Annotating and Navigating Tourist Videos*

Bo Zhang
Microsoft Research Asia
Beijing, China
bzhang@microsoft.com

Qinlin Li†
Sun-Yatsen University
Guangzhou, China
treeislikethis@hotmail.com

Hongyang Chao
Sun-Yatsen University
Guangzhou, China
isschhy@mail.sysu.edu.cn

Bill Chen
Microsoft Corporation
Redmond, WA 98052
bilchen@microsoft.com

Eyal Ofek
Microsoft Corporation
Redmond, WA 98052
eyalofek@microsoft.com

Ying-Qing Xu
Microsoft Research Asia
Beijing, China
yqxu@microsoft.com

ABSTRACT

Due to the rapid increase in video capture technology, more and more tourist videos are captured every day, creating a challenge for organization and association with metadata. In this paper, we present a novel system for annotating and navigating tourist videos. Placing annotations in a video is difficult because of the need to track the movement of the camera. Navigation of a regular video is also challenging due to the sequential nature of the media. To overcome these challenges, we introduce a system for registering videos to geo-referenced 3D models and analyzing the video contents. We also introduce a novel scheduling algorithm for showing annotations in video. We show results in automatically annotated videos and in a map-based application for browsing videos. Our user study indicates the system is very useful.

Categories and Subject Descriptors

H5.1 [**Information interfaces and presentation**]: Multimedia Information Systems - Video.

General Terms

Algorithms, Design, Human Factors

Keywords

Geo-tagged Contents, Video Annotation, Video Navigation

1. INTRODUCTION

Every day, tourists capture an amazing number of videos of their trips. Some tourist videos are concerned with capturing people, like the hustle and bustle of Times Square or the tourists in the Basilica di San Marco. Others capture the layout of the scene, like breath-taking panoramas atop the Seattle Space Needle or sunset in Yosemite. However, almost all tourist videos suffer from the same problems:

- After capturing the video we oftentimes want to annotate the video with text describing landmarks to help us remember the experience.
- Video navigation is difficult, because events and locations are stored sequentially. We may remember our trip by location, instead of time of occurrence.

Annotating a tourist video is important because, unlike other videos, tourist videos often capture a point of interest, such as a modern palace or a historic temple. Such annotations are useful in helping a traveler share her story with friends and family. For example, an annotated video of traveling along the canals of Venice would show the names of churches and monuments as they appear. Unfortunately, the labels are usually static and do not follow the landmarks in the video.

Video navigation is equally important, especially when location can be a valuable cue for remembering our vacation experience. Traditionally, a video is played sequentially. Unfortunately this forces the viewer to play through the entire sequence, some of which may be uninteresting. Alternatively, the viewer could fast forward, but could potentially skip over interesting events. Neither solution is satisfactory.

We present a system that automatically annotates a georegistered video and visualizes it in an interactive map application. To annotate the video, we introduce a global, dynamic programming algorithm that balances maximizing the number of annotations shown against cluttering the video with too many annotations.

After annotation, the video is embedded in an interactive map application for visualization. In this application we introduce several novel techniques for controlling the video in a non-sequential manner. These techniques enable a user to quickly jump to important times in the video, based on its content.

In order to evaluate the effectiveness of these controls, we conducted a user study. Our results indicate that users conveyed great interest in the integrated video and map appli-

*Video at http://www.youtube.com/watch?v=gnnlUrzG890

†This work was done while the author was visiting Microsoft Research Asia.

cation. We also reason about several principles for future designers who are interested in building map and video applications.

In summary, our work has three main contributions:

- a scheduling algorithm to layout annotations in the video
- integrating video and maps in an interactive application
- user interface design principles for controlling video and map components

2. RELATED WORK

Our work spans a large range of areas including annotation, image and video registration, and geographic information systems.

2.1 Annotation

While image annotation covers a large body of work, the most relevant to ours include those that annotate geo-referenced images [5, 9]. First, the images are registered to geo-referenced 3D models. Next, GIS data such as landmark names are projected onto these images using the computed calibration. In both [5] and [9], they manually register the image. Our system enables video annotation, without manually registering every frame.

One clever extension to annotating images is annotating gigapixel images [12]. In their approach, they define a perceptual distance function between individual annotations and views that guide the rendering of both audio and text annotations. This image-based work has some similarity to video annotation, as the viewing conditions (zoom, pan) changes over time according to the user interaction. However, our work must handle more complex issues, such as registering every video frame to geo-referenced 3D models.

2.2 Geographic Information Systems

Several works exploit the vast amount of available GIS data to enrich their own media in a geographical context. [15] propose an interactive system for sharing a long video tour. They provide a map-based storyboard to enable the viewer to navigate the video in a joint location-time space. However, they emphasize how to manually place video clips at reasonable positions on the map. Our work focuses on fully registering the video so that each frame is positioned correctly in 3D.

The most similar GIS work to ours is that of [1], where they describe a system for geospatial video search. They conduct a study on how to quantify, store and query in a scene of captured videos. They assume the video is captured with positioning (e.g. GPS) and orientation (e.g. compass)-capable hardware. Our work differs in that we assume that the video is uncalibrated. In other words, our system could work on tourist videos downloaded from the web, for which there is probably no position or orientation information. One of our main contributions is a technique for obtaining the video track and orientation. Once computed, our calibration can be used as input to the system of [1].

[9] enhance and dehaze photographs by exploiting per-pixel GIS data, such as depth and texture. [19] enable a user to browse geotagged photos on a map. The map also allows for spatial queries, retrieving nearby photos. Finally, it should be noted that many GIS systems are themselves publically sharing data [13, 7] and can present map data in different views (e.g. aerial views, street views, etc.). However, they do not support the means to register personal videos to their data. Our work provides such a tool for assisting a user to geo-positioned video.

