

Geode: A Framework for Social and Context-Driven Browsing for Personal Multimedia

Stewart Greenhill and Svetha Venkatesh
Department of Computing, Curtin University of Technology
stewartg@cs.curtin.edu.au, svetha@cs.curtin.edu.au

ABSTRACT

We present a system that harvests readily available context information (GPS, bluetooth, user) when multiple media such as photos, video, audio, or activity streams (eg. from Twitter, Facebook, etc.) are acquired through cell phones and uses it for multimedia navigation, search and sharing. Separate context streams are recorded on the phone, and related to media captured (on the phone or other devices) based on recorded time. Our framework integrates and unifies all time-based media and uses contextual meta-data to construct novel, rich browsers, facilitating the sharing of both data and meta-data across users. This includes location, co-presence and activity, which are used in new ways for navigation and search in the aggregated media. By considering the meta-data tuple {media, place, time, activities, friends}, rich queries can be made by selecting subsets of available meta-data. Further, synchronous media streams can be played back in parallel, allowing of media aggregation in novel ways. Our implementation and experiments demonstrate the efficacy of this paradigm.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval; H.5.2 [Information Interfaces and Presentation]: User Interfaces

General Terms

Algorithms, Measurement, Design, Experimentation

Keywords

multimedia browser, personal media management, social context, photo, video, audio, filter, query, cluster.

1. INTRODUCTION

With the diversity in both the type of media, and the situations in which it is captured, comes the problems of

management and search. Media can be collected using different devices, such as video or phone cameras. These devices store data in different formats, and captured files must be offloaded from the devices into a file repository which needs to be organised in some way. This has led to the development of photo management applications (eg. PhotoMesa, ACDSee, ProShow, iPhoto and Picassa) which allow photo collections to be organised and navigated in different ways. Many such systems require meta-data to be added manually by the user.

In addition to personal search and retrieval, it is becoming increasingly important to be able to share material with others. This means that the media must be accessible not just to the creator, but to others as well. Sites like YouTube and Flickr are good at disseminating material to a wide audience, but they rely heavily on meta-data such as tags and descriptions for findability. These online sites tend to focus on just *one type of media*, so if someone has a variety of interests they end up with fragmented profiles and social networks on each site. In order to share with particular people (rather than with everyone), those people need to be on the site too. Each site only knows about one facet of our experience, so we lose the ability to find or make relationships between different types of media.

In this paper we explore the harvesting of “free” meta-data to span the gap of fragmentation both across media and users to create the next generation of “social multimedia browsers”. Free meta-data comes when media is collected on cell-phones and may include location based on cell tower ID (coarse) or GPS (fine resolution), and co-presence information, derivable from Bluetooth. Activity information comes from social applications like Twitter, Facebook and Gmail where people communicate their activity or status on-line and in real-time. Our work goes beyond using this information for geo-tagging and presence sharing, and investigates its potential for media navigation. Open problems include a) How can context be used for navigation and organisation? b) How can we integrate different media repositories for single and multiple users into one framework? c) How can we propagate context between users?

Addressing these open issues, this work describes “Geode”, a system for organising and sharing personal media. It consists of two applications: a logger and a browser. The logger runs on a cell phone, collecting contextual information about a users activity. The browser runs on a personal computer, allowing media to be organised and navigated. We provide a framework to handle different types of media such as photos, audio, video, location, and activity. We harvest time

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MM’08, October 27–November 1, 2008, Vancouver, BC, Canada.
Copyright 2008 ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

meta-data across media either directly (via EXIF meta-data of photos, or activity time-stamps for Twitter), or indirectly (file time for video and audio), and create a coherent temporal stream. All the collected meta-data is synchronised with media, integrating multiple users and media in one framework and we exploit relationships between different media types to improve search and navigation. Our framework provides a rich application for personal media management that:

1. Has a consistent approach to handling all time-based media.
2. Uses meta-data collected automatically from the users cell-phone to enrich the interpretation of media. This includes location, co-presence and activity, which are used in novel ways for navigation and search.
3. Allows media to be aggregated, so that synchronous media streams can be played back in parallel.
4. Exploits opportunities for fusion between media streams. For example, spatial and temporal patterns of media distribution can be seen as evidence of places or events, which are significant for browsing and navigation.
5. Delivers a rich browsing experience.
6. Supports sharing of both data and meta-data between users.

We implement the logger and browser and demonstrate our results using a centralised store of data from 8 users. We used Nokia GPS phones to collect context. Phones and other devices were used to collect photos, videos, audio and activity (via Twitter). Bluetooth is used to indicate user co-presence. Both location and co-presence meta-data are propagated across users. Clustering is used to identify places (using GPS and user) and events (using place, users and time). These events and places are used to deliver rich browsing experiences. Our results demonstrate the usefulness and power of this paradigm. We note that although we have implemented a centralised version, the framework can be implemented in a distributed way, allowing each user to maintain their repository separately, yet giving them the power of shared media and context.

The significance of our approach lies in the richness of browsing that is achievable. By considering the meta-data tuple {media, place, time, activities, friends}, rich queries can be made by selecting subsets of available meta-data. Examples include;

1. What did my friends do on the weekend? (User, Time → Place, Media, Activity)
2. What do people do in place X? (Place → Media, Activity)
3. What does this place sound like? (Place → Media, Place)
4. Where do people do activity A? (Activity → Place)

The novelty of the framework lies in departing from the paradigm in browsing that relies on browsers dealing with *single media* at a time for *single users*. Our cohesive framework, integrating multiple media types and multiple users,

transcends the gap required to deliver shared multi-media browsing experiences by effectively harvesting context. Further, the joint use of shared context and multiple media across users enables the synchronisation of media from multiple users to give composite perspective of places and events. For example, audio from one user can be automatically fused with photos from another user to produce an ad-hoc video.

