Clustering Multidimensional Trajectories based on Shape and Velocity

Yutaka Yanagisawa
NTT Communicatin Science Laboratoriesy
NTT Corporation
2-4 Hikaridai, Seika, Soraku, Kyoto, JAPAN
yutaka@cslab.kecl.ntt.co.jp

Tetsuji Satoh
NTT Communicatin Science Laboratoriesy
NTT Corporation
2-4 Hikaridai, Seika, Soraku, Kyoto, JAPAN
satoh@cslab.kecl.ntt.co.jp

Abstract

Recently, the analysis of moving objects has become one of the most important technologies to be used in various applications such as GIS, navigation systems, and location-based information systems, Existing geographic analysis approaches are based on points where each object is located at a certain time. These techniques can extract interesting motion patterns from each moving object, but they can not extract relative motion patterns from many moving objects. Therefore, to retrieve moving objects with a similar trajectory shape to another given moving object, we propose queries based on the similarity between the shapes of moving object trajectories. Our proposed technique can find trajectories whose shape is similar to a certain given trajectory. We define the shape-based similarity query trajectories as an extension of similarity queries for time series data, and then we propose a new clustering technique based on similarity by combining both velocities of moving objects and shapes of objects. Moreover, we show the effectiveness of our proposed clustering method through a performance study using moving rickshaw data.

1 Introduction

In recent years, it has become possible to continuously track various types of moving objects such as walking people and moving vehicles with highly accurate positioning devices. Geographic applications are examples of systems that use obtained trajectories for analyzing the motion patterns of such moving objects. For example, motion patterns can be applied to navigation systems, traffic control systems, geographic information systems, location tracking systems in factories, and so on. Since the amount of trajectory data increases year by year, application system managers must regularly improve their systems to rapidly and efficiently analyze data [10].

One of the most significant analytical technologies is a clustering technique of trajectories based on similarities between moving objects. Clustering techniques can be applied in the following two ways:

A Group Detection System tries to aggregate the trajectories of moving objects that have similar motion patterns. For example, if the system finds a group of customers moving similarly in a shop, the shop's manager can apply that knowledge to optimize the arrangement and placement of products.

An Outlier Detection System tries to detect *outliers* that have obvious different motion patterns from other moving objects. In other words, the system detects moving objects that do not belong to any other class of moving objects. A security system, for example, can find a suspicious person in a station or an airport with this outlier detection technique.

To cluster moving objects, it is necessary to calculate similarity between two trajectories of moving objects based on their shapes and velocities. For calculating similarities, existing detection systems are applied methods to calculate similarity between signal data; Dynamic Time Warping (DTW) [5] and Longest Common Sub-Sequence (LCSS) [9] are the most popular methods. These methods can calculate similarity between two multidimensional trajectories as single-dimensional signal data; however, they focus on trajectory shapes and cannot compare velocities of moving objects. For instance, the methods cannot distinguish two moving objects that have the same shape but different velocities.

As a solution, we present two new and improved calculation methods to cluster trajectories, which can calculate similarity between multidimensional trajectories based on both shapes and velocities. These methods also make it possible to control the calculating ratio of shapes and velocities. One of our proposed methods involves an enhanced DTW to additionally calculate similarity of velocities, while the other features enhanced Euclidean Distance. In this paper,

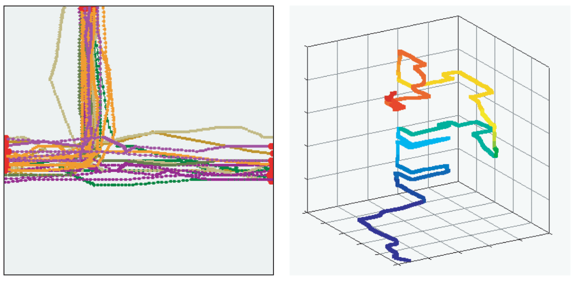


Figure 1. An example of trajectory data

we compare the results of our proposed methods with the results of existing methods using practical trajectory data. The results of experiments show the advantages calculating similarity between multidimensional trajectories based on shapes and velocities.

In Section 2, we mention practical multidimensional trajectories and related work on calculating similarity between signal data. Section 3 describes in detail our two proposed methods to calculate similarity. In addition we explain how to cluster trajectories based on our calculation methods.Experimental results are shown in Section 4, and we conclude our work in Section 5.

2 Similarity of Trajectories

In this section, we mention trajectory data and related work to calculate similarity between trajectories. Moreover, we explain our approach to calculate similarity based on both the shapes and velocities of moving objects.

2.1 Trajectory Data

Generally speaking, the trajectory of a moving object means a sequence of positions through which the objects pass. The trajectory is a continuous line, but positioning devices can track a moving object discretely. Thus, we define a trajectory as λ, that is, a sequence of discrete points $\boldsymbol{p_i}$ consisting of position data x_i, y_i and time data t. For example, a trajectory can be denoted as $\lambda = \langle p_1, p_2, \ldots, p_n \rangle$. In addition, we denote a set of trajectories as $\Lambda = \lambda_1, \ldots, \lambda_m$ and use notations $\lambda(t_i) = (x_i, y_i)$ and $\lambda(i) = t_i$ for the following definition[1].

Figure 1 illustrates an example of practical trajectories of rickshaws working in Nara that we tracked using a hand-held GPS receiver. Furthermore, Fig. 1(b) shows the projection of the trajectory onto t–x–y space, and we illustrate the projection of scaled trajectory onto the x–y plane in Fig. 1(a).

[1]In this paper, we assume that each position on a trajectory is obtained at the same interval.

