray areas) with a track each are illustrated. The dotted lines show the results of the pre-selection step. The solid lines are the real distances. The deviation between δ_{pre} and δ_{ex} increases with the POI's size, see Figure 1(a) and 1(b). δ_{pre} and δ_{ex} are almost equal for POIs consisting of a minimal number of elements, see Figure 1(c).

To compute δ_{ex} between a track and a POI, the minimal geographic distance between the POI's contour and the track's points are calculated. This is similar to calculating the distance from a point to a line, as depicted in Figure 2. The POI is leveled out vertically with respect to the track. If it intersects, δ_{ex} is defined by the length of this line (e.g. d_{min} in Figure 2). If there is no intersection, δ_{ex} is the distance between the track's point and the POI's nearest corner point. To perform this computation, the coordinates have to be transformed from a geographical to a Cartesian system and vice versa after the computation has finished. The computation of δ_{ex} can be summarised as follows:

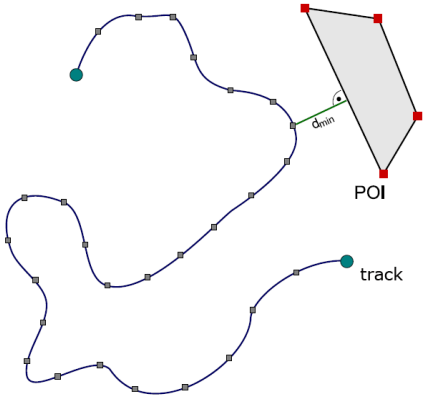

Figure 2: Computing distance between a track and a POI (examination step)

1. **Pre-selection:** Select neighbouring tracks of the POI ($\delta_{pre} < t_1$)
2. Set $MIN\_DIST = \infty$, $P\_RTE\_MIN = NULL$, and $P\_POI\_MIN = NULL$
3. For each point p of a neighbouring track and each line $\overline{AB}$ consisting of two corner points of the POI do:
   (a) Transform the coordinates of point p as well as the points A and B of $\overline{AB}$ from a geographical to a Cartesian system (result: p_{cart} and $\overline{AB}_{cart}$)
   (b) Compute that point p'_{cart} on $\overline{AB}_{cart}$ having minimal distance to p_{cart}
   (c) If $d(p_{cart}, p'_{cart}) < MIN\_DIST$, set $MIN\_DIST = d(p_{cart}, p'_{cart})$ and transform p'_{cart} to Cartesian coordinates.
   Set $P\_RTE\_MIN = p$ and $P\_POI\_MIN = p'$

$d(A, B)$ is defined by the Euclidian distance.

3.3.3 Accuracy

The accuracy of our system is dominated by the accuracy of the following-mode. It is limited by three factors: (1) the accuracy of the POIs' approximation, (2) the amount of recorded points and the correctness (i.e. the deviation from the real position) of the position, and (3) the accuracy of our distance computation. The accuracy of the POIs' approximation depends on the user, which adds the POIs to the backend's database. Since our system has no influence on this, a worst-case estimation of 75 m is used. This value has been heuristically determined by users entering a pre-defined POI.

The amount of recorded positions and the correctness of the position as recorded by the GPS device depend on the GPS system. The deviation of a standard GPS is approx. 10 m. This factor can be reduced by modern approaches such as dGPS, which enhances the position accuracy to centimeters (see [MMH05]). On the other hand, the accuracy of the

GPS device is determined by the frequency, the device records track points. E.g., the GPS device records a track point every 10 seconds while the user walks (velocity is 5 km/h). Each 14 $meters$ a track point is recorded, which means that the maximal deviation is approx. 7 m. To keep the deviation small, the sampling frequency of the GPS device has to be adapted to the user's velocity.

This means that the worst-case accuracy is approx. $75\ m + 10\ m + 7\ m = 92\ m$. The accuracy can be enhanced sharply by reducing the error received from the POI's approximation of the user. This can be achieved, if the user uses high resolution satellite maps as they can be found on GoogleMaps using a high zoom factor.

3.3.4 Performance

The evaluation of the performance has been performed using a real-world scenario. Therefore, one follower and three followed persons have been defined. The followed users recorded their way on campus with a GPS trackstick. Afterwards, two basic questions were answered: (1) How did the execution change, if the number of points forming a track was increased? (2) How did the accuracy of the distance computation change, if the number of corner points forming a POI has been reduced?