2. RELATED WORK

Digital cameras make it easier than ever to capture moments of our lives. [7] examined the different requirements of users, and identified applications in four classes: archiving, sending, co-present sharing, and remote sharing. Digital photo management applications fall in the first category, being systems for retrieving images from storage. [18] found that the two most important features of these applications are sorting into chronological order, and displaying a large number of thumbnails at once. People are familiar with their own photographs, so this is usually enough to allow them to find what they are looking for. The availability of indexing and text-based search did not motivate users to use these features.

This situation changes when photos are shared. [14] explored the behaviours that have developed around on-line photo sharing sites like Flickr. They identify two main groups: the “Kodak Culture” who share narratives in the home around events like birthdays, and the “Snaps” who frequently share with others via on-line forums. In both camps, chronology was found to be a key strategy for organising photos, though most organised their material by event. Snaps regularly tagged their photos, seeing this as a form of social interaction.

With the burgeoning of on-line social media and “web 2.0” user-contributed content is accompanied by meta-data such as tags, titles and descriptions, all of which are essential for items to be discoverable by other users. The incentive to add this data is the potential for increased “hits” and social interactions around the contributed media. In August 2006, Flickr introduced geo-tagging and YouTube followed suit in June 2007. This means that appropriately tagged media can be located via temporal or spatial criteria, although there is currently no way to perform searches across multiple sites.

Research into media management applications has focused on ways to filter, relate and browse images based on criteria that can be deduced from data. PhotoMesa [2] provides a zoomable interface to large collections of photographs. It clusters photographs using an existing directory hierarchy, and uses quantum treemaps to tile images for display in a way that visually preserves the grouping. Other applications have focused on extracting clusters from variations in time, location, or image content. If a cluster can be summarised by a representative subset of images this helps reduce the clutter in displays of large sets of images.

Different methods are used to cluster based on time. Looking at the duration between images, these attempt to identify inter-event gaps using measures like frequency distribution [8], log gap average [16] or similarity [3]. Other factors can be included in the clustering, such as location [15], and image content [13].

Media browsers can use temporal clusters in different ways. The Calendar Browser [8] implements hierarchical clustering, which allows users to progressively “drill down” into sub-events of finer resolution via a tree view or calendar

view. PhotoCompas [15] extends this approach to include hierarchical location clusters which can be browsed on a map, or via automatically generated place names. The PhotoTOC [16] browser shows the complete set of images in a “detail” pane, and a set of clusters in an “overview” pane. Clicking on the overview scrolls the detail pane to the relevant image. It chooses photos to represent clusters using the KL divergence between the histograms of each image in the cluster and the averaged histogram over all images in the cluster. [20] uses a similar event clustering algorithm but adds a PhotoMesa-inspired treemap view and use torso analysis to help identify photos containing the same people based on their clothing. MediAssist [10] uses location and content-based ranking to find images similar to a selected image. Socio-Graph [1] is a multi-user browser which infers social context from location logs and persistent audio records. In addition to temporal events and 3D spatial browsing, it allows selection of images based on social relationship.

A common theme for many of these applications is to simplify the “top-down” search process: to make it easier to find a particular image starting from a large collection. As was identified by [18], this is important for users *locating items in their own collections of media*. While acknowledging this requirement, we explore other ways of browsing that suit multi-user image archives. In this work, we are also interested in implementing “bottom-up” navigation of media collections. Given an image which may be discovered serendipitously, we want to be able to find related images that can serve to reveal the context around that image (eg. What was the event? Who else was there, and what did they see?). We aim to enable a *web of media* which can be navigated like a web of documents. Clusters, however they are deduced, provide links between an image and other images which can also be used for navigating a bottom-up view of a repository.

Many on-line applications exist around data collected on mobile phones. *Moblogging* is the practice of blogging via mobile phone. Moblogs tend to focus around images, as these are easier to capture than text. Applications like ShoZu (shozu.com) can be used to automatically upload media captured on phones to sharing sites (eg. Flickr, YouTube), or to blogs (eg. Blogger, LiveJournal, Wordpress). When ShoZu detects a new photo or video it prompts for descriptions and tags before uploading to a selected web-site; it can also include geo-tags if the phone is GPS-enabled. Some systems provide novel ways to organise and browse media. Glogger [12] implements storyboards and panoramas, and MobShare [19] includes parallel time-line views. Mobile applications focus on immediacy, uploading data directly from the phone over the cellular network. Video streaming from phones is also possible using applications like Qik (qik.com).

The *ContextPhone* [17] platform senses and responds to the user’s activity. The system measures location (via cell tower id, or GPS), nearby bluetooth devices, communication (eg. phone calls, SMS), and application usage generating log files that can be archived over the cellular network. A set of customised applications runs on the phone to adaptively use this information. For example, ContextContacts replaces the normal Contacts application with a version that lists additional information such as the contact’s location, the number of friends and other people nearby, and whether the phone speaker is on or off. This is intended to help the user

decide whether to call now or later.

ContextPhone and runs on Symbian V1 and V2 phones, though it does not support modern V3 phones. The system has been developed commercially by Jaiku (jaiku.com) which provides “presence sharing” services. Users can share availability (based on ring profile), location (based on cell tower ID), co-presence (based on bluetooth devices) and calendar events with the public, or with selected contacts. Users can label their location, which allows cell tower IDs to be collaboratively mapped to place names. Users can also declare what they are doing by posting “jaikus” which are received by their contacts like an instant message.