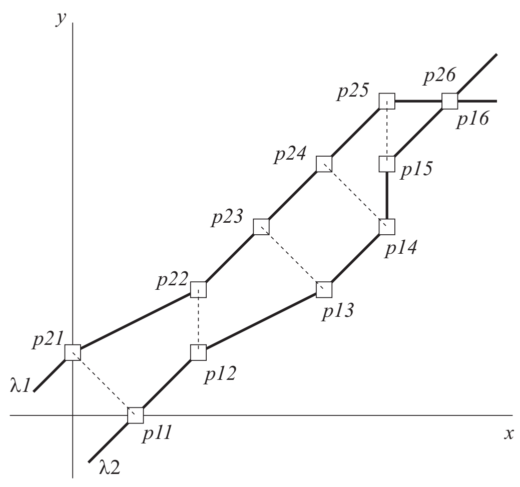


Figure 2. Euclidean Distance

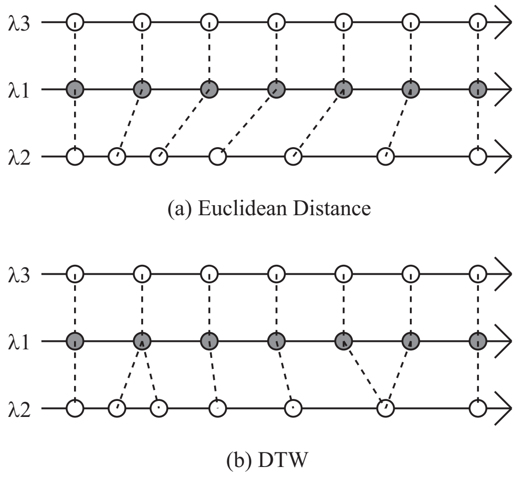


Figure 3. Difference between Euclidean Distance and DTW

2.2 Related Work

Because most previous works deal with trajectories as multidimensional time-series data, they calculate similarity between trajectories based on the similarity of single-dimensional time series data. We came across the following two popular methods to calculate similarity:

1. Similarity based on Euclidean Distance focuses on the similarity of velocities [4] [3].
2. Similarity based on Dynamic Time Warping (DTW)[5] [8] or Longest Common Subsequence (LCSS) [9] focuses on the similarity of shapes.

In Euclidean Distance, similarity between two trajectories is based on the sum of the squares of distances between

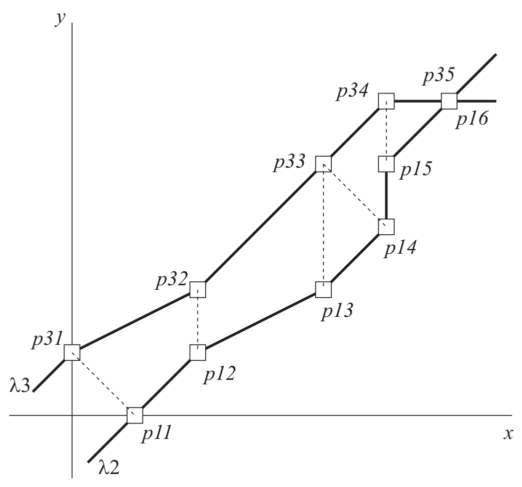


Figure 4. Dynamic Time Warping

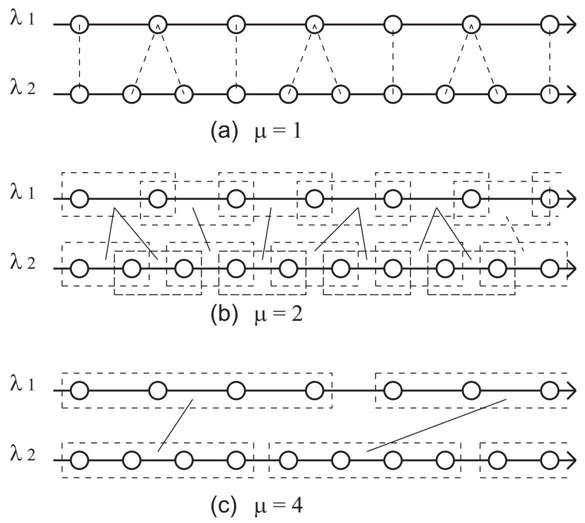


Figure 5. Our proposed similarity between trajectories

each point on each trajectory. Figure 2 shows an example of similarity, with it calculation explained in Appendix A. According to this method, we calculate a distance between points obtained at the same time; this is, each point is just once used to calculate distance. Thus, the space between neighbor points strongly affects the calculation of similarity. For example, trajectory λ_1 in Fig. 3(a) has the same shape as λ_2 and λ_3, but the spaces between points on λ_3 are of different lengths to the other trajectories. In this case, λ_1 is more similar to λ_2 than λ_3 in this calculation method. Because the length of space between points strongly affects this similarity, it is valid to cluster trajectories based on velocities. On the other hand, it is difficult to calculate similarity between trajectories of different length, and moreover, the shape has little effect on the result of similarity calculation.

In the calculation method based on DTW, similarity between two trajectories is base on the distance between the nearest points on each trajectory. In contrast to Euclidean Distance, the similarity of shapes strongly affects the results of calculating similarity between trajectories. We explain the detailed calculation method in Appendix B. As Fig. 4 shows, because the space between neighbor points is ignored in this method, the velocities of moving objects have no effect on the result of calculating similarity. LCSS also features the same characteristics with respect to similarity.

Through the above discussion, we conclude that previous calculation methods cannot reflect both velocity and shape in order to calculate similarity.

