DETECTING PATH INTERSECTIONS IN PANORAMIC VIDEO

Xinding Sun, Don Kimber$^+$, Jonathan Foote$^+$, B. S. Manjunath**

*Department of Electrical and Computer
Engineering, University of California
Santa Barbara, CA 93106
{xdsun, manj}@ece.ucsb.edu

$^+$FX Palo Alto Laboratory, Inc.
3400 Hillview Avenue,
Palo Alto, CA 94304
{kimber,foote}@fxpal.com

ABSTRACT

Given panoramic video taken along a self-intersecting path, we present a method for detecting the intersection points. This allows "virtual tours" to be synthesized by splicing the panoramic video at the intersection points. Spatial intersections are detected by finding the best-matching panoramic images from a number of nearby candidates. Each panoramic image is segmented into horizontal strips. Each strip is averaged in the vertical direction. The Fourier coefficients of the resulting 1-D data capture the rotation-invariant horizontal texture of each panoramic image. The distance between two panoramic images is calculated as the sum of the distances between their strip texture pairs at the same row positions. The intersection is chosen as the two candidate panoramic images that have the minimum distance.

1. INTRODUCTION

A useful application of panoramic video is the FlyAbout system developed by Kimber et al. [4], where panoramic video is spatially indexed for interactive navigation. Figure 1 shows the FlyAbout interface, which allows the user to browse a 360° panoramic video, both by map-like and car-like interfaces. In the FlyAbout system, location data is acquired from a Global Positioning Satellite (GPS) receiver time-synchronized with the panoramic video. The video is taken along arbitrary paths, such as city streets, and therefore will contain "path intersection" images of an identical location that occurs at different times in the panoramic video. Figure 2(a) shows schematic panoramic video frames recorded on two intersecting streets X and Y.

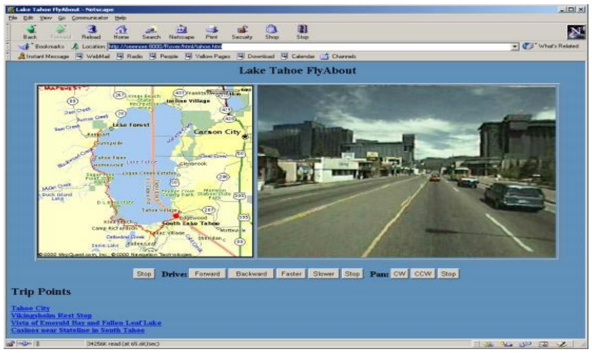
Lake Tahoe FlyAbout - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://seemore:8000/Rover/html/tahoe.html
What's Related
Instant Message WebMail Radio People Yellow Pages Download Calendar Channels
Lake Tahoe FlyAbout
MAPQUEST
Incline Village
Lake Forest
Carson City
Glenbrook
South Lake Tahoe
Stop
Drive:
Forward
Backward
Faster
Slower
Stop
Pan:
CW
CCW
Stop
Trip Points
Tahoe City
Vikingsholm Rest Stop
Vista of Emerald Bay and Fallen Leaf Lake
Casinos near Stateline in South Tahoe
34256K read (at 65.6K/sec)

Figure 1. Flyabout interface.

While the video clips taken at the two paths have a physical intersection, it is not directly available from the data. While it could be determined by manual inspection, this is clearly not practical, as a real system might have hundreds or thousands of intersections. The GPS location can't be acquired at a rate necessary to locate every frame; even for frames that have location data the location estimate is noisy. Thus there is a need for detecting intersection frames directly from the panoramic images. This is useful for a number of applications, for example "virtual turn synthesis." In the intersection of an east-west street with a north-south avenue, we need only to record the video in the north-south and east-west directions. By synthesizing turns we can offer the user any possible turn in that intersection, from any street or avenue onto any other, from any direction.