Several research groups have used the ContextPhone platform. The Reality Mining project at MIT Media Lab used context logs to measure the strength and dynamics of social networks [5]. Communication, location and proximity were used to deduce the type and strength of social ties. The Garage Cinema Group at University of California Berkeley used context data to facilitate photo sharing [4]. The MMM2 system uses bluetooth proximity information to help a user decide who a photo should be shared with. Merkitity (meaning.3xi.org) allows context data (location, bluetooth environment) to be added as tags when images are uploaded to Flickr.

Whilst applications exist for streaming and uploading media from phones to the web, their use of context is limited to what can be done at the time of upload (eg. adding tags, or deciding who to share with). In contrast, our application stores context streams so they can be used by our browser to improve search and navigation.

3. DATA COLLECTION

This work aims to support the collection of context as well as media. Many devices exist for media capture: Digital still cameras, video cameras, and audio recorders. Modern camera phones offer multimedia recording facilities, although this is often targeted to low bit-rate encodings suitable for cellular networks. Multimedia phones like the Nokia N95 have high quality cameras and optics, and are capable of high quality video recording. For example, the Nokia N95 has a 5 megapixel camera and can record 640x480 resolution video at 30fps, with audio up to 48Khz. Phones are increasingly fitted with a range of sensors like GPS and bluetooth, and their programmability makes them an ideal device for both context sensing and media capture.

Our aim is to provide a platform that works across a *wide range of media capture devices*. We use phones for recording context, but we also allow devices like video cameras and digital still cameras. This means that we can’t just embed context tags at capture time. We need to record separate context streams and relate context to media based on recorded time.

For this study we targeted the Nokia 6110 phone, a mid-range multimedia phone running the Symbian S60 V3 operating system. These phones have a 2 megapixel camera, bluetooth, an inbuilt GPS and navigation application. Each phone was installed with a 4Gb micro-SD card for media recording. We wrote a J2ME MIDP 2.0 logging application (called “*rover*”) which runs on the phone to record user activity.

The *rover* application runs as a background task on the phone, collecting GPS positions, bluetooth contacts, calendar events and activity reports. The main limitation is

power consumption: with the GPS active the phone consumes about 120mA which gives less than 7 hours battery life. Rather than sampling continuously, we record position every 10 minutes, switching the GPS off between samples. A “fast sampling” option records position continuously, but reverts to “slow sampling” if a GPS fix cannot be obtained (eg. if the user is inside a building). In this way it is usually possible to run the logger continuously, recharging the phone once a day.

An important parameter for this approach is the GPS “time-out”: how long we attempt to sample position before giving up. When users are outdoors the ephemeris data for all satellites is known and a “hot start” takes only a second or two. Whilst indoors, satellites are not visible so GPS positioning is usually not possible. However, we need to allow enough time to facilitate a “warm” or “cold start” when we are next outdoors. Setting the time-out too short means that a “cold start” will fail, but longer times cost more battery life. In practice we found about 2 minutes to be sufficient. If the user has a GPRS facility with their phone plan, Assisted GPS (A-GPS) dramatically reduces the start time but this is at the cost of 5 to 10kb of traffic every half hour or so.

Bluetooth scanning is comparatively simple. A bluetooth device discovery enquiry uses 30mA for around 15 seconds, so its impact on overall power consumption is small. For each discovered device we log address, name, and device class. We scan for bluetooth devices every 2 minutes; these scans are synchronised with the GPS samples. We call the combination of location and co-presence the users’ *physical context*. This is sampled passively (location every 10 minutes, co-presence every 2 minutes) but is also actively sampled when events such as media creation are detected.

We explored the use of the inbuilt mobile media API (JSR-135, aka MMAPI) but found it too restrictive for use. When recording stream-based media like audio and video, the Nokia implementation records first to the phone’s internal memory, only copying it to the final destination once the recording has finished. This limits the size of collected media to about 20-30Mb, regardless of the amount of storage available on the device. In addition, the MMAPI does not allow the use of higher quality media formats. For example, the inbuilt camera application allows video to be recorded in MP4 format, but the MMAPI only supports the inferior quality 3GPP format.

Given these constraints, we decided that the users should capture media via the inbuilt applications, rather than mediating the capture via **rover**. Instead **rover** periodically monitors the file system to detect new media files. When it discovers a new media object, it activates the GPS to sample position, and initiates a bluetooth enquiry. The standard applications include a camera which captures still images up to 1600x1200 pixels. It records MP4 video for up to 1 hour at a time at 320x240. The voice recorder supports 16-bit 8KHz recordings with durations up to 1 hour. We also ran this application on Nokia N95 phones, which provide similar functionality but capture media with significantly better quality.

Activity streams consist of short status reports, originating from social applications like Twitter and Facebook. Twitter (twitter.com) invites users to keep in touch with friends by frequently answer the question “What are you doing?”. Depending on the community, the main uses of

Twitter are: daily chatter, conversations, sharing information/URLs, and reporting news [9]. We don’t attempt to distinguish these uses, but suggest that posts from a mobile phone are more likely to reflect current activity. **rover** includes a simple Twitter client, allowing users to say what they are doing at any time. It samples physical context for each status report, recording each “tweet” to a log file as well as optionally posting to **twitter.com**. In addition, we can import Twitter streams from the web, although physical context may not be available unless tweets were posted via **rover**.

The result of the logging process is that the user’s position is sampled sparsely, but the locations of media objects are still known with reasonable precision.

To collect data from the phones, we simply copied the relevant media directories to a host PC via USB. With appropriate infrastructure, this data could be automatically retrieved wirelessly. Some of our devices (ie. the N95) had WiFi capability, but the majority had only bluetooth. Depending on data volume, it is possible that a daily connection with a host PC might be sufficient to transfer collected media.