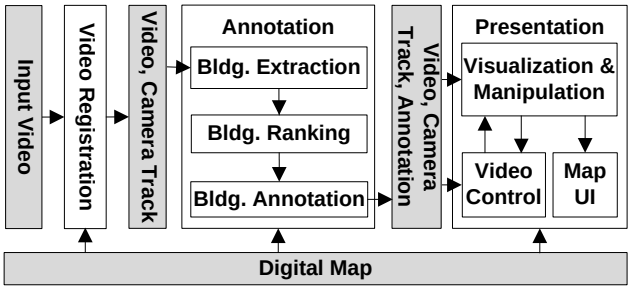


Figure 1: Architecture of the system

3. SYSTEM ARCHITECTURE

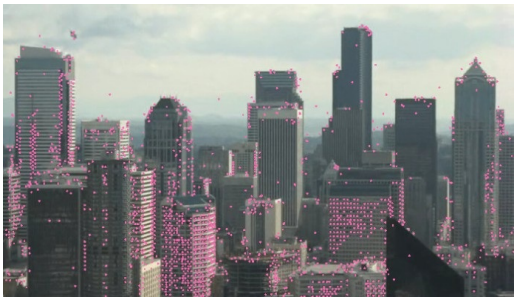
Figure 1 is an architectural diagram of our system. The system consists of three stages: video registration, annotation and presentation. In the video registration stage, a tourist video is registered to 3D models using a key frame registration tool and automatic tracking techniques. The output of this stage is a camera trajectory. This trajectory is input to the video annotation stage, which projects 3D models onto the video and analyzes the scene for annotation. In the last stage, the annotated video is presented to the user in an integrated video and map application. The application enables non-linear browsing of multiple videos and enriches the browsing experience with contextual, geographic information.

4. VIDEO REGISTRATION & TRACKING

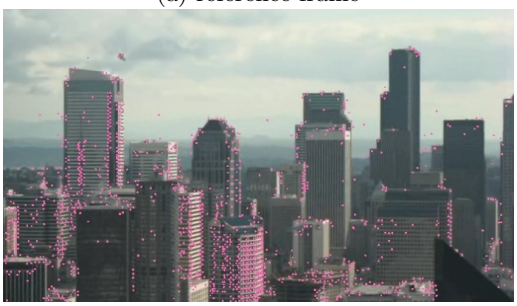
The input is a tourist video clip. We assume the video has reasonable resolution, but need not have any location information (i.e. GPS). In other words it can be captured using a consumer video camera. The key frames can be chosen manually, or semi-automatically with the help of any well-known techinique for video segmentation [14, 4, 8]. The number of key frames depends on the complexity of the shot, but has never exceeded more than a handful.

For each key frame specified, we use an interactive registration tool [3] to align the image to 3D terrain and building models [7, 13](Figure 2). Next, SIFT features [11] are extracted for all the frames (Figure 3). The system then uses structure-from-motion techniques similar to [18] to propagate the calibration parameters from each key frame to other frames in the same shot, with the help of 3D building models and terrain, which are used to filter out SIFT features near occlusion boundaries. The output is a camera trajectory where each video frame has camera calibration. In Figure 4 the camera calibration is visualized by projecting the visible building models onto an intermediate frame (e.g. not a key frame). Notice that the projected models align well to the imagery.

Figure 2: Key frame registration to a digital 3D world. The image is overlayed on top of a rendering of the 3D models.



(a) reference frame



(b) target frame

Figure 3: SIFT features in the reference (a) and target (b) frames.

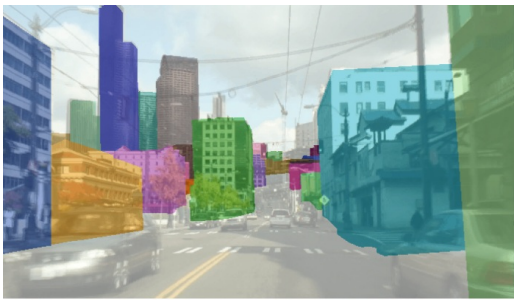


Figure 4: Overlaying 3D models in a street level video.

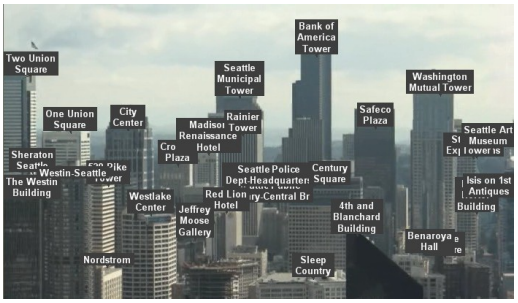


Figure 5: Projecting all labels onto a frame. The resulting image is cluttered and uninformative.

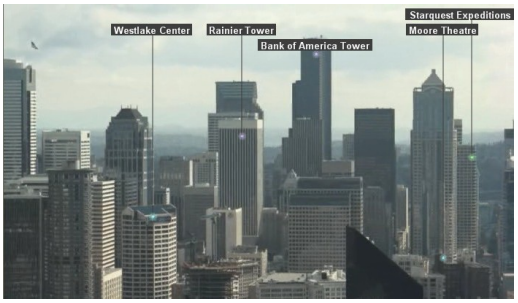


Figure 6: Scheduling annotations by their score. The annotations are less cluttered, as compared to the same scene in Figure 5.