There are several challenges involved in this problem. Roads are typically "crowned" to afford drainage, and are thus not perfectly flat. Thus frames will not align exactly at intersections unless traveled in exactly the same direction and lane. Also, outdoor light changes with time and weather conditions, which makes it difficult to obtain panoramic images under the same illumination condition. There may also be image differences due to moving objects like vehicles or people. Other factors include image warping and other artifacts inherent in many panoramic camera systems [1], not to mention the noise inherent in digital imaging.

In terms of panoramic video applications, similar but different problems can be found in early research on robot navigation. Stein and Medioni [8] use panoramic curves from the sky-ground boundary for localization. It requires curve data at different locations to be pre-computed and stored before localization. It thus cannot be practically applied here since there is no data model available. Jogan and Leonardis [2] use panoramic eigenimages for spatial localization. Their work is based on the training using zero phase representation (ZPR) of panoramic images proposed

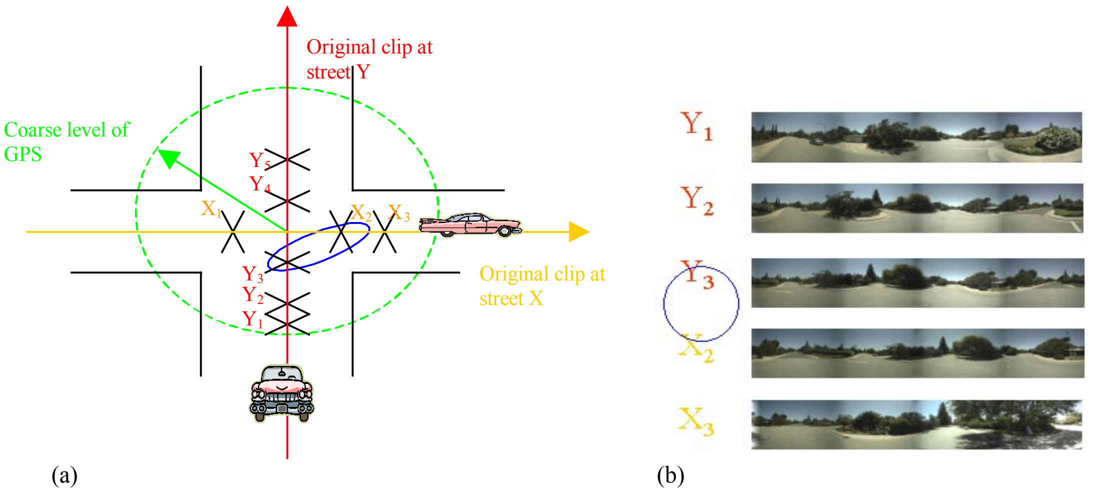

Figure 2. (a) Path intersection illustration. (b) Images extracted from two paths.

by Pajdla and Hlavac [6]. Jogan and Leonardis [3] propose localization based on feature points. The two methods will not work here either, because there are no training data available in this case.

The problem addressed here falls into the image-matching category. Many methods have been proposed for image matching, for example the descriptors described in the MPEG7 standard [10]. Swain and Ballard [9] propose a way of image matching based on histogram analysis; Manjunath and Ma[5] propose image matching based on texture analysis; Smith and Chang [7] propose image matching based on image spatial information. They design feature description for general usage. However, the panoramic image used here has a typical column-wised periodic feature, which provides a strong constraint on the image content. Therefore, a more effective matching method is possible.

In this paper we propose a method for finding path intersections based on video-image content only. The candidates are chosen based on the GPS data first. Detection of the spatial intersection between two paths is posed as the matching of candidate panoramic images from two paths. Each panoramic image is segmented into strips. The 1-D textures of the strips are used to represent the panoramic image. The distance between two candidate panoramic images is calculated as the sum of the distances between their strip texture pairs at the same row positions. The intersection is chosen as the two candidate panoramic images that have the minimum distance.

