Fast Trajectory Matching Using Small Binary Images

Wei Zhuo[1,2], Dirk Schnieders[1], and Kwan-Yee K. Wong[1]

[1] Department of Computer Science, The University of Hong Kong, Hong Kong
[2] School of Electric and Information Engineering, Xian Jiaotong University, China

Abstract. This paper proposes a new trajectory matching method using logic operations on binary images. By using small binary images we are able to effectively utilize the large word size offered in modern CPU architectures, resulting in a very efficient evaluation of similarities between trajectories. The efficiency is caused by the fact that all bits in the same word are processed in parallel. Representing trajectories as small binary images has other advantages, such as a low space requirement and good noise resistance.
The proposed method is evaluated on a publicly available dataset, and is compared to the more sophisticated Longest Common Subsequence (LCSS) method. In addition, synthetic experiments show the good efficiency and accuracy of the proposed method, enabling real time trajectory retrieval on databases with millions of trajectories.

Keywords: Trajectory Matching, Video Surveillance, Image Processing

1 Introduction

Governments and private institutions capture and store large quantities of video surveillance footage for later retrieval. Video surveillance has received great attention in past decades and robust object segmentation and tracking methods are available. The motion path that an object takes in time, i.e. its *trajectory*, can be accurately determined from video surveillance footage. Trajectories provide a rich source of information and can be used to classify an object's behavior and discover objects that followed a certain motion characteristic or moved in a similar way. For instance, by comparing trajectories of newly tracked objects with those trajectories previously obtained, abnormal behavior can be detected. In these types of applications it is crucial to query large quantities of tracking data both quickly and accurately.

In this paper, we propose a novel trajectory matching method using logic operations on binary images. Instead of using raw object trajectory data, i.e. a sequence of image points, we approximate trajectories with small binary images. By doing so we are able to save significant cost in both time and space at the expense of negligible loss in accuracy. The cost saving in time is achieved because binary images allow us to design an effective cost function for comparison of trajectories which effectively utilizes a large word size in modern hardware

architectures. On current hardware architectures, 64 pixels in the binary images are compared to a query binary image in parallel. As a result, we are able to quickly estimate similarities between trajectories. The cost saving in space follows from the fact that the raw trajectory representation typically requires significantly more space compared to our small binary image representation.

The rest of this paper is organized as follows. Sect. 2 reviews the existing literature on trajectory matching and discusses the differences between existing work nd the proposed method. In Sect. 3, we introduce the trajectory representation and in Sect. 4 we propose the similarity measure. In Sect. 5 we present experimental results on synthetic and real data, followed by conclusions in Sect. 6.

2 Literature review

The most widespread prior trajectory representations include chain codes, piecewise linear approximations, polynomials and splines. Chain code [1] is a simple representation of trajectories, consisting of piecewise linear segments encoded with 8 basic directions. Other representations, like splines and polynomial interpolation, are more sophisticated, however often require a more complex computation and are therefore less efficient. Others attempt to represent motion trajectory in a feature space. For example, Bashir et al. [2] segmented each trajectory into multiple atomic sub-trajectories and characterized each sub-trajectory by its principle component. Zhou et al. [3] extracted features using tensor subspace analysis and then performed standard mean shift to cluster the trajectory features. Chen et al. [4] introduced null-space representation which is invariant to viewpoints. Chen and Chang [5] proposed a wavelet-based approach that decomposed a trajectory into sub-trajectories which were described by a feature vector that includes the acceleration, initial velocity, arc length, order and multi scale edge points. In contrast to the existing work above, we represent a trajectory as a small binary image.

To retrieve similar trajectories, a proper matching technique and similarity metric should be developed. The similarity metric will depend on the trajectory representation. Several types of trajectory matching methods have been proposed in the literature, and we will briefly discuss the following: Dynamic Time Warping (DTW), Longest Common Subsequence (LCSS), vector matching and statistical matching. DTW [6] is a dynamic matching technique that uses dynamic programming and has the advantage of being conceptually very simple, but unfortunately is quite sensitive to noise. LCSS [7] has been shown to achieve a very high accuracy and we will use it for experimental comparison in this paper. Vector matching techniques perform matching using a similarity metric. Commonly used distances include Euler-distance, city-distance and Minkowski-distance. Other methods, such as [8] and [9], employ statistical matching techniques to find appropriate trajectories.

