ICIC Express Letters
Volume **10**, Number **2**, February **2016**

ICIC International ©2016 ISSN 1881-803X
pp. 309–315

A VISUALIZATION TOOL FOR ROTATION-INVARIANT BOUNDARY IMAGE MATCHING

Seongwoo Moon[1], Sanghun Lee[1], Bum-Soo Kim[2] and Yang-Sae Moon[1,*]

[1]Department of Computer Science
Kangwon National University
1 Kangwondaehak-gil, Chuncheon-si, Gangwon-do 24341, Korea
{ swmoon; sanghun }@kangwon.ac.kr; *Corresponding author: ysmoon@kangwon.ac.kr

[2]Department of Knowledge Service Engineering
Korea Advanced Institute of Science and Technology
291, Daehak-ro, Yuseong-gu, Daejeon 34141, Korea
bskim@kangwon.ac.kr

Received October 2015; accepted December 2015

Abstract. *In this paper, we present a visualization tool that efficiently supports rotation-invariant boundary image matching. Supporting the rotation invariance in boundary image matching is very important for the more accurate and more intuitive matching result. We use the client-server model to design and implement the rotation-invariant matching. First, the client converts a query image to a time-series and sends it to the server. Second, the server performs the rotation-invariant matching by evaluating the query on the database with indexes. Third, the client visualizes the result received from the server. Experimental results show that our system supports the rotation-invariant property efficiently and provides the intuitive analysis through line and polar charts.*
Keywords: Rotation invariance, Boundary image matching, Time-series data, Rotation-invariant image matching

1. **Introduction.** Recently, there have been a lot of research efforts on exploiting large time-series databases for many applications such as document retrieval, image matching, and biosequence matching [1, 2, 3, 7]. Also, there have been many efforts on visualization of the mining result for time-series, image, and text data [5, 6, 11]. Visualization is a technique to communicate meaningful data or information by using images, charts, diagrams, or animations [6]. Through the visualization, we can improve efficiency of recognition, understanding, analysis, and process of the data [12].

Boundary image matching [4, 10] identifies the similar images by exploiting the time-series extracted from boundary images. By converting boundary images into time-series and exploiting fast time-series matching techniques, the boundary image matching can efficiently support a huge volume of image databases. In this paper, we use the centroid contour distance (CCD) [8, 10] to extract a time-series from a boundary image. The CCD method maps a boundary image to a time-series of length n by evenly dividing 2π into n angles (i.e., $\Delta\theta = 2\pi/n$) and computing the distance to each boundary point from the centroid. Figure 1 shows an example of converting an image into a time-series.

In this paper, we design and implement a visualization tool for rotation-invariant boundary image matching [9]. The rotation-invariant matching finds similar boundary images correctly even though they have some rotations. Recently, efficient rotation-invariant algorithms have been proposed in [9, 10]. To our best knowledge, however, there is no visualization for rotation-invariant matching, so it is difficult to understand the rotation-invariant matching result intuitively. To solve this problem, we develop an efficient tool of visualizing the rotation-invariant matching result for the more intuitive and easier understanding and analysis. The proposed tool visualizes the matching result as not only

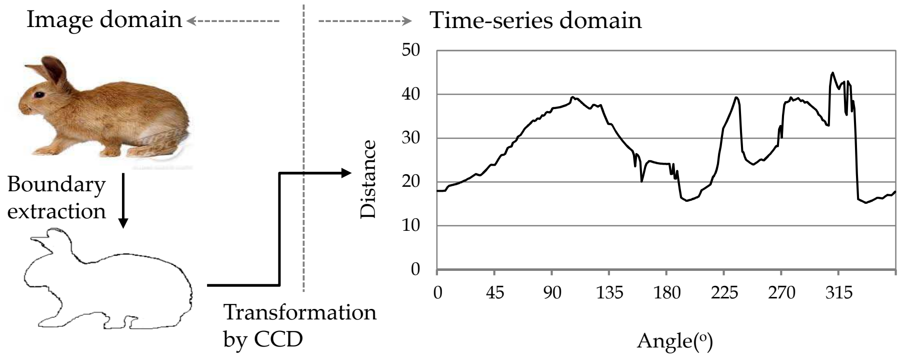


Figure 1. An example of converting an image to a time-series by CCD

images but also line and polar charts for fast comparison and easy validation. In particular, using the polar charts we can intuitively and interactively compare the query image with the similar data images of the matching result.

