Virtual Observers in a Mobile Surveillance System

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

ABSTRACT

Conventional wide-area video surveillance systems use a network of fixed cameras positioned close to locations of interest. We describe an alternative and flexible approach to wide area surveillance based on observation streams collected from mobile cameras mounted on buses. We allow a "virtual observer" to be placed anywhere within the space covered by the sensor network, and reconstruct the scene at these arbitrary points. Use of such imagery is challenging because mobile cameras have variable position and orientation, and sample a large spatial area but at low temporal resolution. Additionally, the views of any particular place are distributed across many different video streams. Addressing this problem, we present a system in which views from an arbitrary perspective can be constructed by indexing, organising, and transforming images collected from multiple streams acquired from a network of mobile cameras. Our system supports retrieval of raw images based on constraints of space, time, and geometry (eg. visibility of landmarks). It also allows the synthesis of wide-angle panoramic views in situations where the camera motion produces suitable sampling of the scene and metaphors for query and presentation that overcome the complexity of the data.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.4.5 [**Image Processing and Computer Vision**]: Reconstruction

General Terms

Algorithms, Measurement

Keywords

observation systems, mobile surveillance, video indexing, scene reconstruction, panorama, virtual observer, spatial query, visibility query.

MM'06, October 23–27, 2006, Santa Barbara, California, USA.
Copyright 2006 ACM 1-59593-447-2/06/0010 ...$5.00.

1. INTRODUCTION

Conventional wide-area video surveillance systems use a network of fixed cameras positioned close to locations of interest. With the inclusion of forward facing cameras on buses (current installed systems include: London and Los Angeles transport systems), government agencies may seek to integrate this roaming video surveillance network into the high-level strategic surveillance and security needs of the city. Mobile surveillance on buses assists in the battle against street crime and antisocial behaviour, in the investigation of traffic accidents, in investigating activities of people and with numerous other events that impact government operations. For example in the London bombings of July 7 2005, law enforcement authorities wanted to quickly access the footage of an area around the bombing and the only footage available was from the bus fleet. There are no systems currently able to effectively address such issues in wide area investigations. A surveillance initiative involving a fleet of buses in a city such as London, delivers a powerful network of over 8,300 mobile security cameras traversing the city's busiest regions, up to 24 hours a day, 7 days a week, significantly complementing and enhancing the city's static surveillance infrastructure.

This paper explores an alternative and more flexible approach to exploit this new and exciting infrastructure by creating the ability to have unlimited numbers of "virtual observers" placed at chosen positions to monitor the scene, spatially and over time. Surveillance along transport routes is significant because they generally link areas where people congregate, and thus provide coverage of important public areas such as central business districts. Our work describes a new approach to such wide area surveillance around transport routes using GPS and frontal camera data acquired from a transport network, and is an example of an observation system [14].

Although mobile cameras are used in specialised applications such as aerial survey (eg. UAVs), or observation of harsh or dangerous environments (eg. undersea robots) there appear to be no current attempts to use networks of mobile cameras to observe the places where people typically live and interact. The use of mobile cameras presents many opportunities, but also many challenges: (a) mobile cameras have both variable location and orientation, (b) mobile cameras sample a large spatial area at low temporal resolution, (c) the views of a particular place are distributed across many different video streams acquired by different sensors on different mobile units, and (d) the cameras are continually moving; there is no stable background image which makes it difficult to do motion-based segmentation. In addition to the challenges of interpreting the data, there is the important issue of how to design systems that deal with large numbers (eg. 1000 or 10000) of real-time streams, including both video and trajectory information. In the data-management field new systems are being developed to deal with the demands of continuous stream-based processing, but these do not generally deal with complex data such as images.

To exploit surveillance across this wide area network, we develop the VIRTOBS system, which allows an operator to see a view of a particular place over time. We use a network of buses from which GPS data and the front-camera video can be extracted. A *virtual observer* represents an imaginary camera that can be placed anywhere in the space covered by the sensor network. Each virtual observer is associated with a position, radius, orientation, and field of view. The system then constructs the view for a virtual observer by indexing, organising, and transforming images collected from the mobile camera network. Where possible, the system may build composite images by combining observations taken at different times. Queries supported by the system include:

- Construction of the view from a virtual observer. Given a source position, radius, orientation and field of view, determine what is visible for that observer.

- Synthesis of panoramic images. Where the desired field of view is wider than the camera view, we combine multiple images taken at different times to give a wide-angle perspective.

- Synthesis of "time-lapse" video, showing how a view of a place changes over time. In a static-camera application this is a trivial problem, but in a mobile environment this requires indexing, retrieval and registration of images from multiple streams.

- Retrieval of views of a particular object or landmark. Given a destination position and range of view angles, retrieve matching images based on simple visibility constraints.

- Selection of images based on multiple spatial, temporal and geometric constraints. For example, images may be selected by choosing a position from a map, or by a temporal constraint based on absolute time, or time-of-day.

- Selection of video sequences based on spatial location, and assembly of these sequences into a time-ordered composite sequence.

To build such a system that answers these types of queries, we focus on the following design issues:

- *Data Access.* Most queries are based on constraints, so it is important that data be organised for both temporal and spatial access.

- *View Synthesis.* To generate wide-angle (panorama) views, and time-lapse views we need to make images or video sequences that combine images taken from different cameras at different times.

- *Visual Query Design.* Queries are based on a multi-dimensional region of interest, which includes spatial, temporal, and orientation constraints. These must be specified in a natural way.

We provide effective *data access* by making video frames accessible via spatial, temporal, or joint spatio-temporal constraints. Each frame of video is associated with a time, a spatial position, and an orientation. This is done by interpolating a position and heading from the GPS track based on the frame sampling time. This enables us to determine what a mobile camera sees at any given time, by incorporating its orientation relative to the vehicle, and the trajectory of the vehicle over time.

Our approach to data management depends partly on features of the existing bus infrastructure. Each bus has 7 cameras which at typical sampling rates generate approximately 225Kb of data per second. A bus has enough storage for 8 to 9 days data. Wireless networks are used to retrieve data when buses return to their depot, but there is only sufficient bandwidth to retrieve about 5 percent of the collected data. Therefore it is important that the system collect data according to demand. Virtual observers can act as standing queries that regulate the collection of data from the network.