2.3 Our Proposed Calculation Methods

In general, the calculating method focuses either on velocity or shape because they are trade-off parameters in the calculation of similarity. This means a serious problem exists in methods that cannot control the ratio of velocity and shape in calculation. Therefore, we propose a new method to calculate similarity that enables us to explicitly indicate the ratio of velocity and shape in calculation. Before constructing our methods, we focus on the characteristics of existing calculation methods:

1. Basically, the distance between points on each trajectory is used to calculate similarity.

2. The method based on Euclidean Distance can be used to calculate partial similarity between parts of trajectories, which have the same μ points. These calculated results are the same as the distance between two sequences that includes μ points.

3. An increase in points included in the trajectory strongly affects the velocity in the calculation results. If the number of points is 1 ($\mu = 1$), the calculation results do not affect velocity.

Hence, we propose the following new methods to calculate similarity:

1. Our methods calculate similarity using the distance between μ points on trajectories instead of the distance between two points on trajectories. The distance between μ points introduces the effect of velocity to similarity, based on DTW.

2. We can show the ratio between velocity and shape with parameter μ with our method. For example, if $\mu =$

1, calculated similarity has no effect on velocity. On the other hand, if μ is equal to the length of the entire trajectory, the similarity has a minimal effect on shape.

We explain this concept in detail using an example illustrated in Fig. 5. Existing methods, such as DTW, calculate similarity using distance $D'(p_{1i}, p_{2j})$ between points on $\lambda_1 = \langle p_{11}, p_{12}, \ldots, p_{1n}\rangle$ and $\lambda_2 = \langle p_{21}, p_{22}, \ldots, p_{2m}\rangle$, shown in Fig. reffig:sim(a). Distance function D' is defined as equation 15 described in Appendix B. In our calculation method, however, similarity is calculated using a Euclidean distance between ordered μ points on trajectories as shown in Fig. 5(b).

When $\mu = 2$, two partial trajectories, $\lambda'_1 = \langle p_{11}, p_{12}\rangle$ and $\lambda'_2 = \langle p_{21}, p_{22}\rangle$ are generated from λ_1 and λ_2, which only contain both the first and second points of the original trajectories. Next, instead of distance function D' in the existing methods, we use Euclidean distance function $D_{euc}(\lambda'_1, \lambda'_2)$ for calculating similarity. Refer to equation 10 in the Appendix for the definition of D_{euc}. Similarly, for each pair of sub-sequences $\langle p_{1i}, \ldots, p_{1i+m}\rangle$, $\langle p_{2i}, \ldots, p_{2i+m}\rangle$, similarity is calculated in order. In this manner we can naturally introduce similarity of velocities into existing shape-based similarity.

On the other hand, when a large μ, such as $\mu = 100$, is given, Euclidean distance $D_{euc}(\lambda'_1, \lambda'_2)$ is larger than in the case of a small μ, such as $\mu = 2$ as shown in Fig. 5(c). In other words, an increase in the of value μ strongly affects velocity in the calculation of similarity. We can adapt length μ to control the ratio between velocity and shape.

In remainder of this paper we describe two calculation methods based on the above discussion.

3 Clustering Trajectories

In this section, we describe our proposed methods, Maximum Trajectory in nearest neighbors (MT-nn) and DTW enhanced for trajectories (DTW-t), to calculate similarity between trajectories for clustering in 3.1 and 3.2. Next, in Section 3.3, we explain a method to cluster trajectories using our calculation methods.

3.1 MT-nn: Maximum Trajectory in Nearest Neighbors

The basic approach of an MT-nn calculation method is to introduce Euclidean distance to the calculation process for the maximum distance between points on trajectories. According to this calculation method based on maximum distance, the distance between two crossing objects is less than the distance between objects moving together, even if they do not cross. This calculation method is used in the Dynamic Range Query [6] to find moving objects with an indicated object.

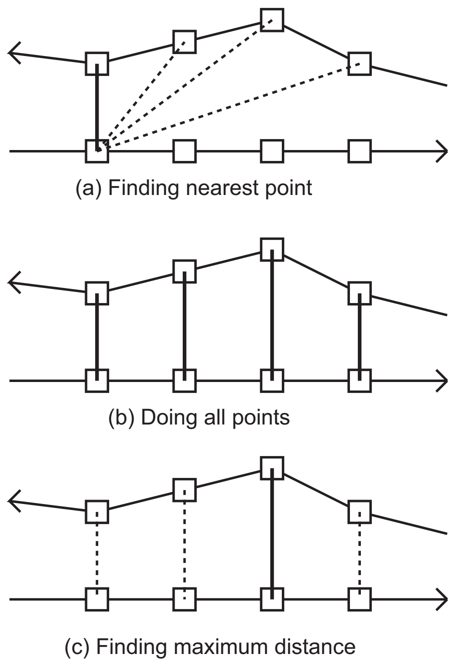


Figure 6. Similarity based on maximum distance

Here we explain how to calculate distance $D_{md}(\lambda_1, \lambda_2)$ between trajectories λ_1 and λ_2 with an example illustrated in Fig. 6. (We assume that each trajectory has n points. For example, in the case of Fig. 6, n is 4.)

1. Search the nearest point to each point p_{1i} on λ_1 in points $p_{21}, \ldots, p_{2n}$ on λ_2, as shown in Fig. 6(a).