The rest of this paper is organized as follows. Section 2 explains the related work. Section 3 presents the concept of the rotation-variant boundary image matching system with its working framework. Section 4 explains the implementation result with its representative screenshots. We finally summarize and conclude the paper in Section 5.

2. **Related Work.** Time-series matching is the problem of finding the data time-series similar to the given query time-series [1]. In order to find similar data time-series, the matching method uses a similarity model based on the Euclidean distance [6, 7] or the dynamic time warping (DTW) distance [5]. In this paper, we use the Euclidean distance since it has been most widely used and easy to visualize the distance between image boundaries.

Image matching [4, 10, 13], also known as content-based image retrieval (CBIR), identifies data images similar to the given query image. In image matching, the representative features are colors, textures, and shapes. Among these features, we focus on image boundaries as the matching features. In this paper, we exploit the CCD method [4, 9, 10] to extract image boundaries as shown in Figure 1. Regarding the rotation-invariant matching, [9, 10] propose novel solutions, but these solutions focus on the matching performance rather than the visualization, which is the major contribution of this paper.

In recent years, there have been a few efforts on visualization of the time-series matching result. First, Lee et al. [5] visualize a two-dimensional matrix of computing the DTW distance between query and data time-series, so we can visually understand the actual process of computing the DTW distance. Second, Weber et al. [11] visualize time-series data as various spiral shapes by varying spiral thickness and color degrees. These two approaches, however, focus on original time-series only, that is, they do not consider image time-series at all. Third, as prior work of this paper, Moon et al. [6] present a visualization tool for basic boundary image matching. The tool visualizes the result of range and k-nearest neighbor queries in boundary image matching with line and polar charts. However, the tool supports the basic matching only and does not consider the rotation invariance, which is the major technical contribution of this paper.

3. **The Concept of Rotation-Invariant Boundary Image Matching.** The rotation-invariant boundary image matching [9] is the problem of finding similar boundary images correctly even though they are rotated any angles. That is, even though an image is rotated, we treat the rotated image as the original one by computing the rotation-invariant distance. For query and data time-series $Q = (q_0, \ldots, q_{n-1})$ and $S = (s_0, \ldots, s_{n-1})$ of length n, which are converted from boundary images, we compute their rotation-invariant

distance, $RID(Q,S)$, by shifting the query time-series one by one and computing the Euclidean distance between the shifted query time-series and the data time-series. More precisely, the rotation-invariant distance, $RID(Q,S)$, is computed as Equation (1).

$$RID(Q,S) = \min_{j=0}^{n-1} D\left(Q^j, S\right) = \min_{j=0}^{n-1} \sqrt{\sum_{i=0}^{n-1} \left|q_{(i+j)\%n} - s_i\right|^2}. \tag{1}$$

In Equation (1), $D(\cdot)$ is the Euclidean distance function, % is the modular operator, and Q^j represents a j-rotated time-series of $(q_j, \ldots, q_{n-1}, q_0, \ldots, q_{j-1})$, which is obtained by shifting the time-series Q by j times. As shown in Equation (1), $RID(Q,S)$ considers all possible j rotations, and thus, by using $RID(Q,S)$, we can extend the basic boundary image matching to support the rotation invariance.