The existing methods in the literature achieve quite remarkable trajectory matching accuracy and are suitable to obtain similar (scale, translation and rotation invariance) trajectories on small databases. However with the increasing

amount of tracking information, comparing millions of database items quickly is very desirable. To achieve such a high performance we assume scale, translation and rotation variance in our approach, which is of particular interest in video surveillance where the camera is stationary. A strong robustness to noise is achieved by utilizing relatively small binary images.

3 Trajectory Representation

Let a raw trajectory be given as a sequence of absolute 2D points in an image of size $m' \times n'$

$$\Gamma = [\, x_0 \; y_0 \,], [\, x_1 \; y_1 \,], \;\; \ldots \;\; , [\, x_e \; y_e \,]. \tag{1}$$

We represent this trajectory as an 8-adjacent path in a small $m \times n$ binary image

$$\mathbf{I} = \begin{bmatrix} I_0 & I_1 & \ldots & I_{n-1} \\ I_n & I_{n+1} & \ldots & I_{2n-1} \\ \vdots & \vdots & & \vdots \\ I_{(m-1)n} & I_{(m-1)n+1} & \ldots & I_{mn-1} \end{bmatrix}. \tag{2}$$

To consistently transfer a trajectory from the modality in Eq. (1) to our modality in Eq. (2), we assume a constant aspect ratio, i.e. $m/n = m'/n'$. In our approach, hardware bit-level parallelism is effectively exploited by representing $\mathbf{I}$ as a *bit vector*

$$\mathbf{T} = \left[\, \mathbf{w}_0 \; \mathbf{w}_1 \; \ldots \; \mathbf{w}_k \,\right], \tag{3}$$

which consists of k words of length w given by

$$\mathbf{w}_i = \left[\, I_{wi} \; \ldots \; I_{w(i+1)-1} \,\right] \quad 0 < i < k. \tag{4}$$

To avoid internal fragmentation we let mn be a multiple of w. This representation is motivated by the fact that on modern hardware w is usually quite large, i.e. on 64-bit architectures $w = 64$, resulting in the following key benefits of our approach:

1. a relative small number of words per trajectory, and
2. an efficient computation by processing w bits in parallel.

4 Trajectory Matching

A simple approach for matching a query trajectory $\mathbf{T}$ to a target trajectory $\mathbf{T}'$ is to maximize the overlapping trajectory segments. This can be achieved by formulating a similarity measure

$$\begin{aligned} s(\mathbf{T}, \mathbf{T}') &= |\mathbf{T} \wedge \mathbf{T}'| \\ &= |\mathbf{w}_0 \wedge \mathbf{w}'_0| + \ldots + |\mathbf{w}_k \wedge \mathbf{w}'_k|, \end{aligned} \tag{5}$$

where $|\mathbf{w}|$ represents the population count operation, i.e. the number of set bits in the word $\mathbf{w}$, and $\wedge$ represents the bitwise AND operator. This equation is motivated by the fact that both bitwise AND and population count are available on modern microprocessors enabling us to evaluate Eq. (5) quickly. We find the trajectory that maximizes this similarity measure as

$$\hat{\mathbf{T}} = \max_{\mathbf{T}'} \; s(\mathbf{T}, \mathbf{T}'). \tag{6}$$

Unfortunately, the dissimilarity measure in Eq. (5) is not optimal because it will only measure the length of overlapping trajectory segments but it completely ignores non-overlapping trajectory segments. Alternatively, we may formulate a dissimilarity function

$$\begin{aligned} d(\mathbf{T}, \mathbf{T}') &= |\mathbf{T} \oplus \mathbf{T}'| \\ &= |\mathbf{w}_0 \oplus \mathbf{w}'_0| + \ldots + |\mathbf{w}_k \oplus \mathbf{w}'_k|, \end{aligned} \tag{7}$$

where

$$\mathbf{w}_i \oplus \mathbf{w}'_i = (\mathbf{w}_i \vee \mathbf{w}'_i) \wedge \neg(\mathbf{w}_i \wedge \mathbf{w}'_i) \tag{8}$$

represents bitwise exclusive disjunction. Note that Eq. (7) measures how long the non-overlapping trajectory segments are, i.e. it measures the absolute length of those paths which appear in $\mathbf{T}'$ or $\mathbf{T}$ but not in both. This measurement is also known as the *Hamming* distance and determines the minimum number of bit substitutions required to transform $\mathbf{T}$ to $\mathbf{T}'$ or vice versa.