4. DATA ANALYSIS

Geode users import data into a *media repository* by nominating folders containing media files. The system recursively scans folders, extracting basic data from the files. For images it gathers the size, creation time, and thumbnail, which are normally available in EXIF data. For video and audio files there are no standards for embedding meta-data. Geode computes start time and duration from the file modification time and media stream duration. If necessary, time corrections can be applied to the file times or EXIF times for particular media folders. This allows us to handle errors in time setting, or offsets due to factors like timezone or daylight savings. At the end of this process, Geode knows the temporal extent of every media object. In addition, Geode scans all **rover** log files to extract position, co-presence and activity information. Activities are treated as media objects (time-stamped text descriptions). All media objects (photo, video, audio, activity) are then compared to the GPS stream to attempt to geo-locate each object. Where possible, location and co-presence is propagated between users. This helps deal with the vagaries of wireless reception, which sometimes result in GPS and bluetooth visibility being different for users at the same location. It also allows location to be available to users without GPS (eg. if they are with a user with a GPS phone).

For each user, Geode builds an index to media repository. At the simplest level this can be used to browse and navigate the media, as can be done with applications like Picassa and iPhoto. Geode also attempts to derive additional layers of meta-data by clustering over time and location across the basic media streams. When multiple users exist, each repository contains separate streams of contextual information. Ultimately, this information will be shared between distributed Geode instances via a network, but for the moment we are exploring the possibilities of this scenario with data hosted on a single machine.

The system uses clustering to attempt to identify *events* and *places*. *Events* are deduced by clustering over the media creation times. Often photos appear in bursts, as the user captures different aspects of a scene over time. We imple-

mented hierarchical event clustering based on the method of [8]. Starting with an initial time difference of 4 hours we derive an initial set of clusters which is then further refined by searching for outliers. We compute the time differences between all photos within a cluster then find the boundary points $Q1$ and $Q3$ for the first and third quartiles of the distribution. An outlier is any difference greater than $Q3 + 2.5 * (Q3 - Q1)$, where 2.5 is an empirically selected value. Outliers are used hierarchically partition clusters into sub-clusters.

Events are used by the system to simplify summaries of user activity. A cluster of images can be replaced with a single image which can optionally be expanded to show the original set. Events are also used as a “coarse” scale at which to navigate a user’s media stream. Lastly, events are used to find objects related to a particular media object.

Places are regions where users spend time or perform activities. The system uses them in two ways. Firstly, if a media object can be associated with a place, it can easily be linked to other objects at that place. In the browser, this corresponds to the operation “Find Other Objects Here”. In this sense it is important that the place is a symbolic object rather than an arbitrary positioning of an area on a map, because we want this operation to be free of user interaction.

Secondly, places are used to simplify the display of objects on maps. If we attempt to display all objects at their location the display quickly becomes cluttered, and objects become obscured by other objects at the same location. Instead, we display a marker indicating the number of objects at the place. The user can “mouse over” the marker to see a representative set of these objects. This clustering can be done hierarchically with increasing granularity so that low-level clusters merge as the user zooms out on the map display.

Geode implements spatial clustering using DBSCAN[6], a density-based clustering algorithm. Advantages of DBSCAN are that it (a) discovers clusters of arbitrary shape, (b) requires minimal assumptions about the data, and (c) works efficiently on large data sets. DBSCAN requires the specification of two parameters: ϵ , the threshold for neighbourhood reachability, and D , the minimum number of points per cluster. We derive places by clustering over the location of media collected by all users. We found that $\epsilon = 100m$ works well for our data, but note that this assumes “urban-scale” activity which might not be appropriate in all scenarios (eg. world travel). These situations would require either a hierarchical approach (clustering at multiple scales of ϵ), or use of variable-density [11].

Events and *Places* offer a wealth of browsing possibilities. For example:

1. *What did my friends do on the weekend?* Given a time, and a set of people, find the places they visited and any associated media.
2. *What do people do in place X?* Given a place, find the media and activities at this place.
3. *What does this place {look,sound} like?* Given a place, find the corresponding images, video, or audio objects.
4. *Where do people do activity X?* Given an activity, find the places where people recorded this activity (eg. having coffee).

5. *Find other images here.* Given an image with a known place, find other images of the same place.
6. *What happens here?* Given an image, find activities that occur at that place.
7. *Find who was here.* Given an image, find the people who were co-present at the time.
8. *Find images of X from others.* Given a person, find all images from other people for which they were co-present. Often this will include images of that person.

5. EXPERIMENTS

We conducted a trial of the system with seven participants. Each participant carried a bluetooth-enabled GPS phone: 6 Nokia 6110s and 1 Nokia N95. Four phones were used online with GPRS, one phone was used online without GPRS, and two phones were used off-line (ie. without a cellular network connection). At the time of writing durations range between 1 and 5 months. Over this time, participants collected 2270 images, 226 videos, and 56 audio files. Other media capture devices were used including digital cameras, video cameras, and audio recorders. Each user contributed their media files to a central file store managed by Geode.

See section 3 for details of the data collection, and section 4 for analysis.

We developed a browser in parallel with the data collection and analysis. Our aim was to have the browsing and visualisation evolve organically around interactions and features we saw in the data.

5.1 Implementation

Geode is implemented in Java and uses the Java Media Framework (JMF) for media access and display. JMF is a flexible multimedia platform, but has been rightly criticised for its narrow range of supported media formats. A small set of formats is implemented in Java, but it relies on “performance packs” to use native codecs, giving poor portability across platforms. We use the FOBS plugins (fobs.sf.net) which address this problem by implementing a JMF interface to the ffmpeg platform (ffmpeg.mplayerhq.hu), supporting a diverse range of codecs, formats and media transport across many platforms (Windows, Macintosh, Unix).