However, $RID(Q,S)$ of Equation (1) requires $\Theta(n)$ Euclidean distance computations, and these many computations incur a critical performance overhead in the matching process. To solve the problem, in this paper we use the triangular inequality-based solution [9]. Figure 2 shows how we use the triangular inequality in computing the rotation-invariant distance. As shown in the figure, if we know $D\left(Q^j, S\right)$ and $D\left(Q^j, Q^{j+1}\right)$, we can compute a lower bound of $D\left(Q^{j+1}, S\right)$ as $\left|D\left(Q^j, S\right) - D\left(Q^j, Q^{j+1}\right)\right|$ by the triangular inequality. If the lower bound exceeds the tolerance, the Euclidean distance $D\left(Q^{j+1}, S\right)$ also exceeds the tolerance, and accordingly, we can discard Q^{j+1} without computing $D\left(Q^{j+1}, S\right)$. Also, we can use the lower bound of $D\left(Q^{j+1}, S\right)$ for computing a lower bound of $D\left(Q^{j+2}, S\right)$. More precisely, if we denote a lower bound of $D\left(Q^{j+1}, S\right)$ as lb^{j+1}, we can compute a lower bound lb^{j+2} of $D\left(Q^{j+2}, S\right)$ as $\left|lb^{j+1} - D\left(Q^{j+1}, Q^{j+2}\right)\right|$. Similarly, we can also compute a lower bound lb^{j+3} of $D\left(Q^{j+3}, S\right)$ as $\left|lb^{j+2} - D\left(Q^{j+2}, Q^{j+3}\right)\right|$[1]. By repeating this procedure of computing lower bounds and avoiding unnecessary Euclidean computations, we can improve the rotation-invariant matching performance significantly [9].

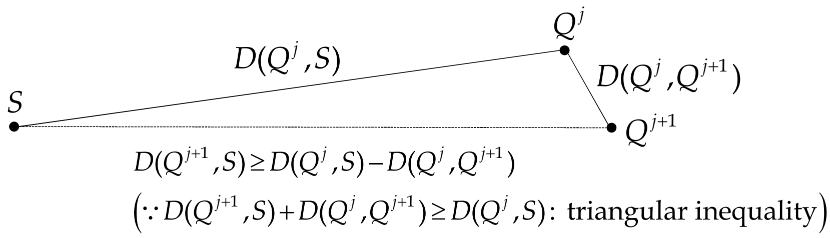


FIGURE 2. Triangular inequality-based lower bound for $RID(Q,S)$

Figure 3 shows an overall framework of the proposed rotation-invariant boundary image matching system, which works as a client-server model. First, the client sends a user-given time-series, which is converted from a query image by CCD, with the user-specified tolerance ϵ to the server. Second, the server performs the rotation-invariant boundary image matching to find similar data time-series from the database with indexes and returns the matching result to the client. Third, the client visualizes the matching result to different image formats and charts.

4. **Implementation Results of Rotation-Invariant Matching System.** We develop the matching and visualization system in the following hardware and software environment. First, we implement the client on Microsoft Visual Studio 2010 on Windows 7 operating system, and we use C# language with MSChart. Second, we implement the

[1]The distances of $D\left(Q^{j+1}, Q^{j+2}\right)$ and $D\left(Q^{j+2}, Q^{j+3}\right)$ are computed only once as self-rotation distances in prior to starting the matching process. Thus, we can ignore the overhead of computing those self-rotation distances for a large volume of image databases.