5. VIDEO ANNOTATION

Once a video is calibrated, we annotate it with semantic content. More specifically, we assign to each pixel labels referring to semantic information like landmark information, anecdotes, etc. Annotated labels may be used to enrich tourist videos and to pick objects in the videos.

Unfortunately, simply projecting all landmark information results in a clutter of labels, as shown in Figure 5. Instead, we calculate a score for each building and we use this score to schedule its annotation in time (e.g. across frames) and space (e.g. in the frame). Intuitively, we wish to balance showing buildings with high scores with maintaining a maximum number of annotations per frame. Figure 6 is an example annotation result.

First, we describe how to score the building importance in each frame. Then we show how to use this score to schedule annotations in the video while keeping under the maximum number of annotations per frame.

5.1 Scoring Building Importance

We seek to compute an importance score for each building at every frame of the video. A low score at a particular frame means that the building is unimportant at that time. We define a scoring function $S_f(b)$ at frame f for a building b as:

$$S_f(b) = \alpha_p P(b) * \alpha_r R(b) \quad (1)$$

where P is a function describing the building's projection area within frame f. R is a function describing the building's

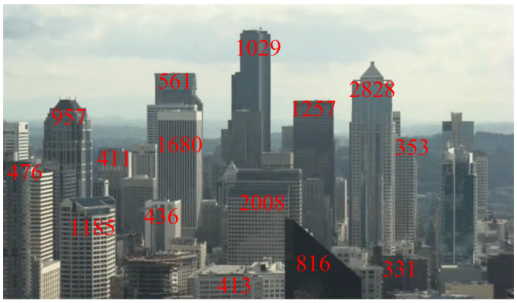


Figure 7: Scoring buildings in a frame. Buildings that have large projection area and are close to the center of the frame have high scores. Note: although a building may have a high score in one frame, this doesn't guarantee the annotation will be shown. The scheduling algorithm (described in the next section) determines the visibility of the annotation.

proximity to a region of interest in the frame. For purposes of clarity, we omit the f subscript in P and F, but both depend on the current frame f. α_p and α_r describe the relative weights between the two functions.

The first term, P, describes the building's projection area in the frame. Intuitively, a large area means the building is important. Unfortunately, this unfairly penalizes buildings that are far away. Therefore we add a bias term that will increase the building score if it is further away: $\log(d)$, where d is the depth of an anchor point on the building. The anchor point is typically the center of the building. This leads to the definition of P:

$$P(b) = \log(d)p(b) \tag{2}$$

where $p(b)$ is the building's projected area.

The second term, R measures the distance between the building's anchor point and the region of interest (ROI). The ROI is typically a window in the center of the frame. We score a building higher if its anchor point is close to the ROI. If we parameterize the ROI by a 2D center point r and define a as the 2D projection of the building anchor point, this leads to the definition of R:

$$R = d_M - \|a - r\| \tag{3}$$

where d_M is a predefined maximum distance.

Figure 7 shows the scores of buildings in one frame. We set $\alpha_p = \alpha_r = 0.5$. We have found empirically that these weights produce intuitive scores that match visual importance. However, adjusting these weights enables a user to specify whether a building's projection is more important than being centered in the frame.

Once we score each building in each frame using Equation 1, we can define the *lifetime* of a building as the set of frames in which the building has non-zero score. Figure 8a visualizes the lifetime of several buildings in a video. At this point we calculate an average score for all the frames in each second of the building lifetime and smooth the scores using a box filter; the scores may vary due to noisy calibration or visibility change across frames. We also remove any building with a lifetime shorter than a predetermined time span.

5.2 Scheduling Annotations in the Video

After scoring the buildings, we schedule when and where to display each annotation. We introduce a dynamic programming algorithm [2] to schedule when each annotation appears. Recall the goal is to show annotations of highly-scored buildings while maintaining a maximum number of annotations per frame. The user defines a maximum number of annotations per frame, m.

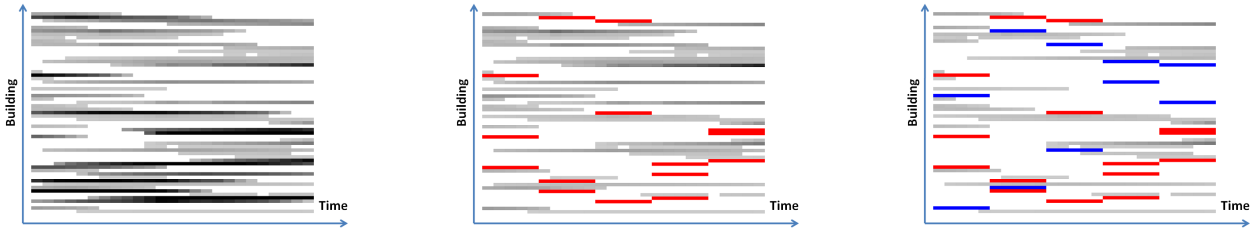
The task is to schedule n annotations in an l-second video sequence, where each building has a non-negative score for each second of its lifetime. First, we calculate how long to show an annotation in the video (e.g. its lifetime), based on its building score. We use a linear mapping from score to lifetime length; this way, the annotation of the building with the highest score remains in the video the longest. Next, annotations are scheduled for display, for some period in the video within the lifetime of the respective building. Annotations are scheduled at increments of seconds. At most m annotations are allowed in any second. The goal is to maximize the sum of the scores of all the scheduled annotations through their respective building lifetimes.