Media repositories are stored separate from the source media files. For each user, the repository includes a media index, thumbnail database (rendered or extracted from original files), a location log, a log of bluetooth contacts, and a temporal and spatial cluster index. These are stored using a mixture of plain-text and binary files, but could easily be stored in a relational database if required. The system works well for moderately sized repositories (eg. the author’s own media library of around 15,000 objects: 20Gb photos, and 50Gb video).

The map viewer retrieves maps from tile servers over HTTP. Maps are available from many providers including NASA, Yahoo, Microsoft and Google. Geode maintains its own persistent local cache of map tiles, so once the system has been primed the network load for map browsing is very low. The cache also allows the map browser to be used off-line.

6. RESULTS

This section describes the design of the browser and illustrates novel features with examples from our data. This

includes synchronised playback (6.3), event browsing (6.4) and contextual navigation (6.5).




6.1 Browser

The browser fulfils several roles: It allows a user to browse and retrieve their own media via meaningful concepts like *time*, *event* and *place*. It provides a visual diary that relates media and activities. It allows exploration of other users' media. It discovers and highlights relationships with other users. Lastly, it allows users to explore events by browsing multiple media streams synchronously.

Figure 1 shows an example of the user interface, which consists of several interlinked displays. The display is divided into regions using resizable dividers. The right-hand side is for *query and selection* and includes *filters* and the “*time-strip*”. The left-hand side is the *player*, including the *media viewer* and *time-line*.

It works like this:

1. To search for particular events, users can specify constraints via filters on time, place, people, and media.
2. The filters select a set of media objects which are presented in the time-strip. This is a chronologically ordered sequence of thumbnail images, intermixed with activity descriptions. We can choose to see all media, or see clusters of media corresponding to events. Items can be selected from the time-strip for display in the media viewer.
3. The media viewer renders multiple media objects, each in its own frame of a tiled display. Each frame provides navigation to contextually related media, and can be used to explore interrelated items.
4. The time-line allows multiple media streams to be played back synchronously. This means that an event can be observed from multiple perspectives if there were multiple observers present.

The “*time-strip*” component (Figure 1, bottom right) summarises all existing media objects using small low-resolution thumbnail images¹. Media are ordered chronologically, with a header showing the date for each new day. For visual media (images, video) the thumbnail is extracted from the media file. For text objects, the text is rendered in-line. Other objects (eg. audio) are shown using icons. Clicking in the time-strip toggles selection of the object for media viewing. Figure 5 shows a time-strip with examples of photos, audio and video clips interspersed with tweets from users. Icons superimposed over the thumbnails indicate media type: video , audio , and geo-tag . To produce a summary or narrative view, time-strips can optionally be displayed with event clustering (see 6.4).

The *filters* component (Figure 1, top right) provides several different ways of filtering media for selection. This includes media type, creator, time and place (see 6.2). The results of multiple filters are applied the time-strip display, and to media displayed for contextual navigation (see 6.5). The filters pane may optionally be hidden, maximising space available for the time-strip.

¹Time-strips extend the traditional “filmstrip” metaphor by incorporating multiple visual and non-visual media in one chronological display.

The *media viewer* (Figure 1, top left) renders one or more media objects in a tiled display. The tiling adapts automatically as the the number of selected objects changes. Time-based media (audio, video) are rendered using JMF players. Multiple video/audio streams can be rendered in parallel. The system handles this situation in one of two ways. In *free* mode each object has an independent time-base allowing non-synchronous objects to be played in parallel. This is useful for producing photo or video montages. In *linked* mode, the time-bases of displayed objects are synchronised to a master clock which is controlled by absolute time.

The *time-line* (Figure 1, bottom left) shows the times at which media exist in the repository. This can be displayed either in *absolute time* or *relative time* (a representation that preserves duration of media, but removes the gaps where no media exist). A cursor in the time-line view is synchronised with the playback time in the media view.

6.2 Filters

A *filter* is a Geode component that selects media based on criteria that the user defines. Filters are arranged in a sequence with the first taking the set of all media as its input, and the last passing its output to the time-strip display (see Figure 2(c)). Some filters (eg. time, place) display their input in different ways as part of the user interaction. These filters tend to be placed downstream, although any ordering is allowed.

- *Media* filter selects media by criteria such as type (photo, audio, video, activity) and constraints like duration and keywords in text. Optionally, objects in the same event cluster can be selected too.
- *People* filter selects media created by a given set of people.
- *Time* filter selects media by creation time. A calendar can be used to select specific days of interest. It displays one thumbnail image from its input for each day (see Figure 2(a)).
- *Place* filter selects media by location. Clusters of media objects are displayed on a map. The user can select groups of clusters using the mouse, or by defining geometric regions. In Figure 2(b), selected clusters are shown in green.

Note that the algorithm used to cluster media for map display is different from the place clustering algorithm mentioned in 4. This is because we want to show the actual distribution of samples in a way that quickly adapts with changes in zoom level, so a quick but granular display is preferred. Each dot in the map display is labelled with the number of objects at that place.

These filters can be combined to answer a variety of questions. For example:

What did my friends do on the weekend. Select friends and time of interest using the the *People* and *Time* filters. This shows all of the relevant activity in the time-strip. The *Place* filter shows the corresponding locations, and can be used to focus on media related to a particular location.

Where do people go for coffee? Use the *Media* filter to select just Activity, then enter the keyword “coffee”. Select “Include Event”. This selects all media in events that include the activity “coffee”. These locations appear on the map in the Place filter.