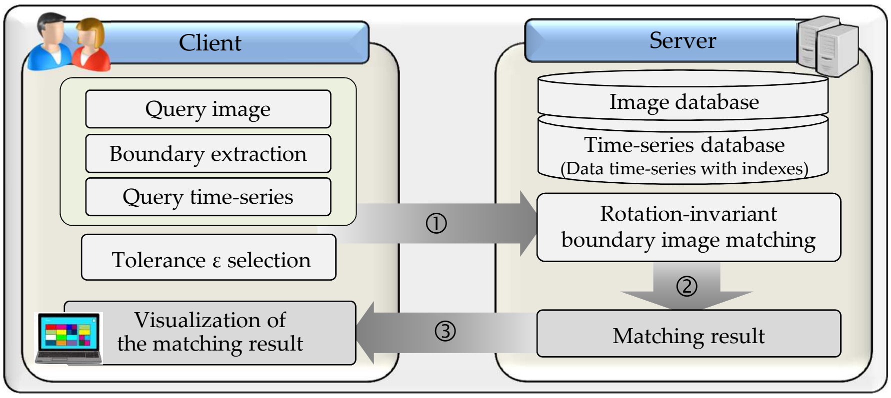


FIGURE 3. Overall framework of the rotation-invariant matching system

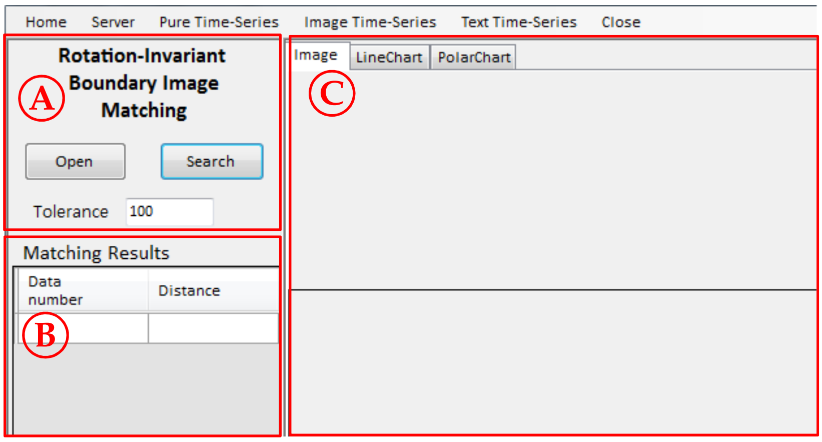


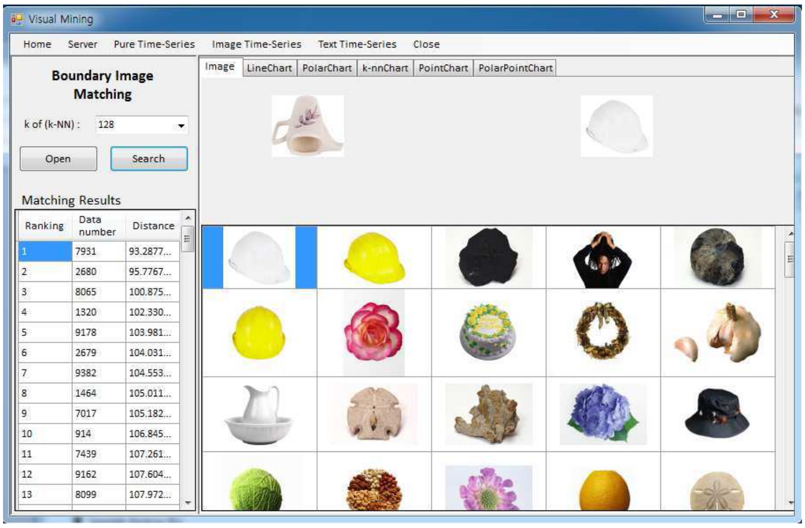
FIGURE 4. Initial screen of a visualization window in the client

server on a Linux machine with CentOS 5.9 and use C language for implementing the matching engine. We use a read data set consisting of 10,000 images, which are obtained from the Web [4, 6] and convert an image to a time-series of length 360 by CCD [9, 10].