2. Generate pairs $\{p_{11}, p_{2i}\}, \ldots, \{p_{1n}, p_{2j}\}$ to obtain the shortest didtance between the points. In the case of Fig. 6(a), we can find pairs $\{p_{11}, p_{21}\}, \{p_{12}, p_{22}\}, \{p_{13}, p_{23}\}, \{p_{14}, p_{24}\}$.

3. Find a pair whose distance is the maximum, so the distance of the pair becomes $D_{md}(\lambda_1, \lambda_2)$. In the previous example, $\{p_{13}, p_{23}\}$ is the maximum pair, and the distance between p_{13} and p_{23} is the result of calculating $D_{md}(\lambda_1, \lambda_2)$.

The following equation represents this process. (D' is a distance function given in equation 15.)

$$D_{md}(\lambda_1, \lambda_2) = \max_{v=1\rightarrow|\lambda_1|}\min_{w=1\rightarrow|\lambda_2|} D'(p_{1v}, p_{2w}) \quad (1)$$

Similarity $S_{md}(\lambda_1, \lambda_2)$ is given as a reciprocal function of $D_{md}(\lambda_1, \lambda_2)$ such that $S_{md}(\lambda_1, \lambda_1)$ equals zero.

The essential idea of the MT-nn calculation method is based on replacing function D' in D_{md} into D_{euc}. Before explaining this method, we must define several symbols: sub-trajectory $\lambda[i,j]$ = $\langle p_i, \ldots, p_{i+j-1}\rangle$ means a part of a trajectory $\lambda = \langle p_1, p_2, \ldots, p_n\rangle$ where $i + j \leq |\lambda|$ and $i < j$, and i, j are natural numbers. Obviously, we find that $|\lambda[i,j]| = j$. Using this definition, we define similarity $S_{mnn}(\lambda_1, \lambda_2)$ based on MT-nn according to the following equation:

$$\begin{aligned} D_{mnn}(\lambda_1, \lambda_2) = \\ \max_{v=1\rightarrow|\lambda_1|-\mu} \\ \min_{w=1\rightarrow|\lambda_2|-\mu} \\ D_{euc}(\lambda_1[v,\mu], \lambda_2[w,\mu]) \quad (2) \\ S_{mnn}(\lambda_1, \lambda_2) = 1/D_{mnn}(\lambda_1, \lambda_2). \quad (3) \end{aligned}$$

Furthermore, we explain the calculation processes.

1. Extract sub-trajectory $\lambda_1[i,\mu]$ that has μ points in trajectory s_1.
2. Find sub-trajectory $\lambda_2[j,\mu]$ such that $D_{euc}(s_1[i,\mu], s_2[j,\mu])$ is the minimum in all other sub-trajectories. The distances D_{euc} are calculated and recorded for each i, from $i = 1$ to $i = |\lambda_1| - \mu$.
3. Finally, retrieve the maximum distance from the distances recorded in the previous steps.

The reciprocal numbers of the maximum distances is similarity $S_{mnn}(\lambda_1, \lambda_2)$ between λ_1, λ_2. This definition of similarity controls the ratio between the effect of velocity and shape with parameter μ, as described in 2.3. For example, if $\mu = 1$, similarity has no effect on velocity; in contrast, if μ is almost equal to the length of the trajectory, similarity has a strong effect on velocity.

3.2 DTW Enhanced for Trajectory

In simple DTW, each item $\gamma(i,j)$ in a matrix (described in Appendix B) is given as distance function $D'(p_{1i}, p_{2j})$ (defined in equation 15). Based on our previous discussion, we replace function $D'(p_{1i}, p_{2j})$ with $D_{euc}(\lambda_1[i,\mu], \lambda_2[j,\mu])$ for calculating the similarity between multidimensional trajectories. We define similarity $S_{pdtw}(\lambda_1, \lambda_2)$ with the following equation:

$$\begin{aligned} \gamma(i,j) &= D_{euc}(\lambda_1[i,\mu], \lambda_2[j,\mu]) \\ &+ \min \begin{cases} \gamma(i, j-1) \\ \gamma(i-1, j) \\ \gamma(i-1, j-1) \end{cases} \quad (4) \\ D_{pdtw}(\lambda_1, \lambda_2) &= \gamma(|\lambda_1| - \mu, |\lambda_2| - \mu) \quad (5) \\ S_{pdtw}(\lambda_1, \lambda_2) &= \frac{\max(|\lambda_1|, |\lambda_2|)}{D_{pdtw}(\lambda_1, \lambda_2)}. \quad (6) \end{aligned}$$

With this calculation method, an increase in value μ also has a string effect on velocity to similarity $S_{pdtw}(\lambda_1, \lambda_2)$.

3.3 Clustering

For clustering trajectories, we adapt a hierarchical clustering method [1] [2] because it is suitable for clustering the data into unknown classes. A hierarchical clustering method, especially the group averaging method, is also appropriate for clustering data with a non-Euclidean distance function that does not satisfy a condition such that $S_{mnn}(\lambda_1, \lambda_3) \leq S_{mnn}(\lambda_1, \lambda_2) + S_{mnn}(\lambda_2, \lambda_3)$. Since the detailed clustering processes are described elsewhere [1] [2], we only summarize them in this paper. Such clustering methods aggregate data that have similar characteristics, though they prefer to separate data having different characteristics rather than aggregate similar data. Therefore, we consider the group averaging method valid for detecting outliers, as mentioned in Section 1.