To find the best matched trajectory we minimize this dissimilarity function

$$\hat{\mathbf{T}} = \min_{\mathbf{T}'} \; d(\mathbf{T}, \mathbf{T}'). \tag{9}$$

Unfortunately, the similarity measure in Eq. (7) is also not optimal because it measures the absolute number of bit substitutions required. It does not directly take into account overlapping path segments and two paths that are non-overlapping may achieve a lower dissimilarity score than two paths that are partially overlapping. To overcome this problem we propose to indirectly take the overlapping path segments into account. Note that the dissimilarity function in Eq. (7) satisfies the following inequality

$$0 \leq d(\mathbf{T}, \mathbf{T}') \leq |\mathbf{T} \vee \mathbf{T}'|, \tag{10}$$

where $\vee$ represents bitwise OR. In this paper, we propose to optimize for the largest ratio between non-overlapping and all paths and find the best matched trajectory as follows

$$\hat{\mathbf{T}} = \max_{\mathbf{T}'} \frac{d(\mathbf{T}, \mathbf{T}')}{|\mathbf{T} \vee \mathbf{T}'|}. \tag{11}$$

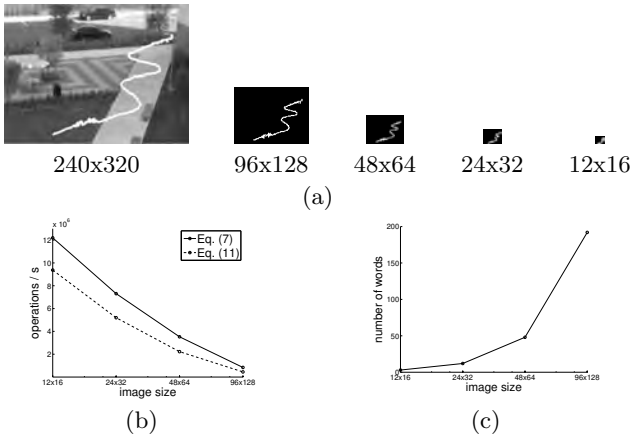

Fig. 1. Efficiency of the proposed method (a) Raw trajectory rendered on image obtained from a surveillance camera [10] and four binary images used in the proposed method (plotted to scale). (b) Theoretical efficiency of Eq. (7) and Eq. (11) using the binary images shown in (a) as the query. (c) Number of words used per binary image against image size.

5 Experimental Evaluation

5.1 Synthetic Data

We first measure the efficiency of the proposed method by measuring the number of times the matching function can be executed per second. This theoretical efficiency is expected to be very high due to the simplicity of the matching function and the availability of hardware implementations. We have implemented Eq. (5), Eq. (7), Eq. (11) and measured their efficiency on an Intel i7 CPU system that natively supported population count. We measured the performance in number of evaluations per second and plot results for different image sizes in Fig. 1. Since we found the performance of Eq. (5) and Eq. (7) to be equivalent we have only plotted the performance of Eq. (7). As expected, the efficiency of Eq. (11) is slightly less than Eq. (7) due to the additional operations performed. Nevertheless, it can be seen that Eq. (11) can be evaluated around 10 million times per second for small binary images of size 12×16. This large number follows from the fact that a binary image of size 12×16 just required $k = 3$ words.

In a second experiment we synthetically generated 30 original trajectories and generated 50 noisy variations for each of these trajectories. To generate noisy trajectory from a given raw trajectory, we first add uniformly distributed noise to the points of the trajectory. The magnitude of this noise was 20% of the standard deviation of all the trajectory points. A bezier-spline was subsequently fitted to these noisy trajectory points. Fig. 2 shows an example of an original trajectory and a synthetically generated noisy trajectory. The generated binary

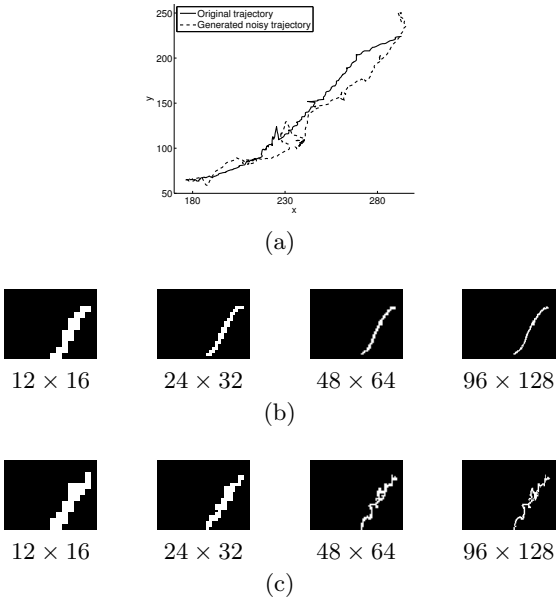

Fig. 2. Noisy trajectory generated for the synthetic experiment (a) An example of an original and noisy raw trajectory from the synthetic experiment. (b) The original trajectory represented as binary images. (c) The noisy trajectory represented as binary images.