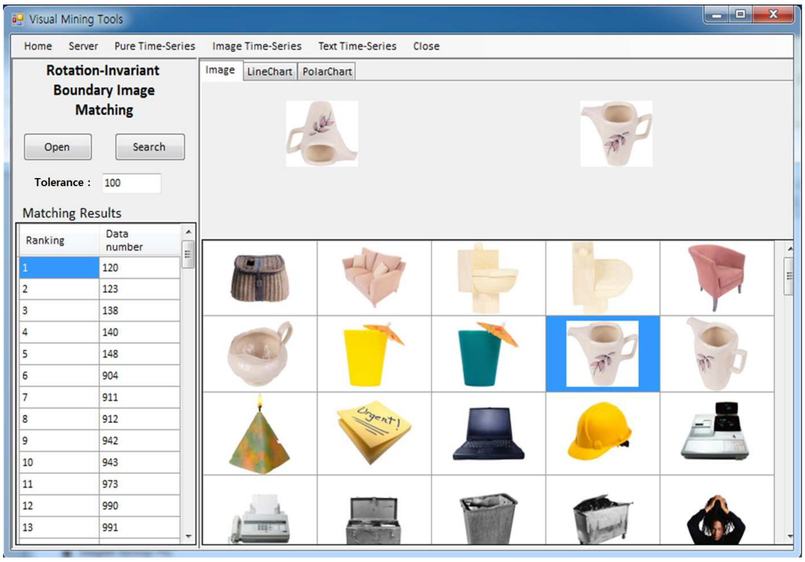
Figure 4 shows an initial screen of the proposed visualization tool for rotation-invariant boundary image matching. In Part Ⓐ of the figure, the "Open" button is for selecting a query image (time-series); a text box is for an input of the user-specified tolerance ϵ; and the "Search" button is for sending the query with ϵ to the server. Part Ⓑ is for showing the list of resulting images, which are received from the server, where each column represents the similarity rank between query and data images, the image number, and its actual rotation-invariant distance. Part Ⓒ is a visualization frame that displays the rotation-invariant similar images.

Figure 5 shows example screenshots of executing the rotation-invariant matching. Figure 5(a) represents an execution result of the basic boundary image matching, and Figure 5(b) shows the corresponding result of the proposed rotation-invariant matching. As shown in the figure, the visualization system displays the returned similar images in a form of grid views. In Figure 5(a), the basic matching system misses some similar images rotated from the original image since it does not support the rotation invariance; in contrast, in Figure 5(b), the rotation-invariant system finds many rotated images correctly since it supports the rotation invariance in the matching process.

Figure 6 shows a line chart which visualizes boundary time-series of query and data images of Figure 5. As shown in the figure, the line chart represents query and data images as their corresponding time-series in a single window, so we can intuitively understand the distance difference between those time-series. A list box in the left part of Figure 6