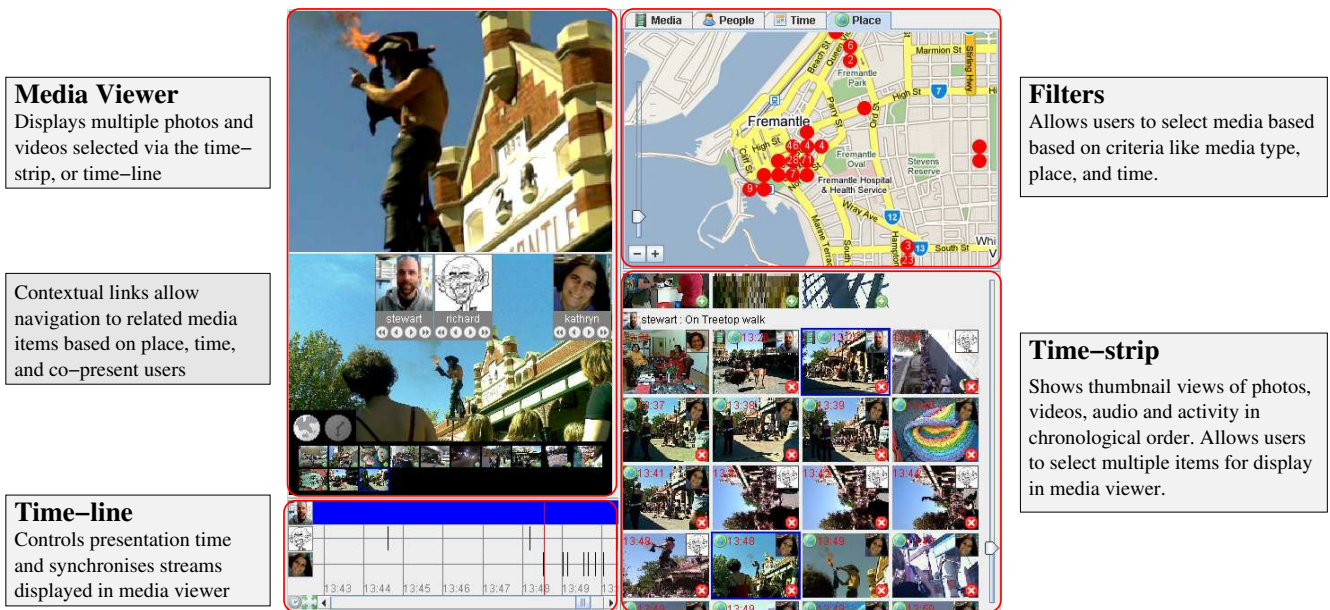


Figure 1: Browser layout showing spatial filter (top right), time-strip (bottom right), media view (top left) and time-line (bottom left). The display shows two perspectives of an event taken by different observers. The images from video and photo streams are synchronised via the media view time-line.

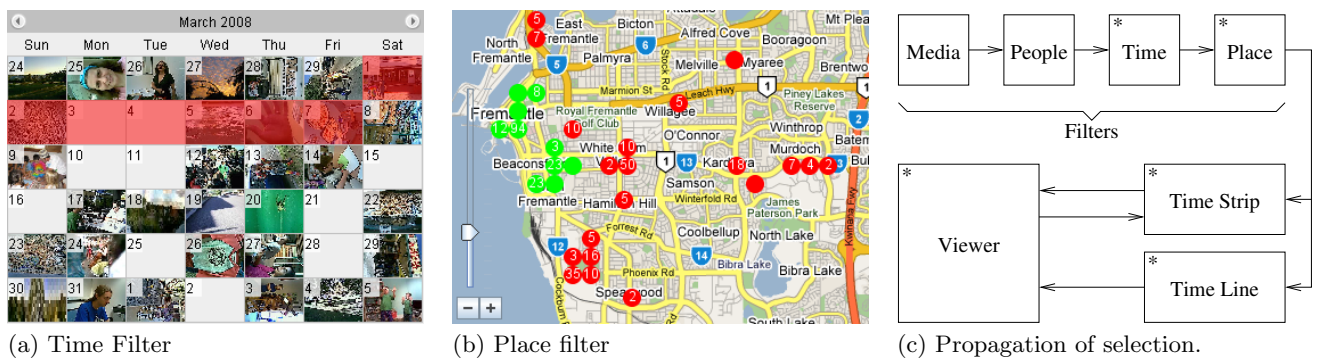


Figure 2: (a) and (b) show examples of Geode filter interfaces. (c) Shows the propagation of selection through filters to viewer. Components marked “*” have visual interfaces that present their inputs to the user.

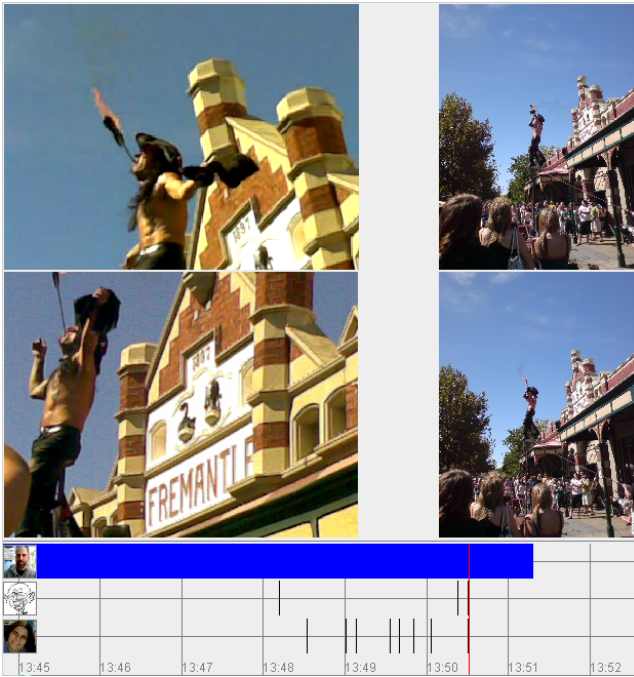


Figure 3: Synchronised media display, with contributions from three users: user 1 contributed the video (top left), user 2 contributed the photo (bottom left) and user 3 contributed two photos (right).