This scheduling problem can be solved using a top-down approach of dynamic programming. The state is defined to be the numbers of available annotation slots in every second. A subproblem is defined as scheduling the last k annotations at a given state. For each subproblem, its first annotation is examined and a set of possible insertion positions are determined according to the current state. A final decision is made to discard it or schedule it somewhere by maximizing the sum of the score of this annotation and the optimal score of the corresponding subproblem. If there is any unsolved subproblem, the algorithm solves the subproblem recursively.

Using this approach, the scheduling problem is formulated as the subproblem of scheduling all the n annotations given the initial state of $\{m, m, \ldots, m\}$ when all the m slots of every second are available.

The state transition function looks fairly straightforward. If an annotation is discarded, the state remains the same. Otherwise, for each second the annotation occupies, the number of available slots decrease by one. However, the state space is quite large if we just stop here. One observation is that given the minimum length of annotation lifetime d seconds, only if there are at least d continueous available slots, an annotation can be possibly scheduled. Thus, we can remove useless slots and reduce the state space dramatically. In our experiments, we consistently observe a reduction of more than 90% and the reduction percentage keeps increasing as the state space becomes bigger (95% for $l = 25, d = 5, n = 13, m = 3$).

The following pseudo-code summarizes the scheduling algorithm. For simplicity, we assume all the annotations have the same lifetime length of d seconds. $Scores[n, l - d + 1]$ is a score array pre-calculated for each annotation and each possible starting position.



(a) Lifetimes of several buildings (b) Annotations scheduled in 1st pass (c) Annotations scheduled in 2nd pass

Figure 8: Illustration of the scheduling algorithm. In (a), the horizontal axis is time. The vertical represents the buildings. A horizontal bar represents a building lifetime. Its lightness corresponds to the score: light is low, dark is high. In (b), red bars represent annotations scheduled in the first pass when m=3. In (c), blue bars represent annotations scheduled in the second pass when m=2.

```
SCHEDULER(k, state, [out]score, [out]policy)
 1  if k = 0
 2    then score ← 0
 3         policy ← EmptyList
 4         return
 5  if MEMORIZED(k, state, [out]score, [out]policy)
 6    then return
 7  SCHEDULER(k − 1, state, [out]score, [out]policy)
 8  policy.Add(−1) // annotation is discarded
 9  for i ← 0 to l − d
10  do if SLOT-AVAILABLE(i, state)
11       then state′ = STATE-TRANSITION(i, state)
12            SCHEDULER(k − 1, state′,
13                      [out]score′, [out]policy′)
14            score′ = score′ + Scores[n − k, i]
15            if score′ > score
16              then score = score′
17                   policy = policy′.Add(i)
18  MEMORIZE(k, state, score, policy)
```

Unfortunately, the problem cannot be solved in polynomial time. One solution is to divide the problem into several independent subproblems and get a near-optimal solution. For example, if the maximum number m is too big, we can run several passes. In each pass we allow a smaller number of simultaneous annotations. Unscheduled annotations are processed in the next pass until we reach the m simultaneous annotations.

In our experiment, we schedule 59 annotations in a 25 seconds video, with a uniform lifetime of 5 seconds and maximum number $m = 5$. Dividing it into two passes using $m1 = 3$ and $m2 = 2$, we solve the problem within two minutes in a mainstream PC, with about six million subproblems in total. Figure 8 shows the result after each pass.

We also compared with a greedy scheduling algorithm that schedules the annotation with the highest average score into the least crowded slot in every step. Our scheduling algorithm outperforms the greedy algorithm by about 10% in terms of annotation score.

Once annotations have been scheduled in the video, in each frame we lay them out spatially. We layout the annotations in the top area of the frame. Each annotation is then connected to the building's anchor point. Because the calibration is not accurate enough, there is visible inconsistency between the anchor point and the building. To alleviate the inconsistency, we check SIFT features close to the anchor point in each frame, and use the feature that belongs to the longest SIFT feature track across video frames as a reference point. Then the translation of the feature pair in successive frames is applied to the anchor point. A comparison in our accompanying video illustrates the benefit of this method. Without the adjustments, the annotations appear to float over the buildings. However, when attached to SIFT features, they stick to the buildings.

Figure 9: Annotations tracking in the video. From left to right, top to bottom, 4 frames of a video are shown. The frames are not consecutive to illustrate the changes in the annotation.

In Figure 9 we show how annotations change visibility over different frames of the video. More results are shown in our accompanying video.

6. INTEGRATED MAP AND VIDEO APPLICATION

Once annotated, the video is presented to the user in an integrated map application, as shown in Figure 10. The left side is a map that visualizes the video path and landmarks. The blue dot is the current camera position and the green frustum is its field of view. The camera track is rendered as a gradient line, which uses different color to represent the capture time. The user also has the option to visualize the

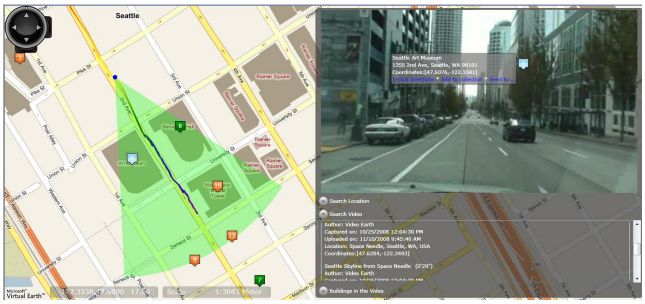


Figure 10: Integrated map and video application.