View synthesis involves the *retrieval* and *fusion* of images for a given query. Many query operations need to determine views with respect to a particular place. This poses several challenges in the context of mobile surveillance. Data is collected in an ad-hoc way, so there is high variability between the images that are available for a particular place and time. The scene is sampled infrequently compared with static-camera surveillance. Along a bus route a place is only imaged when a bus is in the vicinity, so sampling times depend on the frequency of buses on that route, resulting in a *sparse sampling* of the environment. Images of a place are taken by cameras mounted on different vehicles. There may be significant differences due to sensor response, lighting, and perspective. For simple image retrieval tasks, differences between images may not be a problem. However, for panorama generation we need to be able to *select* a sequence of relevant images, and then *register* images with respect to a common reference frame. Orientation derived from GPS data is not precise enough for image registration.

To address these issues we propose the following approach: For image selection, we use constraints on position, heading, and rate-of-change of heading to identify candidate image sequences. For image registration and blending, we use the techniques of Brown and Lowe [5]. The Scale Invariant Feature Transform (SIFT) [17] identifies feature points in images that can be used to compute the relative registration of a sequence of images ("bundle adjustment"). Blending is done at multiple spatial scales to reduce visual artifacts at the joins between images. An advantage of using SIFT features in this application is that they are invariant to many of the differences that arise due to the mobile cameras (ie. changes in perspective and lighting).

Lastly, seeking to provide natural metaphors that the user can use to specify constraints, VIRTOBS includes a visual query system where a user can place virtual observers on a map. Observation parameters can be controlled by manipulating visual markers, and the resulting observer view is automatically updated from the available observations in the database. This allows the user to deal with the complexity of the underlying data in which views of interest may be distributed across time, space and orientation.

We demonstrate the efficacy of the systems using two data sets, one acquired from a car, and the other using data extracted from a real bus network. We show the results of querying and observing the wide-areas using virtual observers. Our work demonstrates the power of the paradigm, whilst identifying open problems for the community using this new infrastructure.

The novelty of our system lies in being the first work to address the issue of wide area surveillance using a transport network and the multi-modal data streams acquired from them. The underlying design allows effective retrieval of frames in a spatio-temporal context from arbitrary perspectives, and to synthesise views of the environment to overcome the issues of sparse sampling. We also introduce novel query and presentation metaphors to make this complex data useful and usable.

The significance of the approach lies in this untapped application: wide area surveillance along transport routes. This is useful since more than 80 percent of crime is committed withing 5 km of a transport route. Given that most transport fleets have frontal cameras and GPS, these streams can be used effectively for law enforcement. The design and solutions presented in this paper forms the foundation of this open and challenging area for multimedia.

The structure of this paper is as follows. Section 2 describes related work. Section 3 describes the data model, and some query operators related to visibility. Section 4 describes the implementation of VIRTOBS, including details of the bus network. Section 5 describes our experiments with two data sets: one collected from a car, the other from a bus network. This includes sample output for queries. Section 6 discusses some open problems identified from our work.

2. RELATED WORK

The problem of mobile surveillance is related to several areas of active research, which are described in this section. Video surveillance systems are increasing in their ability to automate tasks like object tracking and event detection (see 2.1). Many of these problems are the same in mobile surveillance, but camera motion imposes an extra level of difficulty. The design of observation systems includes issues of real-time, continuous queries over stream data. This issue is also examined in many of the new stream-processing data models, which support continuous queries over unbounded data streams (see 2.2). In the spatial-database area, there is considerable interest in data models and efficient queries for moving objects (see 2.3). Some queries in VIRTOBS depend on efficient, automatic alignment and stitching of images. This has become possible using modern feature-based image matching techniques based on SIFT (see 2.4).

2.1 Observation Systems

Video surveillance is an area of active research. While many of today's video surveillance systems simply act as large-scale video recorders, the next generation of systems is being developed to automate many of the tasks that currently require the attention of a human operator. The main areas of research are video-based detection and tracking, video-based person identification, and the design of large-scale systems. Key components of these new systems are algorithms for object detection, two- and three-dimensional object tracking, object classification, object structure analysis and movement pattern analysis [12]. A key challenge in this area is to observe the scene at the correct scales. For situation awareness systems must have a wide area of observation, but for identifying an tracking people we must observe fine details like faces. Active vision techniques allow systems to focus attention on significant objects using pan-tilt-zoom (PTZ) cameras. Most wide-area surveillance systems use static fixed cameras or PTZ cameras. Indeed, many of the algorithms on which these rely (eg. background subtraction for object segmentation) only work when there is no camera motion. There have been no attempts to do wide-area surveillance with mobile camera networks.

An emerging paradigm in this area is that of *observation systems* [14]. Observation systems observe people or objects in the environment, collect data from multiple sensors, analyse and correlate data from multiple sources to derive a record of meaningful events, and provide tools to query and present information about activities in the environment. Observation systems exist in many application areas (eg. surveillance, situation awareness, traffic monitoring, population research, marketing) but share common functionality and design principles.

2.2 Stream Query Systems

In the field of data management, stream processing has become an important issue [10, 4]. Observation systems differ from traditional data-base applications in several ways: (a) data comes from external sources (eg. sensors) and the data-base must actively detect events rather than simply respond to queries, (b) data management exists over a history of events, not just its current state, (c) applications are event-oriented, requiring trigger processing beyond the capability of traditional DBMS systems, and (d) stream data may be lost, stale, or intentionally omitted so queries may have only approximate answers, and (e) real-time response may be important [3].

These requirements have led to the development of generalised stream-management systems such as Aurora [3], Borealis [2], and TelegraphCQ [7]. These systems are designed to handle a continuous inflow of data, and include the ability to handle *continuous queries* which continuously provide new results as they become available. The emphasis of these systems tends to be on real-time scalar data rather than media data such as video. Models have recently been developed for stream queries over media data [16, 11].

2.3 Moving Object Databases

Mobile surveillance systems need to be able to deal with large numbers of moving objects. *Moving Object data-bases* are an emerging area of interest in spatial data-base systems. The aim is to develop data models and query languages that allow the modelling of objects with time-dependent position [15]. Important abstractions are the *moving point* (eg. vehicles, people, or animals), and the *moving region* (eg. hurricanes, forest fires, oil spills at sea).