Within the scenario, the followed students were equipped with a GPS Trackstick II from *TELESPIAL SYSTEMS* (see [Webom]). This device has 1 MByte of flash memory and is able to record months of travel history. Its horizontal accuracy is 2.5 meters and tracks up to 12 satellites. The trackstick was attached to the students' belt via a belt clip. Pictures were captured by the students' mobile devices. To access the University's WLAN, a mobile Internet tablet *Nokia N810* [Webes] has been used. Each student's day contained three periods of 90 to 180 minutes of activity (i.e. walking) on campus. Every 10 seconds a GPS point was recorded, pictures were captured in periods of activity only. The pictures were stored in real-time on the webserver, whereas the GPS data was uploaded at the end of the day and synchronised with the students' pictures. The webserver possessed an *Intel Core 2 Duo* processor with $2.1GHz$ and $2GByte$ RAM. The follower defined three POIs: (1) the University's library (TIB), (2) the University main building (Uni), and (3) the building of the Computer Science department (Appelstr. 4). The first POI (TIB) was described by 5 corner points, the second POI (Uni) by 11 corner points, and the third POI (Appelstr. 4) had 7 corner points. At the end of the day, the follower was informed via e-mail, whether the followed students approached one of the pre-defined POIs.

To analyse how the execution time increases with the number of track points, the recorded tracks were duplicated to a volume of 10,000 data points. As depicted in Figure 3, the execution time increases linearly with the number of track points. The next part of the evaluation focuses on the accuracy of the distance computation depending on the number of track points describing a track. Decreasing the number of track points by 50% did not influence the accuracy significantly (in the worst case the accuracy decreased by 2.8%). The further evaluation showed that the execution time increases linearly with the number of track points. Nevertheless, the accuracy of the distance computation does not change significantly, if the number of track points is halved. This means, that we achieve the same accuracy and halve the execution time, if we reduce the number of track points by 50%.

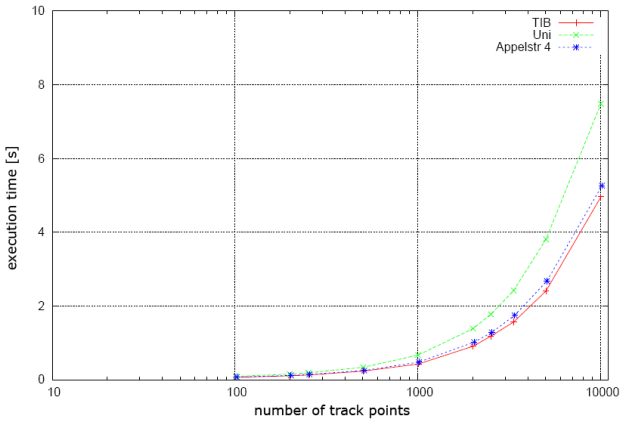

Figure 3: Execution time for distinct POIs

4 Using Visual Saliency for In-House Localisation

Mobile visual twittering relies on the users' mobility and the visual information captured by the MDs and uploaded to the backend. To make way for compressing the amount of data to essential scenes of the day, we propose a method, which localises users possessing the same field of view. Primarily, this method aims at dealing with in-house scenarios, where GPS services are typically not available due to the low signal strength. Therefore, we introduce a metric for measuring visual saliency in video streams using visual information only (see Section 4.1). Based upon this metric we derive visual saliency curves, which were originally used by frame synchronisation techniques to time-synchronise cameras having the same field of view. These curves are correlated and investigated with respect to characteristic peaks. If these peaks arise, the probability is very high that the synchronised cameras captured the same field of view (see Section 4.2). In Section 4.3 the applicability of our method is approved in a real-world scenario consisting of two cameras possessing the same field of view and observing the hallway of an office building.