(a) Basic matching system



(b) Rotation-invariant matching system

Figure 5. Example screenshots of boundary image matching systems

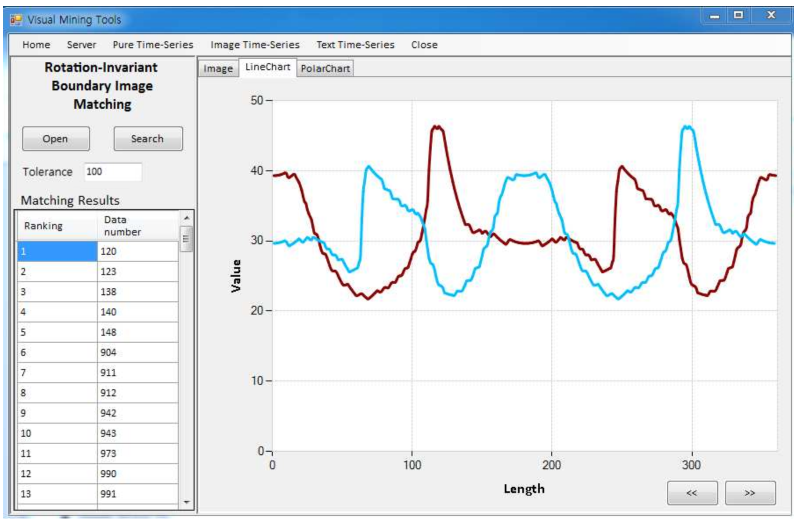
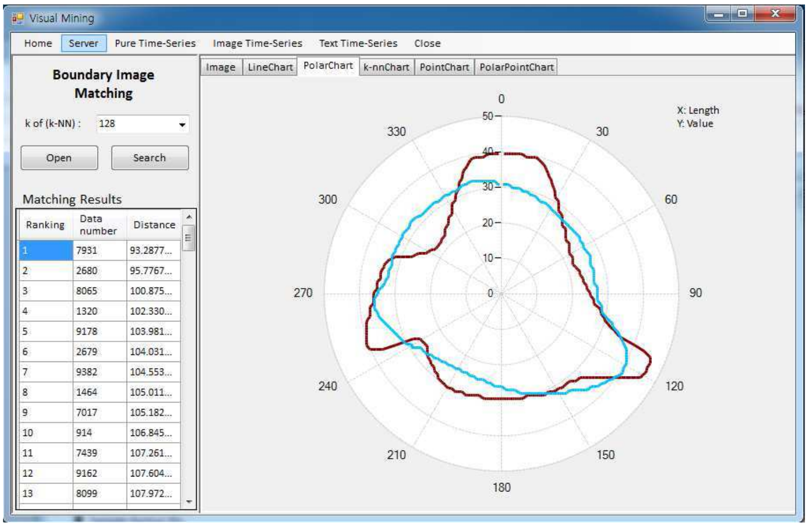
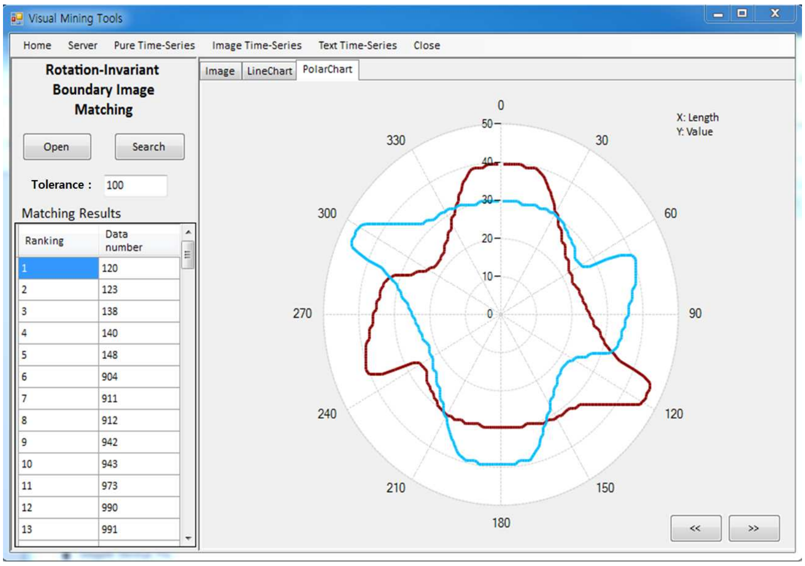


Figure 6. Line chart for visualizing query and data time-series in a window



(a) Basic matching system



(b) Rotation-invariant matching system

FIGURE 7. Polar charts for visualizing query and data time-series in a window

shows the similar data images with their identifiers and actual distances from the query image, so we can easily know the rank of each data image and its actual distance from the query image. Two direction buttons located in the bottom right part of the figure are used for moving forward or backward to next similar images, so we can use those buttons to compare the query time-series with multiple data time-series.

Figure 7 shows polar charts which visualize image boundaries of query and data images in a single window. Using a polar chart, we can visualize a time-series, which is converted from an image, to an image boundary. Image boundaries of Figure 7 correspond to the images of Figure 5. Figure 7(a) shows a basic matching system which does not support the rotation invariance while Figure 7(b) shows a rotation-invariant matching system that supports the rotation invariance in the matching process. In Figure 7(a), two boundary images are not similar with each other, but they are identified as similar ones since the system does not support the rotation-invariant property. In contrast, in Figure 7(b), the system identifies two image boundaries as similar ones since it supports the rotation-invariant property. Actually, those two images of Figure 7(b) are the same except that one is rotated from the other one. If the system does not support the rotation invariance, it identifies those two images as dissimilar ones, but if it supports the rotation invariance, it returns them as similar ones. In particular, using forward and backward buttons of the bottom right part, we can rotate a query image step by step and compare it with a data image interactively.