	12×16	24×32	48×64	96×128
Eq. (5)	87%	91%	92%	89%
Eq. (7)	88%	86%	67%	41%
Eq. (11)	94%	97%	96%	95%

(a)

	12×16	24×32	48×64	96×128
Eq. (5)	89%	95%	99%	98%
Eq. (7)	98%	97%	82%	54%
Eq. (11)	98%	100%	100%	100%

(b)

Fig. 3. Accuracy of the proposed method (a) Percentage of top 50 noisy retrievals that belong to the same set as the query trajectory. (b) Percentage of top 10 noisy retrievals that belong to the same set as the query trajectory.

images are also shown for reference. There are a total of 1500 noisy trajectories that are clustered into 30 groups. We randomly selected a trajectory from the set of original trajectories and query the database to select the top 50 retrievals based on Eq. (5), Eq. (7) or Eq. (11). To test the accuracy of the proposed method, we determined how many retrievals belong to the same cluster as the query trajectory. We repeated this experiment for different sizes of the binary images. The results are shown in Fig. 3. The experiment has also been repeated using just the top 10 retrievals. It can be seen that reasonable good results can be obtained with a very small 12×16 binary image size and that results obtain by Eq. (11) are consistently better than those obtained by Eq. (7) and Eq. (5). This follows from the fact that neither Eq. (7) nor Eq. (5) perform normalization and only compare absolute similarity and dissimilarity respectively. We conclude that a small binary image of size 12×16 results in high accuracy and quick computation.

5.2 Real Data

We evaluated the proposed method on 3206 public available trajectories from the database reported in [10]. Given the high efficiency and accuracy, we generated small binary images of size 12×16 for all the trajectories in the database. Fig. 4 shows the top 10 retrieval results using Eq. (7), Eq. (5) and Eq. (11) for a specific query. To query the whole database took just ~ 0.5 milliseconds when using Eq. (11), whereas LCSS required several seconds. It can be seen that the results obtained by Eq. (7) and Eq. (11) are good and the latter performs superior because overlapping as well as non-overlapping trajectories are considered. Results retrieved by Eq. (5) are not as good because just overlapping trajectory segments are considered. Similarly, LCSS focuses on finding the longest common subsequence and does not always return desirable results.

6 Conclusions

In this paper, we presented a novel trajectory matching method using logic operations on small binary images. By approximating the trajectories using this novel representation, we were able to save significant cost in both time and space at the expense of retrieval accuracy that is negligible in many applications.

References

1. Kaneko, T., Okudaira, M.: Encoding of arbitrary curves based on the chain code representation. Transactions on Communications **33** (1985) 697–707
2. Bashir, F., Khokhar, A., Schonfeld, D.: Segmented trajectory based indexing and retrieval of video data. In: ICIP. (2003)
3. Zhou, H., Tao, D., Yuan, Y., Li, X.: Object trajectory clustering via tensor analysis. In: ICIP. (2009)

Fig. 4. Experimental results on real data: Retrieval results obtained on a query trajectory (a), and top 10 retrieval results using binary images of size 12×16 and Eq. (5), Eq. (7), Eq. (11) and LCSS[7] (b).

4. Chen, X., Schonfeld, D., Khokhar, A.: Robust null space representation and sampling for view-invariant motion trajectory analysis. In: CVPR. (2008)
5. Chen, W., Chang, S.: Motion trajectory matching of video objects. In: Electronic Imaging, International Society for Optics and Photonics (1999) 544–553
6. Hu, W., Tan, T., Wang, L., Maybank, S.: A survey on visual surveillance of object motion and behaviors. IEEE Transactions on Systems, Man, and Cybernetics **34** (2004) 334 –352
7. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. In: International Conference on Data Engineering. (2002)
8. Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. Image and Vision Computing **14** (1996) 583–592
9. Porikli, F.: Trajectory distance metric using hidden markov model based representation. In: ECCV Workshops. (2004)
10. Basharat, A., Gritai, A., Shah, M.: Learning object motion patterns for anomaly detection and improved object detection. In: CVPR. (2008)