2. INTERSECTION DETECTION

2.1. Narrowing down searching area using GPS data

Figure 3 shows example GPS data collected in a city neighborhood. The horizontal axis corresponds to the longitude and the vertical axis corresponds to the latitude.

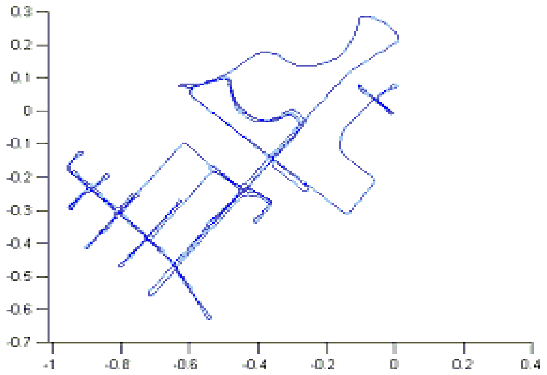

Figure 3. An example camera path, plotted by GPS latitude and longitude.

GPS data is synchronized with the panoramic video, and allows a coarse level of intersection detection. Intersections can be found without the GPS data, but the availability of the data greatly improves the solution by narrowing the possible search candidates. The GPS location can't be acquired at a rate necessary to locate every frame; even frames that have location data the location estimate is noisy. Thus there is still need for detecting intersection frames from the frame image data after processing of the GPS data, and it is discussed in the following subsection.

2.2. Intersection detection based on image matching

GPS data narrow down the detection area to two sets of candidate panoramic images from two paths at the intersection. For example, in Figure 2(a), X_1, X_2, X_3, are in the same set taken at street X; $Y_1, Y_2, ..., Y_5$, are in the same set taken on street Y. Locating the intersection can then be posed as the following manner:

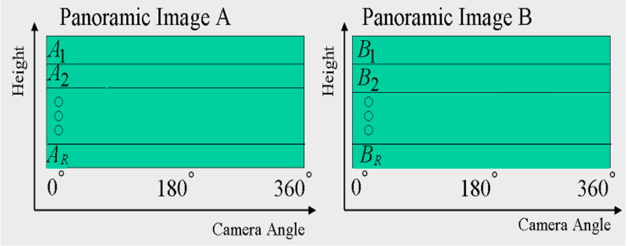

Figure 4. Segmenting panoramic images into 1-D texture strips.

Given two groups of candidate panoramic images:
$X = \{X_1, X_2, ..., X_M\}$,
$Y = \{Y_1, Y_2, ..., Y_N\}$,
Where M and N are the number of candidates of X and Y respectively, we wish to find the two frame indexes $i*$ and $j*$ that are closest, given some distance measure, to each other.

$$(i^*, j^*) = \underset{(1\le i\le M, 1\le j\le N)}{\text{argmin}} \ (\text{Distance}(X_i, Y_j)) \qquad (1)$$

2.2.1. The 1-D texture of a panoramic image

If a panoramic camera rotates around a fixed axis of projection, the captured panoramic images repeat periodically every 360°. Two panoramic images taken at the same spatial location with the same axis of projection differ only in the phase of camera angle at each height position. Here camera angle corresponds to the column direction, and height corresponds to row direction of panoramic images.

Because the magnitudes of the Fourier transform are invariant to phase, they make excellent rotation-invariant features for matching purposes. Note that in this case, the computation of spectra does not require the step of converting panoramic images into ZPR [6] format as posed in [2], because it has basically the same magnitude spectra as the original panoramic image. Thus a straightforward comparison of the row-wise Fourier spectra could serve as the distance measure in (1).