5. **Conclusions and Future Work.** In this paper, we design and implement a rotation-invariant boundary image matching system as a client-server model. In particular, the client supports a visualization tool which not only displays similar images in grid cells but also draws them with line and polar charts. Using the pairwise image comparison and the line/polar chart representation, we can understand the matching result intuitively and make a correct decision even for a variety of image rotations. We believe that the proposed rotation-invariant matching system will be very useful for various boundary image matching applications since it supports the rotation-invariant property efficiently, and at the same time it visualizes the matching result interactively. As the future work, we will consider the visualization of scaling-invariant and symmetric-invariant boundary image matching in the time-series domain.

Acknowledgment. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (NRF-2014R1A2A2A01002548).

REFERENCES

[1] R. Agrawal, C. Faloutsos and A. Swami, Efficient similarity search in sequence databases, *Proc. of the 4th Int'l Conf. on Foundations of Data Organization and Algorithms*, Chicago, Illinois, pp.69-84, 1993.

[2] K. P. Chan, A. W. C. Fu and C. T. Yu, Haar wavelets for efficient similarity search of time-series: With and without time warping, *IEEE Trans. Knowledge and Data Engineering*, vol.15, no.3, pp.686-705, 2003.

[3] J. Han, M. Kamber and J. Pei, *Data Mining: Concepts and Techniques*, 3rd Edition, Morgan Kaufmann, 2011.

[4] B.-S. Kim, Y.-S. Moon, M.-J. Choi and J. Kim, Interactive noise-controlled boundary image matching using the time-series moving average transform, *Multimedia Tools and Applications*, vol.72, no.3, pp.2543-2571, 2014.

[5] S.-J. Lee, J. Lee, H. Cho and W.-S. Han, A visualization tool for ranked subsequence matching in time-series databases, *Journal of KIISE: Databases*, vol.38, no.2, pp.92-103, 2011 (in Korean).

[6] S. Moon, S. Lee, B.-S. Kim and Y.-S. Moon, Visualization tool for boundary image matching based on time-series data, *Proc. of DASFAA Workshops*, Hanoi, Vietnam, pp.286-292, 2015.

[7] Y.-S. Moon, K.-Y. Whang and W.-S. Han, General match: A subsequence matching method in time-series databases based on generalized windows, *ACM SIGMOD International Conference on Management of Data*, Madison, Wisconsin, pp.382-393, 2002.

[8] Y.-S. Moon, B.-S. Kim, M. S. Kim and K.-Y. Whang, Scaling-invariant boundary image matching using time-series matching techniques, *Data and Knowledge Engineering*, vol.69, no.10, pp.1022-1042, 2010.

[9] Y.-S. Moon and W.-K. Loh, Triangular inequality-based rotation-invariant boundary image matching for smart devices, *Multimedia Systems*, vol.21, no,1, pp.15-28, 2015.

[10] M. Vlachos, Z. Vagena, P. S. Yu and V. Athitsos, Rotation invariant indexing of shapes and line drawings, *Proc. of ACM Conf. on Information and Knowledge Management*, Bremen, Germany, pp.131-138, 2005.

[11] M. Weber, M. Alexa and W. Muller, Visualizing time-series on spirals, *Proc. of Int'l Conf. on IEEE Symp. on Information Visualization*, San Diego, California, pp.7-13, 2001.

[12] *Wikipedia*, http://en.wikipedia.org/wiki/Visualization.

[13] X. Yang, X. Bai, S. Koknar-Tezel and L. J. Latecki, Densifying distance spaces for shape and image retrieval, *Journal of Mathematical Imaging and Vision*, vol.46, no.1, pp.12-28, 2013.