This method clusters data into classes, such that distance $d_{ave}(\Lambda_1, \Lambda_2)$ between any two classes Λ_1, Λ_2 satisfies the following equation:

$$\begin{aligned} &d_{ave}(\Lambda_1, \Lambda_2) \\ &= \frac{1}{|\Lambda_1||\Lambda_2|} \sum_{\lambda_1 \in \Lambda_1} \sum_{\lambda_2 \in \Lambda_2} D_{mnn}(\lambda_1, \lambda_2). \quad (7) \end{aligned}$$

Since this equation is adapted into our proposed distance $D_{mnn}(\lambda_1, \lambda_2)$, we can also adapt distance $D_{pdtw}(\lambda_1, \lambda_2)$ into the averaging clustering method. The results of clustering based on this method are provided in Section 4.

To compare the results of clustering, we also apply our calculation method to the farthest-neighbor algorithm, which is also a hierarchical clustering method. In the clustering function of the farthest-neighbor, distance d_{max} is given as the following equation.

$$d_{max}(\Lambda_1, \Lambda_2) = \max_{\lambda_1 \in \Lambda_1, \lambda_2 \in \Lambda_2} D_{mnn}(\lambda_1, \lambda_2) \quad (8)$$

In contrast to group averaging, this method prefers to aggregate similar data rather than separate different data. In other words, data included in a class have less uniformity than data aggregated by group averaging; however, the number of classes generated by the farthest-neighbor is higher than classes by group averaging. Thus, the farthest-neighbor method is valid for group-detection application systems.

4 Evaluation

In this section, we cite experiments on clustering trajectories obtained by positioning devices.

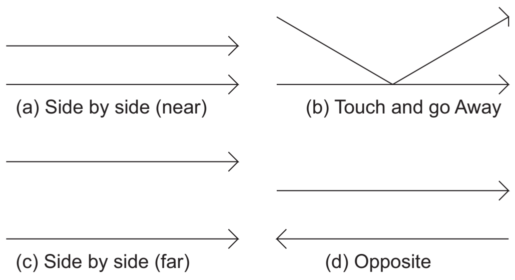


Figure 7. Pairs of trajectories used for evaluation

4.1 Experimental Setting

We performed two experiments to evaluate our proposed method. The aim of the first experiment is to evaluate value μ in calculating similarity; in the second experiment, we examine the performance of our proposed method using practical trajectory data.

In the first experiment, we used two sets of trajectories with different velocity values: 1 m/s and 2 m/s. Additionally, we set up the four variations of value μ shown in Fig. 7. To compare clustering performance, we checked the results for each of the four values such that $\mu = 1, 4, 8,$ and 16. Both MT-nn and DTW-t were applied to cluster trajectories in this experiment. In the second experiment we compared classifications using several calculation methods, including our proposed methods.

For evaluation, we employed the trajectory data from 39 rickshaws working in Nara during January 2003. The average rickshaw velocity was about 10 km/h, with each vehicle driving about 20 km per day. The rickshaws' velocity was generally low when carrying passengers. Before the experiment, we normalized the trajectory data whereby the interval for each point was one second. Furthermore, the maximum error of the positioning device was within 15 m, and the average of errors was around 3–4 m.

We use two types of trajectories to compare the results of clustering: on intersections and on straight roads.

4.2 Calculating Similarity

Figures 8 and 9 illustrate the results of calculating similarity. The y-axis of each graph is the distance between trajectories, the decrease of the y value represents an increase of similarity between trajectories. The value of the x-axis is the value of μ. Descriptions such as a–1 and b–1 in the figures mean a pair of two moving objects shown in Fig. 7; for example, a–2 indicates that the shape of the data is (a), i.e., two moving objects traveling side by side, and '2' means that the velocity of one moving object is twice the velocity of the other.Figures 8 and 9 show the results for the clustering simple data set, where all objects move at the same velocity and the shapes of all trajectories resemble each other.

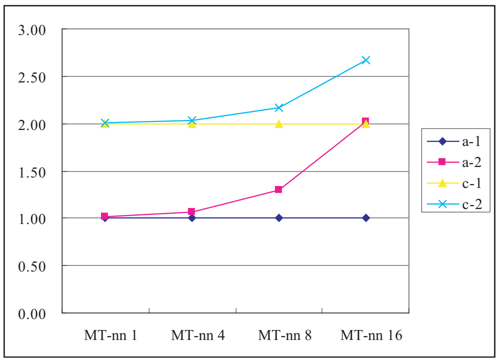


Figure 8. Distance calculated by MT-nn

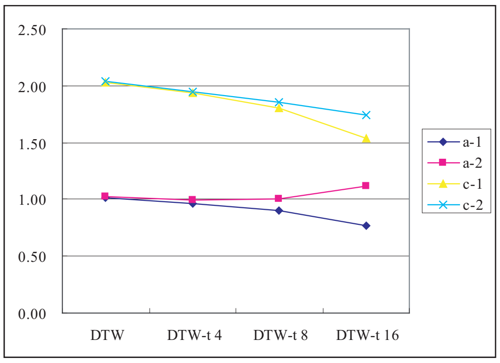


Figure 9. Distance calculated by DTW-t

The results of clustering indicate that an increase in μ strongly affects the velocities of the clustering. In other words, when two objects move at the same velocity, in our methods they belong to the same class even if their trajectories have different shapes. Note that our proposed MT-nn method strongly affects velocity with an increase in μ, but DTW-t has little effect on the velocity.

Figures 10 and 11 visualize the clustering results using trajectories where objects move in opposite directions or their shapes are quite different. These results also show that an increase in μ strongly influences clustering in the MT-nn method. On the other hand, the DTW-t method prefers to aggregate trajectories that have the same velocity than aggregate trajectories whose shapes are similar where the value of μ increases.