However, some practical issues prohibit direct application of such spectral matching. For example, when the axes of projection of the camera are not the same for two panoramic images, they are no longer the same at the same row position. The displacement of the position of camera location also causes changes in panoramic images. These changes can be considered as noise during the capturing process. One solution is to consider the one-dimensional texture across a small strip of the image, formalized as follows:

For a given panoramic image with H rows and W columns, it is first segmented row-wise into R strips. For example, in Figure 4 panoramic image A and B are segmented into R strips, each with H/R (rounded to an integer) rows. Then the image intensity of each strip is averaged at each column position to produce an "average row". Each value is the average of pixels in the same column position in the strip. That is, for an panoramic image strip matrix:

$$Ps = \{P_{i,j}\}, 1 \le i \le H/R, 1 \le j \le W,$$

where $P_{i,j}$ is the pixel value at position (i, j) in the strip, the corresponding average rows are:

$$P_{sAV} = (P_1, P_2, ...P_j, ...P_W), where\ \mathrm{P}_j = \frac{R}{H}\sum_{i=1}^{H/R} P_{i,j}\ . \qquad (2)$$

In Figure 4, $A_1, A_1, ..., A_R$ and $B_1, B_1, ..., B_R$ are used to represent these average rows for panoramic image A and B respectively. In the end, the 1-D texture of a strip is computed as the magnitude spectra of the averaged row. This can be efficiently computed using the Fast Fourier Transform. For example, S_{A_i}, the 1-D texture-coefficient vector of A_i can be computed as $S_{A_i} = \|FFT(A_i)\|$.

Here, vertically averaging each image strips smoothes the row images, and improves the image matching robustness. The idea of 1-D texture is similar to those used for 2-D texture analysis [5]. The added complexity of 2-D texture is not necessary for good similarity measurements.

2.2.2. Distance measure for 1-D texture

The distance between two panoramic images is computed from the distance between corresponding average rows. Given two average strips A_i and B_j, suppose their 1-D texture-coefficient vectors are S_{A_i} and S_{B_j}, then their distance measure is defined as:

$$\text{Distance}(A_i, B_j) = \frac{\left\|S_{A_i} - S_{B_j}\right\|_M}{\text{Max}(\left\|S_{A_i}\right\|_M, \left\|S_{B_j}\right\|_M)} \qquad (3)$$

Where $\left\|S_{A_i} - S_{B_j}\right\|_M$ is the Euclidean distance between the middle band of S_{A_i} and S_{B_j}, where $\left\|S_{A_i}\right\|_M$ and $\left\|S_{B_j}\right\|_M$ are the amplitude of vector S_{A_i} and S_{B_j} in middle band.

In practice, the texture coefficients are truncated to middle frequencies, for a number of reasons. First, the DC and low-frequency components of an image spectrum is quite sensitive to changes in illumination. Second, the high frequencies tend to be noisy and are thus not useful for representation. Also, removing the high frequencies also helps to reduce the effect of object occlusion in a scene. $\text{Max}(\left\|S_{A_i}\right\|_M, \left\|S_{B_j}\right\|_M)$ is used to normalize the distance. It is especially important when several pairs of average rows are used for comparison. In that case, the distance measure will not be biased toward pairs that have higher mid-frequency energy.

Since the distance measure proposed in (3) is already normalized, the distance between two panoramic images can then be computed as the sum of distances between 1-D texture pairs at the same row positions. It is formulated as:

$$\text{Distance}(A,B)=\sum_{i=1}^{R}\text{Distance}(A_i,B_i) \qquad (4)$$

Based on this distance measure, (1) can be used to detect the intersection of the paths.

3. EXPERIMENTAL RESULTS

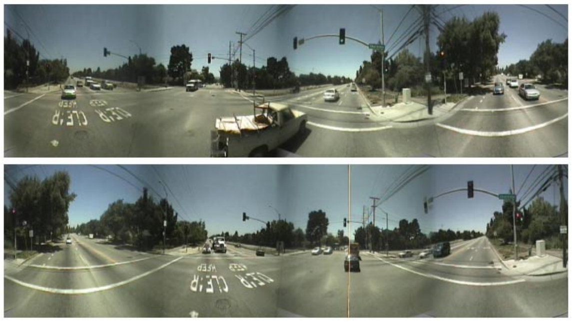

Figure 5. A street intersection with large object occlusion.