4.1 Visual Saliency

Within this section we shortly discuss visual saliency - for further details the reader is referred to [WHHMS08]. Our visual saliency measurement is based on Itti's framework (cf. [IB05]) using the relative entropy (Kullback-Leibler KL) for quantifying surprise. To reduce the computational effort, the model has been adjusted from considering low-level features like luminance or color contrast to just calculating the optical flow vector [HS80] between subsequent frames.

$$S(t, t-1) = KL(P(f_t), P(f_{t-1})) = \int_{\Omega} P(f_t) log \frac{P(f_t)}{P(f_{t-1})} dM$$

This formula is part of Itti's framework, which is a bottom-up model for computing pixel-accurate surprise values in video streams. Ω is defined as the optical flow only to reduce the computational effort in comparison to Itti. To calculate the optical flow vectors we use the Lucas-Kanade algorithm (see [Luc84]) with these parameters: 400 feature points, 3x3 window. As image processing library *Intel's OpenCV* [BK08] is used. The feature channels are the absolute values and angles of the optical flow vectors simplifying the computation again. Therefore, only the absolute values for forming the visual saliency curves are considered, as they better describe an object's movement. Afterwards, the absolute values of the optical flow vectors for each frame are determined, clustered into groups (e.g. defined by thresholds), and a histogram f is calculated. f is the basis for forming the frame-based optical flow distributions $P(f_t)$ (time t) and $P(f_{t-1})$ (time $t-1$), respectively. $P(f_t)$ and $P(f_{t-1})$ are the histograms' relative frequencies of absolute values at the specific times t and $t-1$. These histograms are the basis for calculating the surprise measure $S(t, t-1)$, which is used by the in-house localisation algorithm described in the following section.

4.2 In-House Localisation Algorithm

To allow for in-house localisation, the visual saliency curves of the users' video streams and pictures are computed on the backend server. We assume that the MDs are synchronised in terms of time, e.g. by using algorithms and protocols such as NTP (cf. [Mil08]). Afterwards, the saliency curves of video streams possessing nearby GPS coordinates are correlated. This is performed using short time periods (such as 10 frames). If there is a sharp matching for more than 20 periods, the probability is high, that the cameras possessed the same field of view.

Based on these ideas and the theoretical foundation of visual saliency, the operational process is developed. Therefore, it is assumed that all MDs uploading their tweets are attached to the user (not moving, free field of view). Ideally, this could be determined during operation by using sensor information. When users upload visual tweets (including the corresponding GPS track and the user's ID to identify him) the visual saliency curves for these videos are computed and the correlation process is started. Therefore, the start t_s and the end time t_e (equals t_s + duration) of the video are stored. Both of them are used to find a video v (v_s is its start time and v_e its end time, respectively) in the set of already uploaded tweets, which have been captured within this period of time. The start time for the correlation process is equal to v_s and the end time is chosen as minimum over v_e and t_e. Afterward, this period of time is correlated in steps of 10 frames, if their GPS coordinates are nearby. E.g., in case of in-house scenarios the GPS coordinate is set to the last received GPS position before entering the building. In case of more than 20 matches, the fields of view of both videos are considered to be the same. This is done for each video

matching the conditions mentioned above.

Figure 4: Example of a salient visual event exceeding the threshold

4.3 Evaluation

For the evaluation of the proposed method the real-world applicability has been investigated by observing an office hallway. The cameras were synchronised beforehand and possessed an overlapping field of view. The amount of events appearing have been analysed to investigate the accuracy of the localisation algorithm.

4.3.1 Experimental Setup

The experiment has been performed using two *PTZ Axis 214* IP cams capturing pictures (QCIF resolution, 176x144) of an office hallway. The cameras were situated in an office (on an office table with a distance of 25 cm to each other) with an open door having a free field of view on the hallway. Their axis of view were parallelised and they possessed overlapping fields of view without any particular calibration. Every IP cam was connected via the university's WLAN (IEEE 802.11g) to the visual twitter backend. After finishing the capturing, the correlation process was started. Besides the WLAN access, microwave ovens, wireless sensors, and Bluetooth devices could be found in the building that may potentially interfere with the wireless transmissions of the IP cams. The synchronisation between the cameras has been done using NTP (NTP server: time1.rrzn.uni-hannover.de). Such a setup should be exemplary for real-world deployment assuming an increased popularity of home access points, wireless sensors, and community meshes [AJSS05].

4.3.2 Real-World Applicability

The cameras captured the office hallway from 9:00 AM to 3:00 PM. During this period, 148 events (9 AM to 12 AM: 114 events and 2:05 PM to 3:00 PM: 34 events) with a saliency value above a specific threshold have been measured. The threshold (here: 100) has been adjusted to allow for at least 5% change of content from one frame to the subsequent (e.g. by an obstacle entering or leaving the camera's field of view, see Fig. 4). This corresponds to an event appearance rate of 1 event per approximately 93 seconds ($(148\ events/13,746\ seconds) * 60 \approx 0.65\ events/minute$), proving the real-world applicability of our approach.