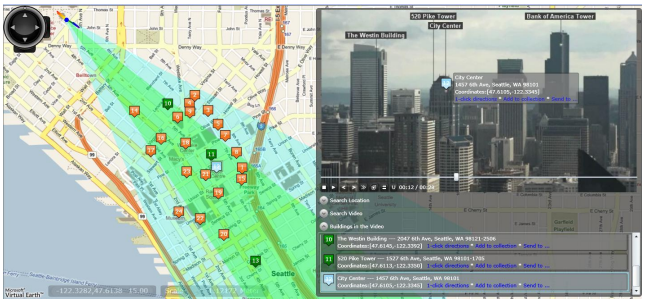


Figure 11: Visualizing the total coverage of the video. The total coverage is shown in light blue.

total coverage of the video, as shown in Figure 11. This visualization is useful for panoramic videos; the current frustum (in green) shows the subset of the panorama (shown in light blue). On the right side of the interface is the video itself, overlayed with annotations.

When the video is playing, we update the current camera position and its frustum. We also change the color of building annotations as their annotation status changes (e.g. scheduled to be shown or hidden). It is also possible to play the video at faster speeds to obtain a quick summary.

As discussed earlier, one advantage of integrating the map and video is to enable non-sequential video navigation. Using the application, there are 5 ways to navigate the video:

- play the video sequentially
- drag along the video trajectory on the map
- play the video within the lifetime of a building
- find the video frame that contains a query building
- find the video frame that matches a query frustum

The first three modes play the video sequentially. However, the second mode enables the user to quickly scan through the video by scrubbing along the video trajectory. In the third mode, the user can play the video cliplets that a particular building appears in. This interaction is exposed by a timeline visualization, as shown in Figure 12.

The last two modes, enable a user to quickly jump to a position in the video. A user can specify a query building by clicking on its thumbnail in the map. The video jumps to the frame with highest score containing the building. Alter-

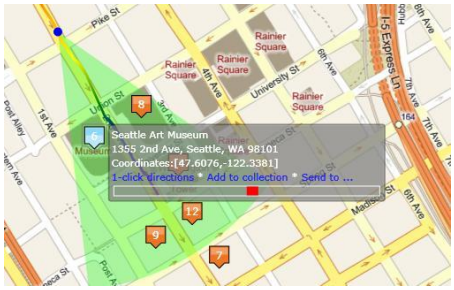
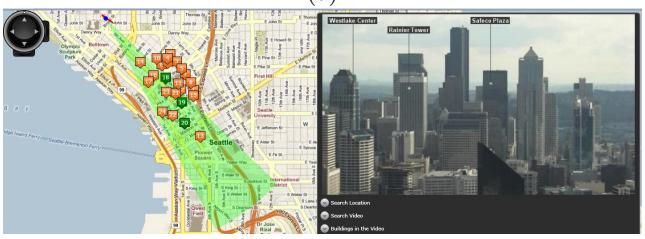


Figure 12: Displaying and navigating within a building's lifetime. A timeline of the entire video is shown as a horizontal bar. The building's lifetime is the red subset in this bar. The user can click in this red area to play that part of the video.

(a)



(b)

Figure 13: Specifying a query frustum and finding the closest camera. In (a) the user specifies a query frustum, shown in blue. In (b) the video jumps to the frame with the nearest frustum.

natively, the user can specify a query frustum and the video jumps to the frame closest to the query. Figure 13 illustrates this interaction.

Users may also add annotations by clicking on the building in the video. The system converts the 2D click into a ray using the current frame's calibration. This ray is intersected with the geo-referenced 3D geometry to find the target building. Once found, this building is highlighted on the map or added as a new annotation on the map (shown in Figure 14).

Finally, multiple videos can also be displayed in the same map (Figure 15). Users can have a better idea on how those videos are related to each other and switch between different video quickly.

The accompanying video illustrates more of the dynamic features of the user interface.

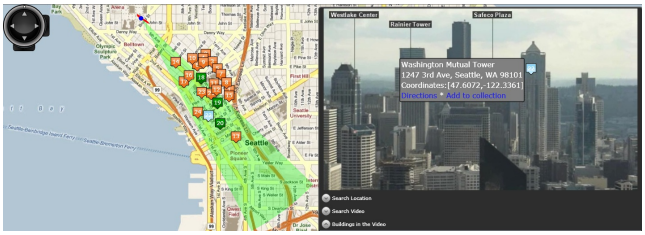


Figure 14: Adding annotations in the video. The user may also add annotations (indicated by the light blue tag) by clicking on the building.

7. USER STUDY

We conducted a user study to evaluate our system, focusing on the video presentation user interface. The main purpose is to learn from the users, finding what scenarios are more important to them, and what features are more useful. We also wanted to know how they used our system so that we can improve the user interface.

7.1 Experiment Setup

The experiment was run on a laptop computer with a 2.0GHz Intel Core 2 Duo T7300 CPU and 2GB RAM. The user interface was implemented as a SilverLight 2.0 application embedded in an ASP.NET web page, which was browsed in an IE7 browser on Windows Vista Enterprise, with a screen resolution of 1440*900. An external display was also connected to the laptop, with a duplicate screen to be recorded by a video camera to avoid disturbing the users during the test.

7.2 Participants

We had 12 participants with different professional and experiences. Their demographics information is listed in Table 1. All of them were not familiar with the place where the sample videos were captured. 11 participants used some kind of digital map service. 5 of them are frequent users. 5 used digital map with geo-tagged photos and 10 used video sharing services. 5 were familiar in shooting tourist video. But only one of them was a video editing expert and had experience in sharing video with others.