An important operation when dealing with moving objects is to efficiently index object trajectories. For unconstrained motion in two dimensions (eg. for ships moving at sea), various approaches exist based on grid decomposition, hashing, or hierarchical spatial decomposition (eg. R-Trees) [9, 19]. Where vehicles move on a road network, specialised data structures can be used to reduce the dimensionality of the search problem (eg. the FNR-Tree, and MON-Tree) [8].

2.4 Image Stitching

Image alignment and stitching algorithms have long been used to create high-resolution images out of mosaics of smaller images. The earliest applications include the production of maps from aerial photographs and satellite images. Recently, these algorithms have been used in hand-held imaging devices such as camcorders and digital cameras. Image stitching requires several steps [21]. Firstly, a motion model must be determined, which relates pixel coordinates between images. Alignment of pairs of images is computed, using *direct* pixel to pixel comparison, or using *feature-based* techniques. Next, a globally consistent alignment (or "bundle adjustment") is computed for the overlapping images. Next, a compositing surface is chosen onto which each of the images is mapped according to its computed alignment. The mapped images are then blended to produce the final image. The blending algorithm needs to minimise visual artifacts at the joins between images, and needs to care for difference in exposure between the source images.

Image stitching applications vary in the way they handle motion, image alignment, and blending. Direct alignment methods rely on cross-correlation of images, and tend not to work well in the presence of rotation or foreshortening. Modern feature detectors can be quite robust in the presence of certain amounts of affine transformation. Of particular note is David Lowe's SIFT (Scale-Invariant Feature Transform) [17]. In a recent survey of a number of feature descriptors [18], SIFT was found to be the most robust under image rotations, scale-changes, affine transformation, and illumination changes. Brown and Lowe [5] describe an automatic panorama stitcher based on SIFT feature matching. This is one of the first implementations that can *automatically recognise multiple panoramas* from an input set of images. A commercial version of this algorithm, Autostitch [6], is used under license in several photographic applications.

In the context of wide-area surveillance, image stitching (or "mosaicing") is important because it can be used to improve the effective resolution of a camera. Pan-tilt-zoom cameras can be used to scan a scene at different scale factors. By stitching many images collected at a high "zoom" factor, a high-resolution virtual field of view can be created. Heikkila and Pietikainen [13] describe a system that builds image mosaics from sequences of video taken by a camera that scans a scene. The implementation is similar to [5], but with a few modifications to deal with large numbers of images. SIFT features are used in image alignment. Gaussian blending is used for compositing images, but also to identify small problems with image registration.

Panoramic stitching is typically applied to static scenes. With a dynamic scene, different parts of the scene are seen at different times so it is not possible to reconstruct a true panoramic video by scanning the scene in this way. However, by stacking the image frames in a three-dimensional space-time volume, it is possible to generate movies by sweeping this volume with a *time-front* surface. This process is described by [20] as a "dynamic mosaic" or "Dynamosaic". Different interpretations of the original scene can be derived by manipulating the time-front surface.

3. DATA AND QUERY MODEL

VIRTOBS manages data collected from mobile cameras. As vehicles move around the environment their trajectory is recorded via GPS. Each camera attached to a vehicle has a known orientation relative to that vehicle; there may be more than one external camera per vehicle. This allows the system to determine a *position* and *heading* for each image in the video stream.

At the base level the system records raw GPS streams and video streams. Within each stream samples are ordered by time, although the time-bases may be different. Video data is stored as JPEG or MJPEG (motion JPEG) files. In this application it is important not to use motion-based coding. Motion coding tends to reduce the spatial resolution but more importantly, interpolated video is inadmissible as evidence in many legal jurisdictions.

A *track* is an association between a video stream and a GPS trajectory. GPS positions for vehicles are recorded every second. Video normally is sampled at a higher frame rate (eg. 5 frames per second). Therefore, it is necessary to interpolate between GPS position fixes in order to obtain accurate image positions. Currently, linear interpolation is used. Within a track, data is indexed by time; the track association includes calibration between video and GPS time-bases.

3.1 Geometric Queries

At the lowest level, spatial queries are implemented using geometric operators on tracks. Users define points or regions of interest through the visualisation system; the system interactively produces views of those regions based on these queries. Results from these queries are returned as times, or intervals of times. Given a track and a time, we can easily determine the associated spatial location, and the associated frame of video. This section describes three query operators implemented by VIRTOBS: $proxobs$, $viewIn$, and $viewOut$.

Formally, let V be a track. We define $trajectory(V)$ to be the trajectory associated with track V. The trajectory is a function that maps any point in time to a point in space using using linear interpolation on the recorded GPS track. The location of the vehicle at any point t in time is therefore $trajectory(V)(t)$, or simply $trajectory(V, t)$ for short.

Also associated with V is a video sequence. We define an *observation* to be a tuple $\langle I, t\rangle$, where I is an observed image, and t is the time at which the observation occurred. The video sequence $vid(V)$ is a sequence of N images I_i taken at discrete times t_i, $[\langle I_0, t_0\rangle, ..., \langle I_{N-1}, t_{N-1}\rangle]$. In many cases, $vid(V)$ will be sampled periodically, but we do not require it to be so. We can treat $vid(V)$ as a function that maps a time to an observation. Define $vid(V, t)$ to return the observation $\langle I, t'\rangle$ such that t' is closest to t over all observations.

A *track observation* is a tuple $\langle V, t\rangle$, where V is a *track*, and t is a time. A *track segment* is a tuple $\langle V, t_1, t_2\rangle$ where t_1 and t_2 are times, and $t_1 \leq t_2$. Track observations and track segments are returned by geometric queries. Associated with each track observation is a unique observation (a time-stamped image) $vid(V, t)$. Associated with each track segment is an observation sequence (a time-stamped video segment) $[vid(V, t_1), ..., vid(V, t_2)]$.

The simplest queries map a point in space to a track observation using recorded vehicle trajectories:

DEFINITION 1 ($proxobs$: CLOSEST OBSERVATION). *Let P be a point in space. Let T be a set of tracks. We define the function $proxobs(P, T)$ to return the track observation $\langle V, t\rangle, V \in T$ such that the distance from $trajectory(V, t)$ to P is minimised over all times and tracks.*

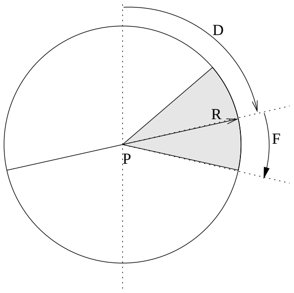


Figure 1: A simple visibility constraint defines a circular region of space with radius R centered at P, and a range of view directions $D - F$ to $D + F$.