A panoramic camera and a GPS receiver were mounted on a car to collect data. The car's speed was typically 10 MPH when crossing intersections. The time difference between the times when the car crosses the same intersection varies from less than a minute to half an hour. On several occasions the car passes the same intersection more than twice.

The candidate panoramic frames were extracted from each path, limited to within 3 to 10 seconds of the estimated intersection point derived from the GPS data. Each panoramic frame was composited from four digital video frames, with a resultant dimension of 2384x448 pixels. Every image is segmented into 8 strips, and the texture coefficients are computed using the FFT. After normalization, bands 2 to 100 were used for distance measure.

Figure 2(b) shows the detected intersection at street X and Y, where X_2 and Y_3 are chosen as the intersection. The scene change is illustrated by the candidate images Y_1, Y_2, Y_3 and X_1, X_2. There is an angle difference between X_2 and Y_3, but the algorithm successfully locates the best match. Figure 5 shows two detected panoramic images at a major street intersection. The algorithm again successfully detects the correct intersection point despite the presence of a large truck in the image during the first pass through the intersection.

So far, the algorithm has been tested on the panoramic video at city neighborhood settings. Altogether, one hour of panoramic video was captured, including 20 intersections. The intersections from the video are first manually selected. Since it is quite subjective, if the computer-chosen frame at the intersection is within 2 frames around the manual pick, then it is considered to be correct. It is found that the algorithm detects the intersections correctly in all the cases.

4. CONCLUSION

Manual search of intersections in a panoramic video up to an hour is prohibitive in practice. A novel method for automatic path intersection detection is presented. Detection of intersection of two paths is posed as the matching of candidate panoramic images from two paths. The intersection is chosen as the two candidate panoramic images that have the minimum distance. The distance measure is based on 1-D texture representation of panoramic images. Experiments have been taken on street panoramic video. Results show the method proposed here is robust and has a promising application future.

Based on the work presented here, it is possible to use correlations to estimate angle difference between two crossing paths. This angle difference information can then be used to create virtual transition between paths. Other future work includes the detection of intersections from multiple paths at the same location.

Acknowledgement: The work presented here was conducted when the first author was a summer intern at FX Palo Alto Lab. The paper was written at UCSB, where he was supported in part by the following grants/awards: Office of Naval Research grant #N00014-01-1-0391, an NSF instrumentation award #EIA-9986057, and an NSF infrastructure award #EIA-0080134.

5. REFERENCES

[1] J. Foote and D. Kimber, "FlyCam: practical panoramic video and automatic camera control," *Proc. ICME'2000*, pp. 1419-1422, 2000.

[2] M. Jogan and A. Leonardis, "Panoramic eigenimages for spatial localization," *CAIP'99*, pp.558-567, 1999.

[3] M. Jogan and A. Leonardis, "Robust localization using panoramic view-based recognition," *ICPR'2000*, v4, pp.136-139, 2000.

[4] D. Kimber, J. Foote, and S. Lertsithichai, "FlyAbout: Spatially Indexed Panoramic Video," *Proc. ACM MM'2001*, 2001.

[5] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Tran. PAMI*, 18(8), pp. 837-842, 1996.

[6] T. Pajdla, and V. Hlavac, "Zero phase representation of panoramic images for image based localization," *CAIP'99*, pp.550-557, 1999.

[7] J. R. Smith and S. –F. Chang, "Integrated spatial and feature image query," *Multimedia Systems*, 7(2), pp.129-40, 1999.

[8] F. Stein and G. Medioni, "Map-based localization using the panoramic horizon," *IEEE Tran. Robotics and Automation*, 11(6), pp.892-896, 1995.

[9] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, 7(1), pp. 11-32,1991.

[10] URL: http://www.cselt.it/mpeg