Students: 6 (1 undergraduate)	Employed: 6
Male: 8	Female: 4
Technical: 9	Non-tech: 3
Age 19-29: 9	Age 30-39: 3
Use computer 8+ hours/day:8	$\leq$ 8 hours/day: 4

Table 1: Participants Demographics

7.3 Test Session

The participants had individual sessions in a closed environment with one test coordinator. The whole session lasted about 40 minutes on average. We first explained the whole process briefly and asked each participant to fill a pre-test questionnaire. The test coordinator then explained our system in detail and demonstrated the features using a sample video. The participants had about five minutes to play around freely using the same video before they were given the task list. All the tasks were designed to use other video clips. A video camera recorded the external screen when they were working on the tasks. Their interactions were also logged by the application. After they finished the tasks, they were asked to fill another questionnaire. Finally, the test coordinator had a conversation with them to collect more feedback and reasons why they disliked some features.

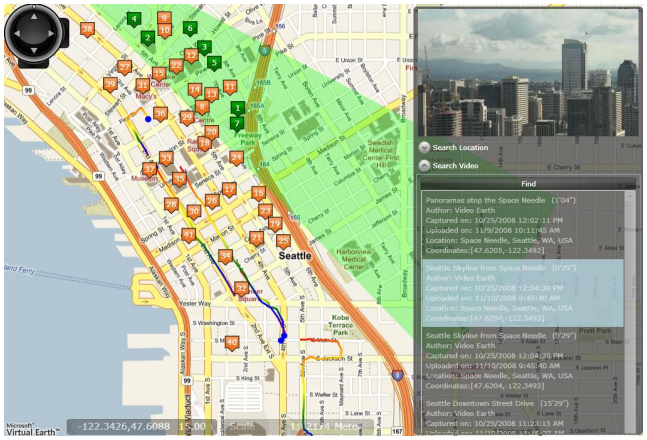


Figure 15: Displaying multiple videos in the same map. The primary video is fully visualized and only the camera track is displayed for other videos.

7.4 Pre-test Questionnaire

We mainly collected the participants' demographics information, their experience in using digital maps, geo-tagged photos and videos, video sharing services, and their levels of familiarity for video capturing, editing and sharing.

7.5 Tasks

There were five time-constrained tasks. The first three tasks used the "Seattle Skyline from Space Needle" video (28 seconds, opening video of our accompanying video). We had two versions of this video, one with numbered tags and one with text labels as building annotations. The participants could choose at their will. We recorded the time they used to accomplish each task.

- Given a picture of a building, find it in the video and locate it in the map. Write down the name of the building. (3 min)
- Given a building in the map (Bank of America Tower), find the most representing video frame with this building. Write down the time of the frame in the video. (3 min)
- Given the same building, quickly browse the video frames where this building appears. (1.5 min)

The fourth task used the "Seattle Downtown Street Drive" video (1'48", similar to the closing video of our accompanying video).

- Manipulate the map visualization of the video to jump to different part of the video. (2 min)

The last task asked the participants to revisit the two video clips again.

- Browse these two video clips freely. Describe each video in one sentence. (1 min each)

7.6 Post-test Questionnaire

In this part, we collected the participants' favor for different scenarios in video annotation, navigation and video-map interaction. They were also asked to give each feature a rank ranging from 1 to 5 in order of usefulness.

7.7 Results

Table 2 lists the result of the first three timed tasks. Most participants finished all the tasks successfully. However, their speed varied a lot.

Task	Success	Time Range	Average	Std. Dev.
1	10	35-170	87	36
2	10	25-145	76	43
3	12	20-80	46	18

Table 2: Task execution result (time unit: seconds)

The purpose of the first two tasks is to examine if users can connect the video with the map naturally. This is a bit different from their past experiences. We found most participants did this well but there were still several participants who worked on them separately.

For the first task, most participants used the slider in the video player to browse the video and find the matching building. However, there were three participants who used the frustum to quickly browse the video and found the matching building. They could scan the whole video without missing important buildings. After finding the building, most participants clicked on it in the video and found the associated highlighted building tag in the map immediately. However there was one participant who looked for the building tags in the map manually, because she didn't realize that one click could solve this problem. This problem suggests that we show more visual affordances on the video itself. Perhaps a 'hover' mode for the mouse that highlights buildings could alleviate such problems.

For the second task, our purpose was to learn whether the users were good at locating a known location after the map view changed. Although we warned all the participants that the map would change when the video opened and they should remember the location and use it to locate the building in the video, and they did try different methods to remember the location, only four of them successfully relocated the building in the map quickly. Two participants found the building by text annotations in the video. That would be time-consuming if the video was long. Others just looked at each building tag in the map, resulting in four successes and two failures (found a buidling with similar name). This problem suggests that we should not change the map view as we open a video. It affects many users who are not good at locating.

For the third task, we expected the participants to use the timeline in the building information box. We introduced this feature in the demonstration part. However, eight participants didn't use the timeline at all for this task (one had a look at the timeline and gave up). Six of them used the slider in the video player and two used the frustum. Four participants used the timeline but one finally reverted to using the frustum. The problem may be that the visual affordance for the timeline is entirely new to the users. This suggests an alternative design where the timeline is shown as a subset of the slider on the video player. If this subset is highlighted when a building is selected, this may better indicate the building's lifetime in the video.