Thus, we conclude that the MT-nn method is suitable for clustering moving objects with similar velocities, whereas the DTW-t method is valid for clustering moving objects with similar shapes. Moreover, it is possible to use μ to control the ratio of velocity and shape in clustering trajec-

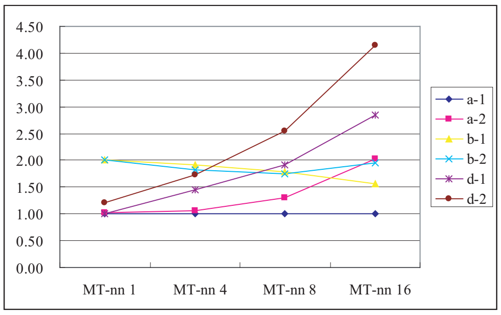


Figure 10. Distance calculated by MT-nn (2)

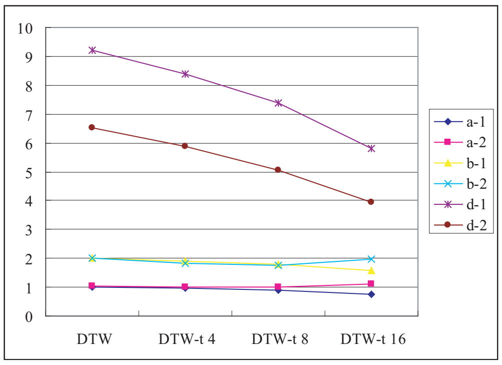
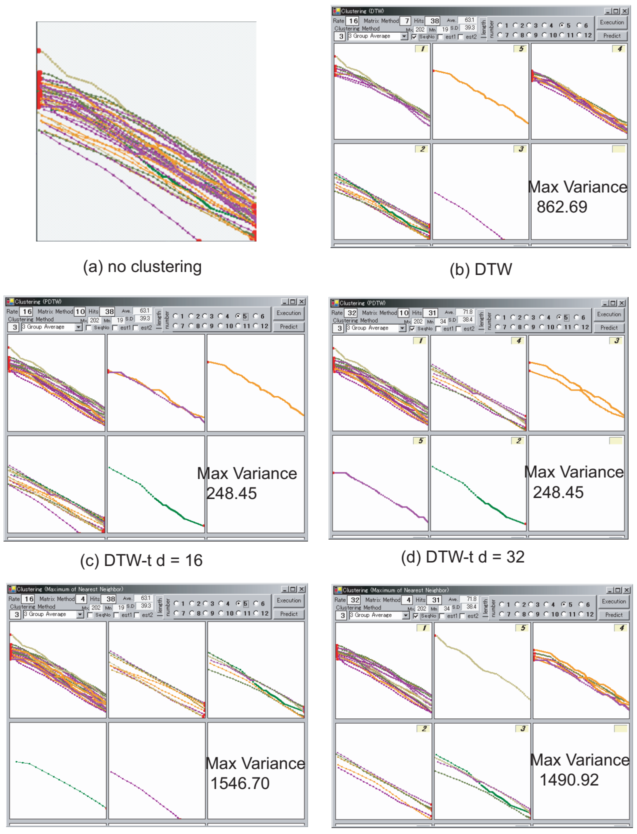


Figure 11. Distance calculated by DTW-t (2)



(a) no clustering (b) DTW (c) DTW-t d = 16 (d) DTW-t d = 32 (e) MT-nn d = 16 (f) MT-nn d = 32

Figure 12. Results of the clustering trajectories by grouping average

tories. The results satisfy the discussion in 2.3.

4.3 Clustering Results

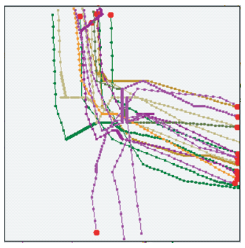
Figure 12 illustrates the results of clustering 38 trajectories of rickshaws, which move on a straight street, with the group-averaging and similarity-calculation methods. Figure 12(a) shows the original trajectories, where every rickshaw moves from right to left or from left to right. It is clear that although the directions of motion may differ, the shapes of trajectories are quite similar. For clustering such data, we generally prefer to use the similarity of velocities to the similarity of shapes.

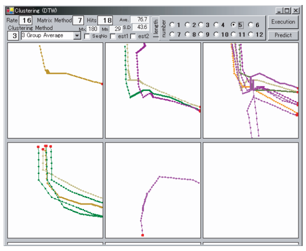
We provide the clustering results as shown in Figs. 12 (b) to (f) with the original DTW, DTW-t, and MT-nn. The important point here is not the similarity between the shapes of trajectories in a class, but the similarity of the velocities. To evaluate the similarity of velocities, we have to calculate the variance of objects' velocities in each class. Comparing the variance of objects, we can validate the performance of the calculation method to cluster trajectories based on the similarity of velocities.

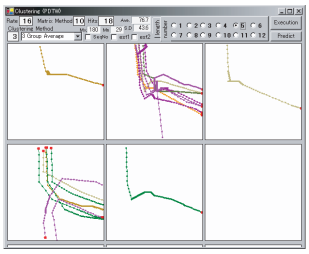
The variance of clustering results with the original DTW shown in Fig. 12(b) is 1,040.03, which is quite high. In contrast, any variance with our proposed DTW-t is 248.45, which is far lower than the original DTW. We cannot find any significant difference between the variances of (c) and (d). The second-highest variance, 226.35, is in (c), but the second-highest variance in (d) is the lower 196.35. The result means that DTW-t more strongly influences velocity than does the original DTW. Although DTW-t is good for clustering such data, the MT-nn method is not because both the shapes and velocities of trajectories are quite similar to each other. For clustering such data, then, DTW-t has an advantage over the other methods. The above results are coincidental with the discussion described in Section 4.2.