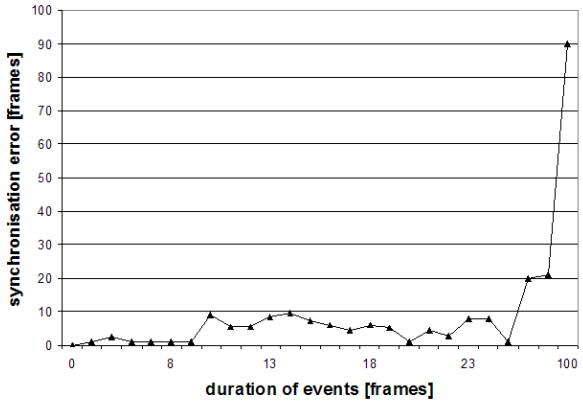

Figure 5: Synchronisation Error: In case of only considering events with a length of 10 frames, the mean error is about 2,55 frames

4.3.3 Localisation Accuracy

To measure the accuracy of this method, the setup described in Section 4.3.1 was used to observe the office hallway. After having observed a set of events, the video streams have been uploaded to the backend. The video streams of both cameras had the same GPS coordinates and were evaluated in terms of their saliency curves. This has been followed by the execution of the localisation algorithm. The evaluation as depicted in Figure 5 showed, that video streams of short events ($< 10\ frames$) can be correlated with an high accuracy (i.e. $+/-1\ frame$), in contrast to long events ($> 10\ frames$). Considering all events, the mean synchronisation error is $9.51\ frames$ with an standard deviation of $+/-11.71\ frames$. For events with a maximum length of 10 frames, the mean error is about $2.55\ frames$ with a deviation of $+/-6.69\ frames$. The events with a long duration can be filtered by the backend by dropping those with significant peaks beyond the length of 10 frames. This means, that our localisation algorithm can determine cameras having the same field of view by correlating their saliency curves. Nevertheless, if the events appearing in the video stream are longer than 10 frames, the accuracy of the localisation is reduced.

5 Conclusion

This paper presented a system for mobile twittering using mobile phones in pervasive environments on campus. We postulated a vision for integrating upcoming techniques from the Internet into a pervasive academic environment and forming communities for groups of students sharing the same interests. The setup of the system has been described in combination with a first evaluation of some basic techniques used for the approach. As the system relies on the existence of position data like GPS, we proposed an algorithm to enable in-house synchronisation based on salient visual events in the absence of position

information.

As the paper introduces a first step towards a fully working system, some research fields have still not been investigated. In future, we aim at developing a system, which is able to work with a pervasive environment (e.g. by supporting searches). Additionally, the development of the community platform has to be finished in order to allow for the establishment of sub-communities and the automatically generated summary of the day using the most salient events.

References

[AJSS05] Aditya Akella, Glenn Judd, Srinivasan Seshan, and Peter Steenkiste. Self-management in chaotic wireless deployments. In *MobiCom '05: Proceedings of the 11th annual International Conference on Mobile Computing and Networking*, pages 185–199, New York, NY, USA, 2005. ACM Press.

[BBMS+03] Jürgen Brehm, George Brancovici, Christian Müller-Schloer, Tarek Smaoui, and Sebastian Voigt. The Interactive Lecture - An Integrated Multimedia-Supported Teaching Experiment. In *Proceedings of the ICL 2003, Villach*, 2003.

[BDM+06] Michael Bramberger, Andreas Doblander, Arnold Maier, Bernhard Rinner, and Helmut Schwabach. Distributed Embedded Smart Cameras for Surveillance Applications. *Computer*, 39(2):68–75, 2006.

[BK08] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., 1st edition, October 2008.

[BKR07] Philipp Bolliger, Moritz Köhler, and Kay Römer. Facet: Towards a Smart Camera Network of Mobile Phones. In *Proceedings of Autonomics 2007 (ACM First International Conference on Autonomic Computing and Communication Systems)*, Rome, Italy, October 2007.