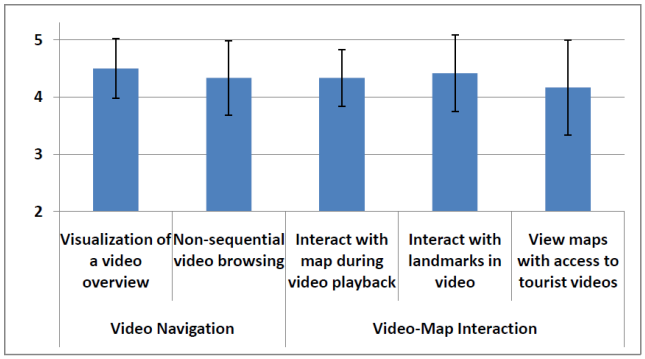


Figure 16: What did users like? (using a Likert scale: 5 = best; error bars are standard deviations.)

For the fourth task, we just wanted to watch how the participants would behave when they were given various video visualization components in the map and had no specific task in hands. Most of them tried almost every visualization component in the map. The common features they used most are dragging the camera position, dragging the frustum, and clicking building tags. Some of them were a little confused by the frustum query.

For the fifth task, we wanted to watch how they would behave when they had a specific task. Some of them just let the video play, and paid close attention to the visualization updates on the map. One participant just compared the visualization of the two videos and gave two accurate descriptions.

Figure 16 illustrates participants' attitude towards each scenario. They had quite similar attitudes for the first four scenarios but somewhat disagreed with each other on the last scenario. Three participants thought they would not have strong need for tourist videos when they browsed a map.

Figure 17 illustrates their ranking about the usefulness of the features. Most features were ranked high by many participants. But there were a few low scores and we also collected comments from the participants:

- Text annotation could be annoying or distracting.
- User might have no interest in camera track, especially for panoramas.
- Frustum query was not as quick as clicking a building tag to jump to a best frame.
- Don't always switch to best map view when opening a video. Make it an option.

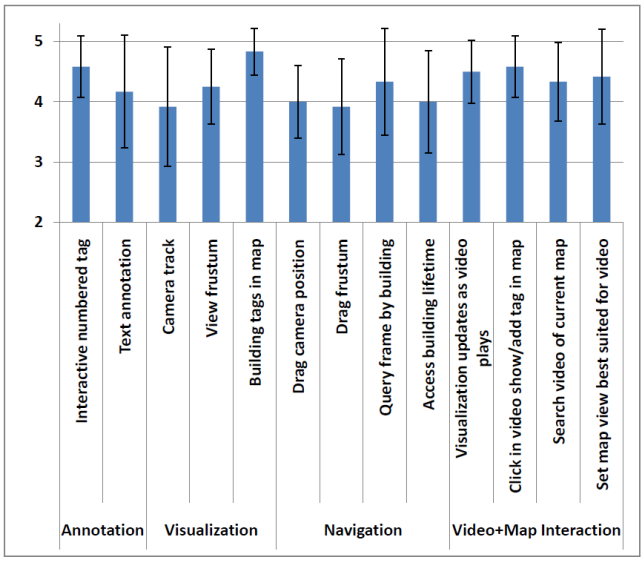


Figure 17: Feature ranking chart (using a Likert scale: 5 = best; error bars are standard deviations.)

7.8 Discussion

All the participants expressed great interest in our system. They liked the idea of connecting the tourist video with the map. It provided them with brand-new video browsing experiences, from both the navigation and the interaction point of views.

Our system provided novice map users a better understanding through interaction between the map and the video. Although none of the participants were familiar with the location and many of them were infrequent digital map users, most participants accomplished their tasks. One participant has never used a digital map. She was slower and sometimes couldn't find the best way. But she still managed to finish all the tasks successfully, and learned to make good use of many simple features.

Our application had many features and we only had several minutes to demonstrate them through a short sample video. It was quite a burden for the participants to remember every feature, especially for those non-technical and / or inexperienced participants. However, they were still able to make good use of many features. They liked those simple or familiar features that have a similar usage in other applications. One good example was the building tag: it was simple and already exist in many map applications. We did have features that were very different from participants' past experiences. For example, although they found frustum query very useful, many mistook it as the view direction feature seen in some street view maps and got frustrated when they didn't get expected result. After we explained the difference, they still wanted us to improve it. Another example is the building lifetime. Most participants thought it was good when we introduced it. But they still reverted to frustum query or the timeline slider in the video player when they wanted to quickly browse the frames with the building.

People like annotations, visualizations, and more controllable components, but not in all situations. That's part of the reason why people were not in great favor of the visible frustum because it sometimes occupied too much map space. They wanted more intelligent placement of all those components. One possible improvement is that we can provide multiple view modes, each of which contains a different setup tailored for a certain scenario or certain level of user. Users should also be able to switch between the modes quickly using hot keys.

Another interesting topic is how to combine multiple types of media together. With a digital map, tourist videos, high quality pictures, text and voice annotations, and even more materials online, we can think about how to make the best use of them. It also answered many participants' complaints about the low quality video: with the help of a high quality picture, it would be easier for them to find the building in the first task.

In summary, we learned several design principles that are also useful for other video and map application designers:

- A deep intergration of video with maps can improve users' navigation in both the video and the maps.
- Use simple and familiar features. Beware of features that look similar but have different meanings.
- Provide multiple modes for different levels of users, or different scenarios. Switching between modes should be easy and intuitive.
- Make good use of other materials in addition to videos and maps.

8. CONCLUSION

We have presented a system for registering videos to geo-referenced 3D models and propagating calibration from key frames in the video. We show how to schedule the visibility of annotations in the video without too much clutter while maximizing the number of annotations.

Annotated, geo-referenced videos enable new ways of video navigation, as evidence by our map application. Our application enables a viewer to browse the video by semantic objects (i.e. buildings and other visually prominent objects with 3D models) and also by interacting with the map.