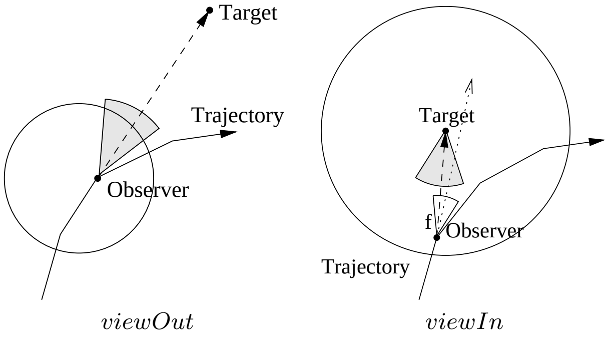


Figure 2: Interpretation of observer-target constraints for view operators.

The *proxobs* operator is computed by finding closest point on each trajectory to P, and choosing the trajectory that minimises this distance.

Visibility queries are more complex, being based on a set of spatial constraints. A *simple visibility constraint* is a tuple $\langle P, R, D, F \rangle$, where P is a point in space, R is a visibility radius, D is a view direction, and F is a field of view. This is depicted in Figure 1. A simple visibility constraint defines an acceptance area and view range. The area is a circle of radius R centered at P. The view range is the range of directions between $D - F$ and $D + F$. Visibility constraints are used by *view operators* to select observations based on visibility.

We use visibility constraints to reconstruct the view at a particular point in space. The two fundamental visibility operators are $viewOut$ and $viewIn$. Both operators use simple visibility constraints, but interpret the constraints differently as shown in Figure 2. In both cases, the observer is located inside the view area. For the $viewOut$ operator, the view target is generally outside the defined area, although its location is unknown to the system. The angular constraint is on the direction from the observer toward the target. For the $viewIn$ operator, the view target is the center of the defined area, and the constraint is on the direction from the target to the observer. The additional parameter f is related to the camera field of view and constrains the angle between the target and the trajectory heading, defining how central the target must be in the observed image.

Formally, let $C = \langle P, R, D, F \rangle$ be a visibility constraint, and T be a set of tracks.

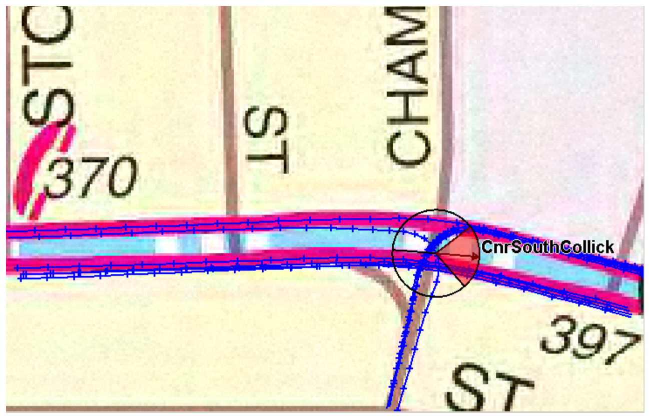


Figure 3: Schematic view of virtual observer placed on a map. The trajectories on the map indicate the paths of vehicles. The background is an ECW image of the street directory.

Definition 2 ($viewOut$: View from a place). *We define the function $viewOut(T, C)$ to be the set of track segments $\langle V, t_1, t_2 \rangle$ where $trajectory(V, t)$ is entirely contained within the circle of radius R centered at P, and the heading at $trajectory(V, t)$ is between $D - F$ and $D + F$ for $t_1 \leq t \leq t_2$.*

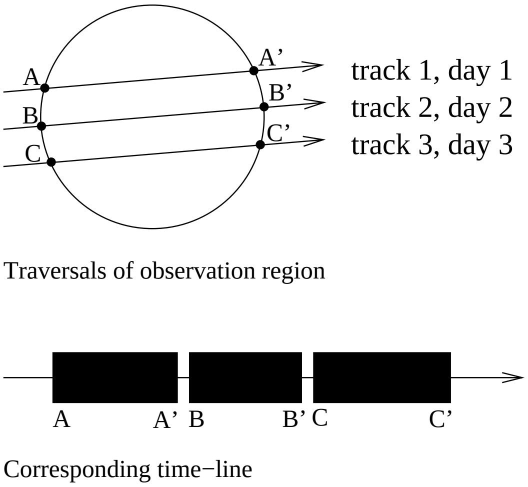
Definition 3 ($viewIn$: View toward a place). *We define the function $viewIn(T, C, f)$ to be the set of track segments $\langle V, t_1, t_2 \rangle$ where $trajectory(V, t)$ is entirely contained within the circle of radius R centered at P, and the heading of the line between P and $trajectory(V, t)$ is between $D - F$ and $D + F$ and is within the camera field of view f of the trajectory heading at t, for $t_1 \leq t \leq t_2$.*

These view operators can be rapidly computed from available trajectory information without reference to the associated video data. The operators produce a set of track segments that can be used in various ways by the system as described in the following sections. Virtual observers use view operators to create views of places; these can be defined interactively through the visualisation system (see 3.2). Sets of track segments can be used to construct “pseudo timelines” for navigation of video data (see 3.3). Track segments can also be used as observations for panorama generation (see 3.4).

3.2 Virtual Observers

VIRTOBS includes navigation of available data based on map displays. These are layered spatial displays that show trajectories for one or more tracks, marker objects (including virtual observers) placed on the map by the operator, and geographic meta-data. Spatial meta-data can be imported from geographic information systems. The system supports the use of ECW (Enhanced Compressed Wavelet) imagery as display layers. This can be used to show street maps, or aerial images associated with a spatial region.