On the other hand, the MT-nn method has an advantage over cluster trajectories that have different shapes and velocities, as Fig. 13 indicates. In calculating the DTW-t method, if a shape of a sub-trajectory is similar to a part of another trajectory, their similarity is calculated as quite near, even if the shapes of all trajectories are not similar. These experimental results clearly exhibit the different characteristics of each calculation method.

Moreover, we performed an experiment to compare clustering methods, group averaging, and farthest-neighbor, as


(a) no clustering


(b) DTW


(b) DTW-t d = 16

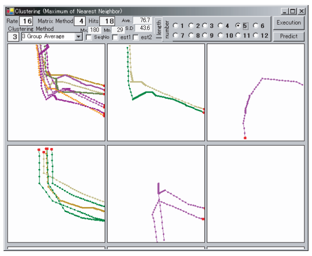
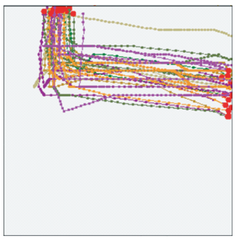
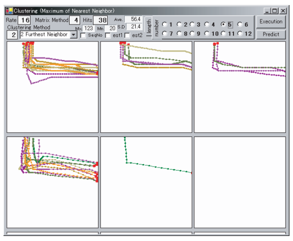
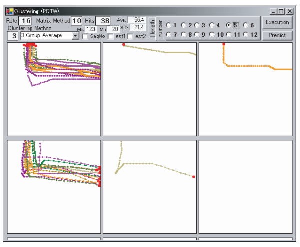

(b) MT-nn d = 16

Figure 13. Results of the clustering trajectories at an intersection


(a) no clustering


(b) MT-nn d = 16 (MAX)


(c) DTW-t d = 16 (Ave)

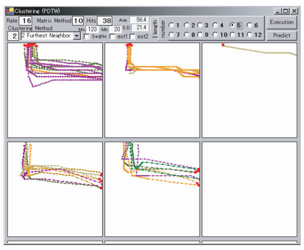

(d) DTW-t d = 16 (MAX)

Figure 14. Results of the clustering by different methods

described in Section 3. Figure 14 illustrates the clustering results. In group averaging, because the clustering system prefers diverging trajectories whose similarity is low, outliers belong to an independent class from the others, as shown in Fig. 14(c): the farthest-near method prefers to increase the number of trajectories in a class as much as possible. Consequently, these results indicate that this method is not suitable for outlier detection.

These experiments confirmed that our proposed methods can cluster trajectories with controllable ratios of shape and velocity.

5 Conclusion

We proposed two new methods to calculate similarity between multidimensional trajectories for clustering, based on the characteristics of both velocity and shape. This paper also reported on experiments conducted to evaluate our calculation methods using the practical trajectory data of working rickshaws. From the experimental results, we conclude that our proposed methods are indeed suitable for clustering multidimensional trajectories, and have advantages over other methods.

Though in this paper we only dealt with extended DTW and Euclidean Distance, our approach can be easily applied to other calculation methods such as LCSS. Future work will include experiments with other extensions.

References

[1] D. Buzan, S. Sclaroff, and G. Kollios. Extraction and clustering of motion trajectories in video. In *ICPR 2004 Proceedings*, pages 521–524, 2004.

[2] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2000.

[3] E. Keogh, K. Chakrabarti, S. Mehrotra, and M. Pazzan. Locally adaptive dimensionality reduction for indexing large time series databases. In *SIGMOD2001 Conference Proceedings*, pages 151–162, 2001.

[4] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, 3(3):263–286, 2001.

[5] E. J. Keogh. Exact indexing of dynamic time warping. In *VLDB2002 Conference Proceedings*, pages 406–417, 2002.

[6] I. Lazaridis, K. Porkaew, and S. Mehrotra. Dynamic queries over mobile objects. In *EDBT 2002 Conference Proceedings*, pages 269–286, 2002.

[7] S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung. Similarity search for multidimensional data sequences. In *ICDE 2000 Proceedings*, pages 599–608, 2000.

[8] S. Park, W. W. Chu, J. Yoon, and C. Hsu. Efficient searches for similar subsequences of different lengths in sequence databases. In *ICDE Proceedings*, pages 23–32, 2000.

[9] M. Vlachos, C. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *ICDE 2002 Proceedings*, pages 673–684, 2003.

[10] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang. Moving objects databases: Issues and solutions. In *Statistical and Scientific Database Management (SSDM'98) Conference Proceedings*, pages 111–122, 1998.

[11] Y. Yanagisawa, J. Akahani, and T. Satoh. Shape-based similarity query for trajectory of mobile objects. In *MDM2003 Conference Proceedings*, pages 63–77, 2003.