However, our system is not without its limitations. With regards to propagating calibration, our algorithm fails when the video turns too quickly or moves too fast. Also, when the camera zoom level is high, we lose accuracy in positioning the camera. Figure 18 shows a failure case. In these cases more manual work is necessary in specifying key frames to lock down the video.

Another limitation is that our 3D models do not model trees, vehicles or other similar objects. This can impede calibrating key-frames and can confuse the propagation. Again, the result is to manually specify more key frames.

With regards to the video annotation, it is still difficult to get very stable annotations, due to the error of SIFT feature matching.

However, even with these limitations, we have found that

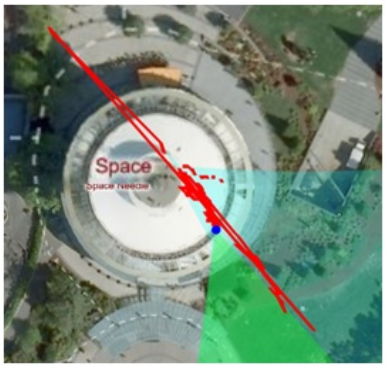


Figure 18: Failure case in calibration propagation. The red plot shows the computed camera position. Clearly the video was captured at the top of the Seattle Space needle. The calibration failed due to the high-zoom of the camera. More key frames were specified.

people can still obtain useful information from the noisy calibration. Also, for navigating with the map, this is fairly robust to miscalibration.

The world is progressing into more geo registered sensors. The new iPhone 4 already includes a GPS, accelerometers, a compass and a gyro. As the quality and avialbility of camera phones increases we will see more geo-positioned tourist videos. Also, with the help of automatic matching of user photos to geo-positioned street photos (such as Google street views, and Microsoft StreetSide)[10], the calibration process can be done fully automatically.

The geo-positioning of videos allows mapping from the map to the video frames, and annotation of the videos, but it has other benefits too. It can be used to measure the interests of the tourists: the more interesting an object is in a scene, the more likly it is to be photographed by multiple photographers and in longer video clips. Accumulating the frustrum of the frames, on the map, allows the discovery of interest locations, and allows more semantic organization of the videos (following [6] and [17]).

In addition, we would like to layout the annotations better and make them more stable across the frames. Annotations may originate from the map as displayed in this paper, but also from geo-positioned media (from past geo-positioned photos and videos[16]). This leaves a place for more tags to enter the space, directly from the media we are geo-positioning.

We believe that in the near future, using such tools as presented in this paper will rapidly facilitate the accumulation of geo-positioned video imagery as well as user annotation. This metadata in turn, can be fed back into the system to better enrich the experience for future users.

9. REFERENCES

[1] S. A. Ay, R. Zimmermann, and S. H. Kim. Viewable scene modeling for geospatial video search. *ACM International Conference on Multimedia*, pages 309–318, 2008.

[2] R. E. Bellman. *Dynamic programming*. Princeton University Press, 1957.

[3] B. Chen, G. Ramos, E. Ofek, M. Cohen, S. Drucker, and D. Nister. Interactive techniques for registering images to digital terrain and building models. Technical report, Microsoft Research, 2008.

[4] L. Cheong and H. Huo. Shot change detection using scenebased constraint. *Multimedia Tools and Applications*, 2001.

[5] P. L. Cho. 3D organization of 2D urban imagery. *Applied Image Pattern Recognition Workshop*, 0:3–8, 2007.

[6] B. Epshtein, E. Ofek, Y. Wexler, and P. Zhang. Hierarchical photo organization using geo-relevance. In *ACM International Symposium on Advances in Geographic Information Systems*, pages 1–7, New York, NY, USA, 2007. ACM.

[7] Google. Google Maps. http://maps.google.com.

[8] W. Heng and K. Ngan. An object-based shot boundary detection using edge tracing and tracking. *Journal of Visual Communication and Image Representation*, 2001.

[9] J. Kopf, B. Neubert, B. Chen, M. Cohen, D. Cohen-Or, O. Deussen, M. Uyttendaele, and D. Lischinski. Deep photo: Model-based photograph enhancement and viewing. *ACM Trans.on Graphics (Proceedings of SIGGRAPH Asia 2008)*, 2008.

[10] M. Kroepfl, Y. Wexler, and E. Ofek. Efficiently locating photographs in many panoramas. *submitted to ACM GIS 2010*, 2010.

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

[12] Q. Luan, S. M. Drucker, J. Kopf, Y.-Q. Xu, and M. F. Cohen. Annotating gigapixel images. In *ACM Symposium on User Interface Software and Technology*, 2008.

[13] Microsoft. Bing Maps. http://maps.bing.com/.

[14] H. Nicolas, A. Manury, J. Benois-Pineau, W. Dupuy, and D. Barba. Grouping video shots into scenes based on 1D mosaic descriptors. *Proc. Intl. Conf. on Image Proc.*, 2004.

[15] S. Pongnumkul, J. Wang, and M. Cohen. Creating map-based storyboards for browsing tour videos. In *ACM symposium on User Interface Software and Technology*, pages 13–22, New York, NY, USA, 2008. ACM.

[16] T. Rattenbury and M. Naaman. Methods for extracting place semantics from Flickr tags. *ACM Trans.on Web*, 3(1):1–30, 2009.

[17] I. Simon, N. Snavely, and S. M. Seitz. Scene summarization for online image collections. *IEEE International Conference on Computer Vision*, 2007.

[18] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM SIGGRAPH 2006 Papers*, pages 835–846, New York, NY, USA, 2006.

[19] K. Toyama, R. Logan, and A. Roseway. Geographic location tags on digital images. *ACM International Conference on Multimedia*, 2003.