A virtual observer combines a view operator with a simple visibility constraint $\langle P, R, D, F \rangle$ and is created interactively by placing a marker on a map display. These markers may be named and used for navigation in the map display. Figure 3 shows the schematic representation of a virtual observer on



Traversals of observation region

Corresponding time−line

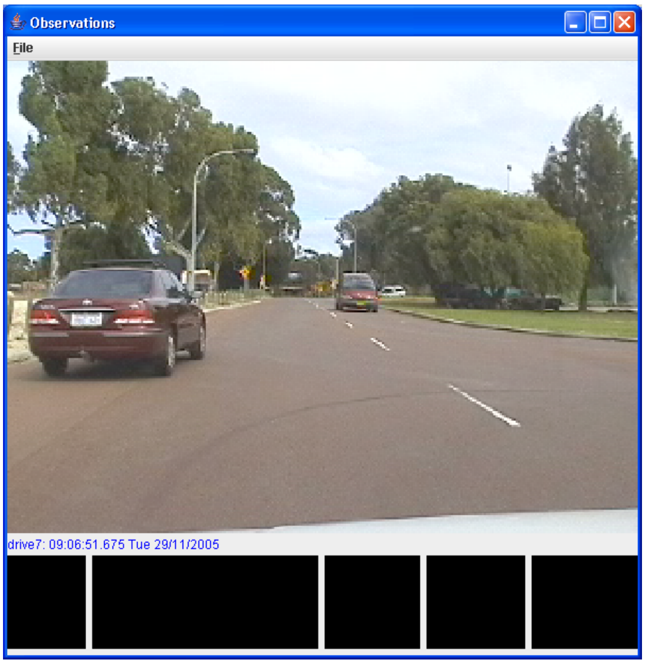
Figure 4: Pseudo time-line generated from multiple track segments.

the map display. Trajectories on the map indicate the paths of vehicles. The acceptance area is shown as a coloured circle of radius R centered at P. The view direction is shown by an arrow with direction D. The field of view is indicated using a shaded arc from heading $D - F$ to $D + F$. These parameters may be varied by dragging points in the image (eg. the circle boundary to change R, the circle interior to move the center P, the arrow-head to rotate the view direction D, and the arc boundaries to change the field of view F). A virtual observer is selected using the mouse, or by choosing its name from a list. When selected the system executes the associated view query, searching for trajectories matching the associated constraints. The result is a set of track segments where each image has a position within the defined area and a heading within the constraint of the query operator.

In addition to virtual observers, map displays implement a "tracking" mode in which the user can move a cursor in the display to select the closest matching observation. Given a point P, the system computes $\langle V, t\rangle = proxobs(T, P)$ and displays the associated image $vid(V, t)$. Depending on cursor modifiers, T is either the set of all tracks, or a particular selected track. Tracking can be used to generate a kind of "virtual drive" effect, where a video sequence can be generated for an arbitrary trajectory through a map.

3.3 Pseudo time-lines

Virtual observers act as filters that select sets of track segments. It is important to be able to display and navigate the associated video data. Showing the segment times on a linear time-line would not be very useful, since the durations of the track segments are short compared to the times between the segments. Instead, the system displays a *pseudo-time-line* with just the duration of the segments, ordered according to their start time. *This clearly shows that the segments of video are discontinuous, but allows them to be navigated as a continuous sequence.* Figure 4 shows a scenario where three tracks traverse an observation area. The associated track segments are bounded by the times that the camera is resident within the area. The corresponding time line shown below orders the segments as AA',BB',CC'. This resulting time-line can be used to continuously navigate the set of non-continuous video segments by moving the cursor across the pseudo-time line.



drive7: 09:06:51.675 Tue 29/11/2005

Figure 5: Observation view showing pseudo-time-line and associated image. Blocks in the time-line indicate the order and relative duration of the track segments, but not their absolute time.

Figure 5 shows example output from the system. In this instance, there are five video segments in the data-base that match the view constraints. The relative durations are indicated by the different segment lengths. In the longer segment (for which the image is shown) the vehicle had to stop to wait for an oncoming car.

A unique point in space $trajectory(V, t)$ is associated with any time $t_1 \leq t \leq t_2$ selected from a track segment $\langle V, t_1, t_2\rangle$. The system implements a *space-time cursor* which allows the user to see correspondence between points in the spatial map display and the time-line display. When selecting points in the time-line, the system highlights the corresponding location in the map. Additionally, the user can select points on tracks in the spatial display and see the corresponding images.

3.4 Panorama Generation

When a vehicle turns, the forward-facing camera pans across the scene producing a sequence of images which can be combined to form a composite, wide-angle image. When a virtual observer is placed at an intersection or turning in the road, the matching track segments define a sequence of images suitable for stitching. Alternatively, the system can identify candidate track segments by looking for regions where the rate-of-change of heading is high (10 degrees per second seems to give good results). Example panoramas are shown in Figures 7, 8 and 10 in section 5.

VIRTOBS uses the method of Brown and Lowe [5] to build panoramas from a set of images. This involves several steps. Feature points are identified using the SIFT [17] key-point detector. Each key-point is associated with a position, a scale, and orientation. SIFT features are robust to small amounts of affine transformation. SIFT features are calculated for each input images. The k nearest-neighbours are found for each feature. For each image, the algorithm considers m images that have the greatest number of feature matches to the current image. RANSAC is used to select a set of inliers that are compatible with a homography between the images. A probabilistic model is then used to verify image matches. Bundle adjustment is then used to solve for all of the camera parameters jointly. Once the camera parameters have been estimated for each image, the images can be rendered into a common reference frame. Multi-band blending is then used to combine images.

VIRTOBS uses Autostitch [6] to implement panorama construction. Although designed for photographic work, it also works well for images taken from mobile video cameras.

4. IMPLEMENTATION

VIRTOBS is implemented in Java. We use Java Swing components for user interface and visualisation. Media I/O is done using either core Java classes, or QuickTime APIs. Third-party components are used to render ECW images. Autostitch[1] is used for panorama stitching. There are several main parts to the implementation. A low-level stream-based storage management system handles video and GPS data, which are stored on disk and indexed by time. At a higher level a track management system relates video streams, camera parameters and trajectories. This permits retrieval based on spatial constraints such as proximity and visibility.

4.1 Data Sets

We used two data sets to evaluate the system. The first data set ("car") was collected in a regular passenger vehicle. A DV camera was mounted on a tripod in the passenger seat. Video was recorded in colour at 5 frames per second, 320x200 resolution to motion JPEG video files. Raw NMEA GPS log files were recorded for the same time period, and an offset (video time relative to GPS time) was determined by interactively aligning the video and trajectory in the application. The second data set ("bus") was collected from a commercial MDR (Mobile Digital Recorder) bus surveillance system developed by DTI [1]. This data was recorded at 1 frame per second, 384x288 resolution as JPEG images. The data is stored in a proprietary format which includes time-stamped images for multiple camera channels as well as GPS positions; this was then extracted to standard formats which are used by our system. Our trials used data collected from the existing bus network. Some features of this network are outlined below.