6.3 Synchronised Playback

In the Geode media viewer, the time-line works like the familiar *seek bar* in media players. However, note that the Geode’s time-line represents a *global timebase* across the entire media repository. Temporal extents are computed by Geode in the initial analysis phase so that media objects can be related in a common time frame.

When we move the cursor in the time-line, Geode dynamically creates JMF players for media objects, synchronising their time-bases as required. The media view includes embedded players for stream-based media, and rendered images for other media like photos. By synchronising all media, we get the impression of a “compound stream”, composed of contributions of multiple users.

One issue in this type of display is how we deal with objects that have no inherent duration (eg. photos). We need to allow photos to persist for a few seconds but also need to minimise the number of re-tiling operations in the media player; both rapid flickering and retiling can be disturbing. One approach is to allow just one image per stream per user to be displayed, but to have each image persist until the next one is available. Another option is to display each image in its own frame for a fixed time-window around its creation time. The latter approach can increase the level of visual concurrency at busy times, but results in a less cluttered display between photo events.

Figure 3 depicts notorious sword-swallower Matty Blade busking in Fremantle’s Henderson Street mall. The images are from three separate users, synchronised via Geode’s media viewer. The top left frame is a video stream, the other frames represent photos from two other users. The time-

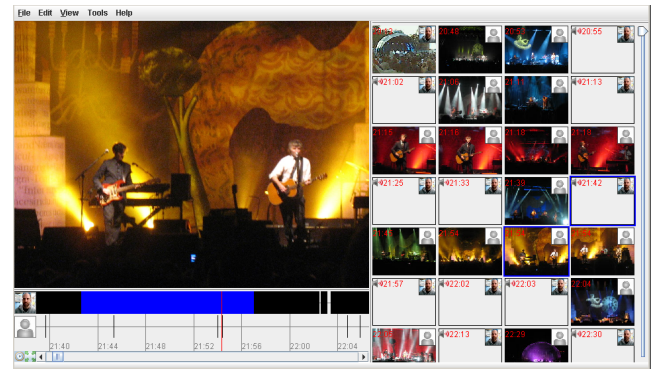


Figure 4: Ad-hoc media fusion in the Geode media viewer. Audio stream is mixed with another users’ photo-stream using synchronised playback.

line (below) shows the overlapping extents of the video and photo times. Note the absolute time on the time-base.

One of the users (richard) was not participating in our trial, so we did not have context logs from his phone. However, he posted his photos to his Flickr account, which we imported into Geode. This procedure involved 1) creating a new user and importing his avatar image, 2) downloading the original photos from Flickr into the repository, and 3) defining the bluetooth address of his phone, and 4) defining a time offset for the imported images. This is sufficient to have all his photos geo-tagged via propagation of location information, and synchronisable to media of other users.

One exciting possibility enabled by Geode is the ad-hoc fusion of media based on synchronism (ie. events occurring at the same time). In this example, an audio stream recorded at a concert (Crowded House, November 16 2007) is synchronised with another users photos to create an ad-hoc audio-video presentation. The audio consists of discrete clips of roughly 10 minutes each. The images were imported from a Flickr photo-stream, and are synchronised with the audio via the Geode media viewer. Both streams are seekable in parallel using the time-line without any special pre-processing.

6.4 Event Browsing



The time-strip can be used to browse many images at once, but if we want a concise summary of activity over a day we can use event clustering to simplify the display. In “event” mode, media in the same temporal cluster is reduced to one representative image. Controls on the thumbnails allow the clusters to be opened  and closed .

Figure 5 shows how this works. On the left is the raw time-strip. On the right is the event-clustered time-strip. On this day, 123 photos were collected by 5 users. This reduces to just 8 events; User1: shopping, child1 birthday party, Earth Hour celebrations, User2: child2 birthday party, User3: adult party, User4: concert, User5: child3 swimming lessons, playing at the river. One event is shown with the cluster expanded (including one photo and two videos). These can be identified by the “close” control in the bottom right corner.

6.5 Context-based Navigation



Figure 5: Two modes of the time-strip : sequential and event display.

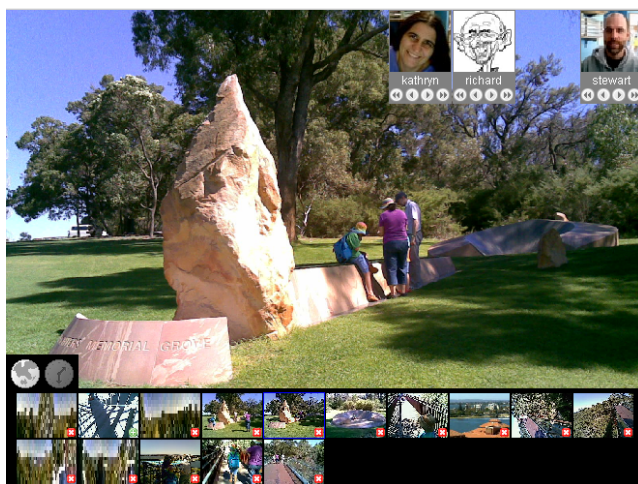


Figure 6: Context display showing the media producer (top right) and other people present. The filmstrip at the bottom shows related images.

Contextual information allows us to answer many meaningful questions about an image: where is this place, who else was here, and what did they see? Section 4 outlines examples of relationships that can be established from available data. Geode exposes these contextual queries via the image frame in the media view. When the user moves the mouse over an image frame, contextual data overlays the image.