A Euclidean Distance

Euclidean Distance gives a distance between two sequences $s_1 = \langle q_{11}, q_{12}, \ldots, q_{1n} \rangle$ and $s_2 = \langle q_{21}, q_{22}, \ldots, q_{2n} \rangle$ having the same items. When trajectories have n items, Euclidean distance $D_{euc}(s_1, s_2)$ is given by the following equation:

$$D_{euc}(s_1, s_2) = \sqrt{\frac{\sum_{k=1}^{n} (q_{1k} - q_{2k})^2}{n}}. \tag{9}$$

Similarity $S_{euc}(s_1, s_2)$ between two sequences is given as $1/D_{euc}(s_1, s_2)$. Many signal processing systems use similarity based on Euclidean Distance for sub-sequence matching, which is one of the most significant techniques in signal processing. In general, sub-sequence matching means a process to find an answer sub-sequence from sub-sequences $s_1 \ldots, s_m$, such that the answer sub-sequence has the shortest distance to given sequence s_q. Equation 9 gives a distance between two single-dimensional sequences; similarly, the distance between multidimensional sequences can be defined. When $|\lambda_1| = |\lambda_2| = n$ and s_1, s_2 in Eq. 9 are replaced with λ_1, λ_2, the distance can be given as the following equation (a detailed definition is provided in [7] [11]):

$$D_{euc}(\lambda_1, \lambda_2) = \sqrt{\frac{\sum_{k=1}^{n} (x_{1k} - x_{2k})^2 + (y_{1k} - y_{2k})^2}{n}}. \tag{10}$$

Similarity $S_{euc}(\lambda_1, \lambda_2)$ of multidimensional trajectories can be given as $1/D_{euc}(\lambda_1, \lambda_2)$, an equation that shows velocity characteristics affect the similarity of trajectories more strongly than do shape characteristics.

In the case of the trajectories shown in Fig. 2, $D_{euc}(\lambda_1, \lambda_2)$ can be calculated as follows:

$$\begin{aligned} D_{euc}(\lambda_1, \lambda_2) &= \sqrt{\frac{2+1+2+2+1+0}{6}} \\ &= \sqrt{\frac{8}{6}} = 1.2. \end{aligned} \tag{11}$$

Using this equation, the similarity is given as $S_{euc}(\lambda_1, \lambda_2) = 1/1.2 = 0.83$.

Table 1. Example of a matrix $\gamma(i, j)$

$\gamma(i,j)$	p_{31}	p_{32}	p_{33}	p_{34}	p_{35}
p_{11}	1.4	3.7	8.7	15.1	22.1
p_{12}	3.4	2.4	6.0	11.0	16.7
p_{13}	7.5	4.4	4.4	7.6	11.2
p_{14}	12.9	7.6	5.8	6.4	8.7
p_{15}	18.8	11.2	6.8	6.8	7.8
p_{16}	26.0	16.2	9.1	7.8	6.8

B Dynamic Time Warping

Dynamic Time Warping (DTW) is one of the most popular ways to calculate similarity between trajectories with different numbers of points. DTW features several variations for adaptation to target data, and in this paper, we mention a simple DTW [8].

To calculate similarity between two time-series data $s_1 = \langle q_{11}, q_{12}, \ldots, q_{1n} \rangle$, $s_2 = \langle q_{21}, q_{22}, \ldots, q_{2m} \rangle$, the calculation system generates a matrix whose item is represented as $\gamma(i, j)$. The system decides each item according to the following equation:

$$\begin{aligned} &\gamma(i, j) = \\ &D(q_{1i}, q_{2j}) + \min \begin{cases} \gamma(i, j-1) \\ \gamma(i-1, j) \\ \gamma(i-1, j-1) \end{cases} \\ &\gamma(0,0) = \gamma(0,i) = \gamma(j,0) = 0. \end{aligned} \tag{12}$$

In this equation, $D(q, q')$ is a function that gives a distance between q, q', for instance, that we can define as $D(q, q') = |q - q'|$. Similarly, when the length of each s_1, s_2 is $|s_1|, |s_2|$, distance $D_{dtw}(s_1, s_2)$ can be defined as

$$D_{dtw}(s_1, s_2) = \gamma(|s_1|, |s_2|). \tag{13}$$

Similarity S_{dtw} of time-series data is given by the next equation:

$$S_{dtw} = \frac{\max(|s_1|, |s_2|)}{D_{dtw}(s_1, s_2)}. \tag{14}$$

It is easy to extend these calculation processes for multidimensional trajectories. For extension, function $D'(q_{1i}, q_{2j})$ in Eq. 12 is replaced by function $D'(p_{1i}, p_{2j})$, which gives the distance between two trajectories $\lambda_1 = \langle p_{11}, p_{12}, \ldots, p_{1n} \rangle$ and $\lambda_2 = \langle p_{21}, p_{22}, \ldots, p_{2m} \rangle$. Function $D'(p_{1i}, p_{2j})$ is given as follows:

$$D'(p_{1i}, p_{2j}) = \sqrt{(x_{1i} - x_{2j})^2 + (y_{1i} - y_{2j})^2}. \tag{15}$$

This function D' introduces the distance function D_{dtw} in DTW.

$$\gamma(i,j) = \\ D'(p_{1i}, p_{2j}) + \min \begin{cases} \gamma(i, j-1) \\ \gamma(i-1, j) \\ \gamma(i-1, j-1) \end{cases} \tag{16}$$

$$D_{dtw}(\lambda_1, \lambda_2) = \gamma(|\lambda_1|, |\lambda_2|). \tag{17}$$

Each element in a matrix is calculated recursively from $\gamma(1,1)$ to $\gamma(|\lambda_1|, |\lambda_2|)$, and D_{dtw} can be calculated from $\gamma(|\lambda_1|, |\lambda_2|)$. We can also obtain similarity S_{dtw} from D_{dtw}. Table 1 shows matrix $\gamma(i,j)$ generated from trajectories λ_1 and λ_3 as shown in Fig. 4. In this example, we obtain the results $D_{dtw}(\lambda_1, \lambda_3) = \gamma(6,5) = 6.8$ and $S_{dtw}(\lambda_1, \lambda_3) = 6/6.8 = 0.88$.