4.2 Bus Network

Each bus has 7 cameras that record 24-bit colour images at 384x288 resolution. The global sampling rate is around 15 frames per second; this is shared between cameras as required, giving around 2 images per second for each camera. The sampling rate can be increased for particular cameras by reducing the rate for others. Using JPEG compression, a typical images is around 15Kb, giving an overall data rate of approximately 225Kb per second. Typically, a bus operates around 85 hours per week, resulting in about 67Gb of data per week. Each bus is fitted with 80Gb of storage, so images can be retained for 8 to 9 days.

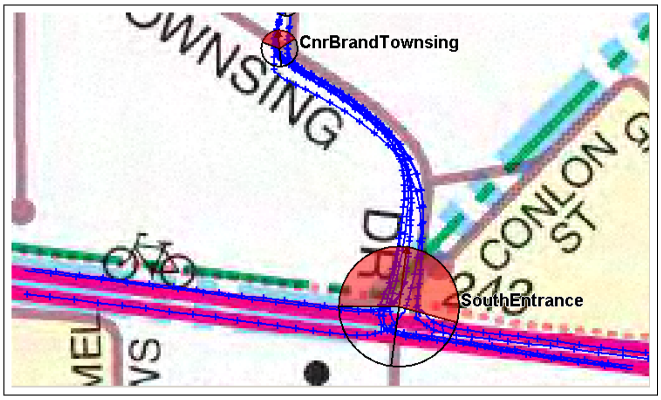


Figure 6: Placement of observers for car panorama experiments.

When buses return to their depot, data is downloaded via wireless LAN. The average operational time is 12 to 15 hours per day, which leaves about 8 to 10 hours per day for downloads. Each depot has about 100 buses, but they all converge around the same time, outside of "rush hours". The wireless link is 802.11g but despite the 54Mbps bandwidth, the effective throughput is about 15–20Mbps. This leaves in the worst case around 540Mb of data per bus per day. This is sufficient to retrieve about 5 percent of the video data. Thus, it is critical that the system is selective about what data is retrieved and what data is discarded.

Given the constraints of the sensor network, it is important that the system collect data based on demand. Rules must be used to determine what data needs to be systematically recorded. For external cameras, these constraints could be based on desired spatio-temporal resolution at different places and times. Virtual observers provide another mechanism for regulating data collection. Each observer indicates an area of interest that may be stable over long periods of time. Data around these points should always be collected at high resolution in time and space.

5. EXPERIMENTS AND DISCUSSION

We evaluated VIRTOBS on the above two data sets to determine how well the view operators work, and under what circumstances panoramas can successfully be generated. There are some significant differences between the two sets. The car data-set has a camera with high quality optics, a high sampling rate, and highly linear images (ie. no distortion). The bus data set has a lower sampling rate, and significant spherical distortion due to a wide-angle lens. In addition, the bus camera is mounted near the roof of the bus, and its view out the windscreen is partially obstructed by a strip of tinting that produces colour distortion.

Initial tests show results for the car images. Figure 6 shows the placement of two observers on a map. The top observer (labelled "CnrBrandTownsing") is placed at a corner close to a road-works site. The bottom observer (labelled "SouthEntrance") covers an intersection with views in many directions. We explored the generation of panoramas at these sites.

[1] The demonstration version of Autostitch [6] is called automatically as part of our visualisation process.

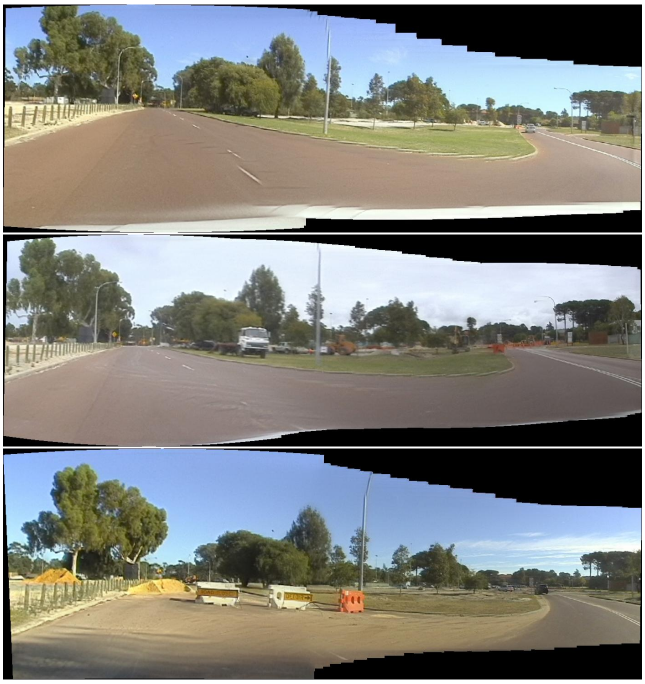


Figure 7: Three panoramas generated for virtual observer "CnrBrandTownsing" shown in Figure 6.

5.1 Time-Lapse Panorama

Figure 7 shows several panoramic views generated automatically from the "SouthEntrance" virtual observer using the *viewOut* operator. Each panorama corresponds to a separate traversal of an intersection, and is based on 20 to 30 frames taken over roughly 5 seconds. The panoramas are not completely linear in size since the turn involves some forward motion as well as a rotation. This means that the later images are enlarged (ie. "zoomed") relative to the earlier images. During bundle-adjustment these images are scaled down to fit a consistent reference frame. There are also small variations in the shape of the resulting image, due to differences in the original trajectories.

The virtual observer is located close to the site of road works. The first image shows the the scene before the road work starts. Subsequent images show the changes as works progress: arrival of trucks in the second, and the erection of portable barriers in the next.

5.2 Temporally Non-Continuous Panorama

An important feature of the panoramic stitching process is that it simply relies on common features to compute image registration. The previous panoramas are generated from temporally contiguous samples, but this is not necessary for the stitching to work. Providing there is sufficient overlap sub-sequences can be taken at different times.

Figure 8 shows an example of the kind of scene that can be generated by stitching together images taken at different times. The output is from the "SouthEntrance" observer. As a vehicle turns at an intersection, the forward facing camera pans across part of the scene. The left-hand portion of this image is derived from a right-turn from the west-bound lane. The right-hand portion is derived from a left-turn from the east-bound lane. When interpreting such an image, it is important to recognise that the image is a composite constructed from observations at different times. While the large-scale structure will probably be correct, it may be misleading to make assumptions about objects moving in the scene.