Figure 6 shows an example of the context overlay. The “postage stamp” image in the top right indicates the user who created the image. The other avatars show other users that were present, based on either their GPS location, or bluetooth co-presence. For each user present, the associated transport controls jump to the previous or next image by that user, or the previous or next cluster of images.

The filmstrip at the bottom of the image indicates images or events occurring at the same place. Clicking on an image replaces the current image tile with the linked image. Clusters can be expanded or collapsed by clicking the open and close icons in the corner of each thumbnail.

The filmstrip can also be used to display other relationships. For example: other events happening around the same time as the image (ie. at different places). Clicking on any of the avatars shows the stream of events for that user around that time. This shows us what was happening for other users at that time.

This kind of browsing can often reveal common interests. For example, while browsing your photos of an event, you may discover other users who were at the same event, or at other events at the same venue. By automating the discovery and presentation of contextual relationships, we enable both purposeful and serendipitous navigation of image collections.

7. CONCLUSION AND FUTURE WORK

We have presented a system that collects contextual information via mobile phones, and uses it for multimedia navigation and sharing. We provide search via spatial and temporal filters, which are used together with a chronologi-

cal time-strip view for the traditional “top-down” navigation of media collections. We also provide contextual navigation using co-presence, spatial and temporal clustering. This enables an alternative “bottom-up” navigation that reveals relationships between images as well as exposing the context of an image (eg. what was the event? who else was there, and what did they see?).

This work serves to enhance and share personal media collections. Some of the context analysis algorithms could be improved or extended. For example, we could implement joint space-time clustering, or image similarity based on content features. Other sources of context could be added (eg. calendar or blog entries), and activity streams could be analysed for semantics using topic modelling. Future work is also expected to focus on models for sharing context and media in a distributed application, and to improve the collection and interpretation of contextual data.

8. REFERENCES

- [1] B. Adams, D.Q. Phung, and S. Venkatesh. Extraction of social context and application to personal multimedia exploration. In *ACM Int. Conference on Multimedia, Santa Barbara, USA*, Oct. 2006.
- [2] Benjamin B. Bederson. Photomesa: a zoomable image browser using quantum treemaps and bubblemaps. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 71–80, New York, NY, USA, 2001. ACM.
- [3] Matthew Cooper, Jonathan Foote, Andreas Girgensohn, and Lynn Wilcox. Temporal event clustering for digital photo collections. *ACM Trans. Multimedia Comput. Commun. Appl.*, 1(3):269–288, 2005.
- [4] Marc Davis, Nancy Van House, Jeffrey Towle, Simon King, Shane Ahern, Carrie Burgener, Dan Perkel, Megan Finn, Vijay Viswanathan, and Matthew Rothenberg. MMM2: mobile media metadata for media sharing. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1335–1338, New York, NY, USA, 2005. ACM.
- [5] N Eagle, A Pentland, and D Lazer. Inferring social network structure using mobile phone data, 2007.
- [6] M. Ester, H-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1994.
- [7] David Frohlich, Allan Kuchinsky, Celine Pering, Abbe Don, and Steven Ariss. Requirements for photoware. In *CSCW '02: Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 166–175, New York, NY, USA, 2002. ACM.
- [8] Adrian Graham, Hector Garcia-Molina, Andreas Paepcke, and Terry Winograd. Time as essence for photo browsing through personal digital libraries. In *JCDL '02: Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*, pages 326–335, New York, NY, USA, 2002. ACM.
- [9] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter: understanding microblogging usage and communities. In *WebKDD/SNA-KDD '07: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65, New York, NY, USA, 2007. ACM.
- [10] N O'Hare G Jones, C Gurrin, and A F Smeaton. Combination of content analysis and context features for digital photograph retrieval. In *in 2nd IEE European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies*, 2005.
- [11] P Liu, D Zhou, and N Wu. Vdbscan: Varied density based spatial clustering of applications with noise. In *2007 International Conference on Service Systems and Service Management*, 2007.
- [12] S Mann, J Fung, and Raymond Lo. Cyborglogging with camera phones : Steps toward equeveillance. In *ACM Multimedia 2006, 23-27 October, Santa Barbara, USA*, 2006.
- [13] Tao Mei, Bin Wang, Xian sheng Hua, He qin Zhou, and Shipeng Li. Probabilistic multimodality fusion for event based home photo clustering. *ICME*, 0:1757–1760, 2006.
- [14] Andrew D. Miller and W. Keith Edwards. Give and take: a study of consumer photo-sharing culture and practice. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 347–356, New York, NY, USA, 2007. ACM.
- [15] Mor Naaman, Yee Jiun Song, Andreas Paepcke, and Hector Garcia-Molina. Automatic organization for digital photographs with geographic coordinates. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pages 53–62, New York, NY, USA, 2004. ACM.
- [16] John C. Platt, Mary Czerwinski, and Brent A. Field. Photoc: Automatic clustering for browsing personal photographs, 2002.
- [17] Mika Raento, Antti Oulasvirta, Renaud Petit, and Hannu Toivonen. Contextphone: A prototyping platform for context-aware mobile applications. *IEEE Pervasive Computing*, 4(2):51–59, 2005.
- [18] Kerry Rodden and Kenneth R. Wood. How do people manage their digital photographs? In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 409–416, New York, NY, USA, 2003. ACM.
- [19] Risto Sarvas, Mikko Viikari, Juha Pesonen, and Hanno Nevanlinna. Mobshare: controlled and immediate sharing of mobile images. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 724–731, New York, NY, USA, 2004. ACM.
- [20] Bongwon Suh and Benjamin B. Bederson. Semi-automatic image annotation using event and torso identification. Technical Report Tech Report HCIL-2004-15, Computer Science Department, University of Maryland, College Park, MD, 2004.