[CBK02] Jonathon N. Cummings, Brian Butler, and Robert Kraut. The quality of online social relationships. *Communications of the ACM*, 45(7):103–108, 2002.

[DEF+06] E. Dahlman, H. Ekström, A. Furuskar, Y. Jading, J. Karlsson, M. Lundevall, and S. Parkvall. The 3G Long-Term Evolution - Radio Interface Concepts and Performance Evaluation. In *VTC Spring*, pages 137–141. IEEE, 2006.

[Fie00] Roy T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

[HS80] B. K.P. Horn and B. G. Schunck. Determining Optical Flow. Technical report, Cambridge, MA, USA, 1980.

[HvHKT06] Inga Herbold, Ulrike von Holdt, Marc Krüger, and Thanh-Thu Phan Tan, editors. *Proceedings of the eTeaching and eScience meeting 2006 (german): "Lehren und Forschen mit Neuen Medien an der Leibniz Universität Hannover"*, 2006.

[HWHMS08] Martin Hoffmann, Michael Wittke, Jörg Hähner, and Christian Müller-Schloer. Spatial Partitioning in Self-organising Camera Systems. *IEEE Journal of Selected Topics in Signal Processing*, 2, Aug 2008.

[IB05] L. Itti and P. Baldi. A Principled Approach to Detecting Surprising Events in Video. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 631–637, Washington, DC, USA, 2005. IEEE Computer Society.

[Luc84] B. D. Lucas. *Generalized Image Matching by the Method of Differences*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 1984.

[Man97] Steve Mann. Smart Clothing: The Wearable Computer and WearCam. *Personal Technologies*, March 1997. Volume 1, Issue 1.

[Maw08] Neil Mawston. Nokia Reaches 40% Share as 332 Million Cellphones Ship Worldwide in Q4 2007. Technical report, Boston, MA, USA, 2008.

[Mil08] D. L. Mills. Network Time Synchronization Bibliography. 2008.

[MMH05] Luis Sardinha Monteiro, Terry Moore, and Chris Hill. What is the accuracy of DGPS? *The Journal of Navigation*, 58(02):207–225, 2005.

[MNSH01] Lothar Merk, Martin S. Nichlous, Thomas Stober, and Uwe Hansmann. *Pervasive Computing Handbook*. Springer-Verlag Telos, 1 edition, 2001.

[Sch05] Hartmut Schmeck. Organic Computing – A New Vision for Distributed Embedded Systems. In *Proceedings of the 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'05)*, pages 201–203, 2005.

[Web09a] Web. data-quest GmbH: http://www.studip.de/. Web, 2009.

[Web09b] Web. Facebook, Inc.: http://www.facebook.com/. Web, 2009.

[Web09c] Web. Google Inc.: http://maps.google.com/. Web, 2009.

[Web09d] Web. Sevenload GmbH: http://www.sevenload.com/. Web, 2009.

[Web09e] Web. studiVZ Ltd..: http://www.studizv.net/. Web, 2009.

[Web09f] Web. Twitter Inc.: http://www.twitter.com/. Web, 2009.

[Web09g] Web. YouTube, LLC: http://www.youtube.com/. Web, 2009.

[Webll] Web. Gordon Bell's Homepage. http://research.microsoft.com/en-us/um/people/gbell/.

[Webwe] Web. Dann Elli's Homepage. http://www.ee.columbia.edu/ dpwe/.

[Webhu] Web. FFmpeg - audio/video conversion tool. http://ffmpeg.mplayerhq.hu/.

[Webom] Web. Trackstick II - telespial systems. http://www.trackstick.com/.

[Webet] Web. enkin for android. http://www.enkin.net.

[Webes] Web. Nokia N810 Technical Specification. http://www.nseries.com/nseries/.

[Webor] Web. Marvell unleashes 1GhZ XScale ARM processor. http://tamspalm.tamoggemon.com/2009/01/15/marvell-unleashes-1ghz-xscale-arm-processor/.

[WHHMS08] Michael Wittke, Martin Hoffmann, Jörg Hahner, and Christian Müller-Schloer. MIDSCA: Towards a Smart Camera Architecture of Mobile Internet Dvices. In *ICDSC08*, pages 1–10, 2008.

[WOL02] W. Wolf, B. Ozer, and T. Lv. Smart Cameras as Embedded Systems. *Computer*, 35(9):48–53, 2002.