5.3 Simple time-lapse

A virtual observer can be used to select a sequence of images at a particular place. When viewed in sequence, this give a time-lapse picture, showing how the scene changes over time. For a fixed camera, this is a trivial operation. For mobile cameras the system must retrieve and sequence images from difference video streams based on spatial location and orientation. An advantage of simple time-lapse over panoramic time-lapse is that it does not require any special camera motion. A disadvantage is that a relatively narrow view of the scene is obtained.

Figure 9 shows a short time-lapse sequence captured from the bus data set. It consists of three sequences retrieved at 08:58:39, 16:01:28, and 16:29:15 from the bus network footage. The first frame shows the scene in the morning at 08:58:39. The second frame at 16:01:28 shows a crowd of people in front of a building; the next frame at 16:29:15 shows that the crowd has dispersed, although a parked vehicle can be seen in the same place.

Note that this figure demonstrates some problems with accuracy in the GPS measurements that can occur when the signal paths are obscured by buildings. This suggests that better approaches to positioning (eg. involving intertial navigation) may be required in some environments.

5.4 Low sampling-rate Panorama

The bus data set presented a few difficulties in the generation of panoramas. Firstly, the sampling rate (1 frame per second) is low, so there is significant difference between the frames that make up a track segment. Higher sampling rates produce more overlap between frames which gives more feature-point matches and better registration. Secondly, spherical distortion affects both feature matching and registration of images (see Figure 9 for sample images). SIFT features have an orientation which is used for feature matching. The distortion leads to a relative scaling and rotation of features between frames, which the SIFT is designed to handle, but which may reduce the number of matches. More significant is the effect on bundle adjustment, since the spherical view produces point correspondences that do not fit the camera model. Despite these problems, some panoramas could be generated. Figure 10 shows an example.

6. DISCUSSION AND OPEN PROBLEMS

While this paper lays the foundation for exploiting this new and exciting infrastructure, we outline several open challenges from our study.

Algorithms to deal with distortion and low sampling rates: To overcome problems of lens distortion on buses we could use a linear (non-distorting) lens, de-warp the images, or change the camera model. Future trials will use a linear lens positioned to avoid the window tinting. Combined with a higher sampling rate, this is expected to significantly improve the panorama quality. In the car data set, it was found that taking every second frame did not significantly reduce the ability to match frames, or the quality of the panorama. However, taking every third frame did reduce

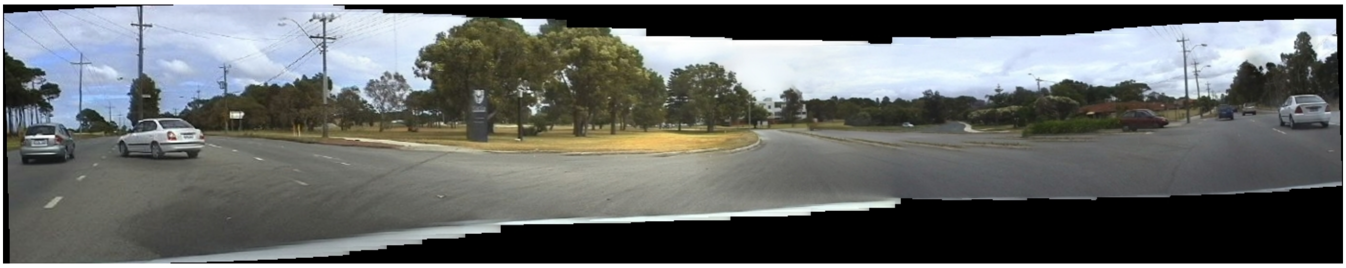
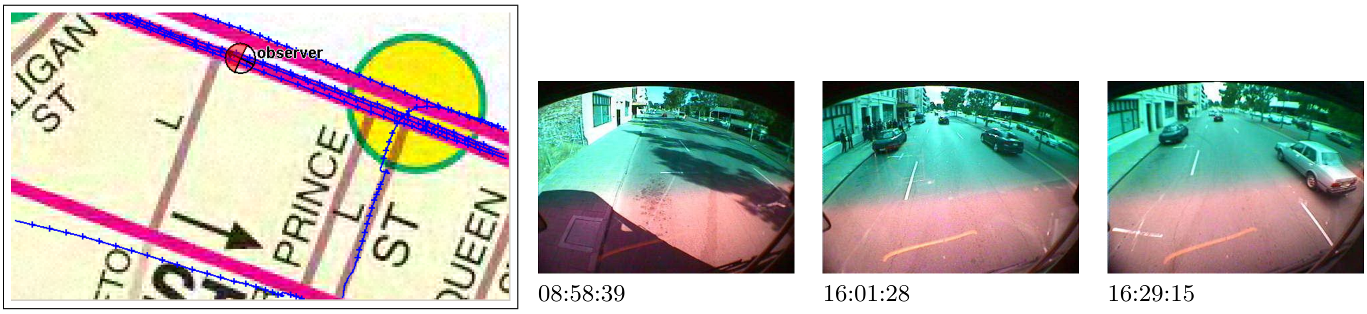


Figure 8: 180 degree synthetic panorama generated by fusing two sequences of video observations taken at different times. The left-hand portion of this image is derived from a right-turn from the west-bound lane. The right-hand portion is derived from a left-turn from the east-bound lane. This scene is generated by the "SouthEntrance" virtual observer shown in Figure 6.



08:58:39

16:01:28

16:29:15

Figure 9: Time-lapse view of a place generated by the virtual observer shown in the inset.

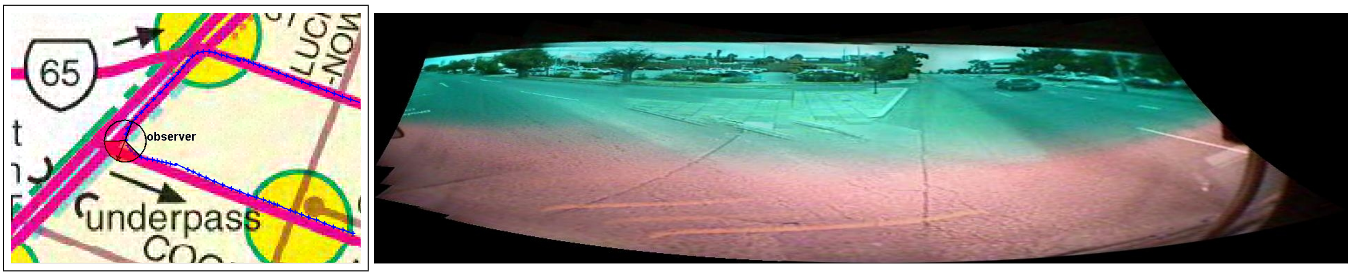


Figure 10: Panorama of bus images generated by the virtual observer shown in inset.

the effectiveness of image matching. Therefore, roughly 2 frames per second is probably a minimum sampling rate for this application.

Improved Blending Algorithms: In our experiments, most of the processing time is required during the blending phase of the algorithm. Using a simpler blending algorithm (eg. linear blending instead of multi-band blending) improves processing time dramatically. In an interactive setting where response time is significant, it may make sense to progressively improve the blending quality as images are viewed for longer periods. For example, the initial image may be presented using linear blending, while a multi-band blend is started as a background process, taking maybe 20 or 30 seconds to complete with high quality settings.

Alternative configurations for data acquisition: The current implementation assumes that the camera sweeps across the scene by rotating around a common optical center. It also works well where some forward motion occurs during the rotation (ie. a forward-facing view from a turning vehicle). Another model for sweeping a scene would be to have a camera facing perpendicular to the direction of motion (ie. a side-facing view from a vehicle). This latter model has the advantage that almost any motion of the vehicle would scan the scene, whereas the former model requires a turning motion. It is expected that the approach of Brown and Lowe would also work for side-facing cameras, although some variation to the formulation of the camera homographies would improve the camera modelling. Indeed, this approach (moving perpendicular to the view axis) is used in most aerial photography applications. A related technique is to use a scanning vertical "slit" to build "route panoramas" [22]. These give a strip-like panoramic representation of the view to the side of a road.

7. CONCLUSION AND FUTURE WORK

The development of VIRTOBS is an attempt to understand how we can deal with the complex data that arises from mobile cameras. We have designed and implemented a system that allows flexible querying of multiple observation streams. We present the design of the system, the query operators, and visualisation environment and demonstrate its efficacy using two data-sets. We have received positive interest from both public transport and law enforcement authorities, which suggests that it is worth extending this prototype to a fully-fledged system.

8. ACKNOWLEDGEMENTS

StreetSmart Street Directory images were provided courtesy of the Department of Land Information, Western Australia, 2006. Bus data was provided by Digital Technology International [1].

9. REFERENCES

[1] Digital Technology International. Web site visited April 2006. http://www.dti.com.au/.

[2] D J Abadi, Y Ahmad, M Balazinska, U Çetintemel, M Cherniack, J-H Hwang, W Lindner, A S Maskey, A Rasin, E Ryvkina, N Tatbul, Y Xing, and S Zdonik. The Design of the Borealis Stream Processing Engine. In *Second Biennial Conference on Innovative Data Systems Research (CIDR 2005)*, Asilomar, CA, January 2005.

[3] D J Abadi, D Carney, U Çetintemel, M Cherniack, C Convey, S Lee, M Stonebraker, N Tatbul, and S Zdonik. Aurora: a new model and architecture for data stream management. *The VLDB Journal*, 12(2):120–139, 2003.

[4] B Babcock, S Babu, M Datar, R Motwani, and J Widom. Models and issues in data stream systems. In *PODS '02: Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–16, New York, NY, USA, 2002. ACM Press.

[5] M Brown and D G Lowe. Recognising panoramas. In *9th IEEE International Conference on Computer Vision (ICCV)*, pages 1218–1227, 2003.

[6] Matthew Brown. Autostitch. Web site visited April 2006. http://www.autostitch.net.

[7] S Chandrasekaran, O Cooper, A Deshpande, M J Franklin, M Hellerstein, W Hong, S Krishnamurthy, S R Madden, F Reiss, and M A Shah. TelegraphCQ: Continuous dataflow processing for an uncertain world. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 668–668, New York, NY, USA, 2003. ACM Press.

[8] V T de Almeida and R Hartmut Güting. Indexing the trajectories of moving objects on networks. *Geoinformatica*, 9(1):33–60, 2005.

[9] V Gaede and O Günther. Multidimensional access methods. *ACM Comput. Surv.*, 30(2):170–231, 1998.

[10] L Golab and M T Özsu. Issues in data stream management. *SIGMOD Rec.*, 32(2):5–14, 2003.

[11] A Gupta, B Liu, P Kim, and R Jain. Using stream semantics for continuous queries in media stream processors. In *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, page 854, Washington, DC, USA, 2004. IEEE Computer Society.

[12] A Hampapur, L Brown, J Connell, A Ekin, N Haas, M Lu, H Merkl, and S Pankanti. Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking. *Signal Processing Magazine, IEEE*, 22(2):38–51, 2005.

[13] M Heikkilä and M Pietikäinen. An image mosaicing module for wide-area surveillance. In *VSSN '05: Proceedings of the third ACM international workshop on Video surveillance & sensor networks*, pages 11–18, New York, NY, USA, 2005. ACM Press.

[14] Ramesh Jain. White paper on observation systems. personal communication.

[15] J A C Lema, L Forlizzi, R H Güting, E Nardelli, and M Schneider. Algorithms for moving object databases. *The Computer Journal*, 46(6):680–712, 2003.

[16] B Liu, A Gupta, and R Jain. MedSMan : A streaming data management system over live multimedia. In *MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia*, pages 171–180, New York, NY, USA, 2005. ACM Press.

[17] D G Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[18] K Mikolajczyk and C Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(10):1615–1630, 2005.

[19] M F Mokbel, T M Ghanem, and W G Aref. Spatio-temporal access methods. *IEEE Data Eng. Bull.*, 26(2):40–49, 2003.

[20] A Rav-Acha, Y Pritch, D Lischinski, and S Peleg. Dynamosaics: Video mosaics with non-chronological time. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 58–65, Washington, DC, USA, 2005. IEEE Computer Society.

[21] R Szeliski. Image stitching and alignment. In N Paragios, editor, *Handbook of Mathematical Models in Computer Visions*, pages 273 – 292. Springer, 2005.

[22] Jiang Yu Zheng. Digital route panoramas. *IEEE MultiMedia*, 10(3):57–67, 2003.