ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2007, Vol. 36, No. 2

APPLICATION OF CLUSTERING ALGORITHMS IN EYE GAZE VISUALIZATIONS

Oleg Špakov

Department of Computer Science, University of Tampere
FIN-33014 Tampere, Finland

Darius Miniotas

Department of Electronic Systems, Vilnius Gediminas Technical University
LT-03227 Vilnius, Lithuania

Abstract. Eye tracking devices generate enormous amount of data, which requires a well-balanced approach to selective visualization of the data. This approach involves employing some data clustering algorithm. Most of the traditional algorithms, however, are too slow as well as inadequately deterministic to be applied to eye gaze data. This paper describes our software implementation of two modifications of the clustering algorithm suitable for visualization of eye gaze data. Such a visualization greatly facilitates data analysis by grouping the individual samples into more meaningful units referred to as gaze fixations.

Keywords: eye tracking, data clusters, clustering algorithms, information visualization, pie charts.

1. Introduction

The amount of data produced by eye tracking devices is huge. Software for data recording may receive 30 to 4000 samples of gaze position per second [6]. Most of the time, the data is stored in files for further off-line analysis. Plotting every sample of gaze position as a separate point makes the display overwhelmed and unclear, and the duration of recording may be prolonged. This complicates analysis of the data.

Often, when analyzing multiple recordings over the same stimulus image, the number of areas of high observation intensity is relatively small, whereas most other areas receive much less attention. Therefore, showing the data points of little relevance only hinders the analysis.

To avoid such a situation, some techniques have been developed to reduce the amount of the data displayed. Commonly, these techniques employ an algorithm for clustering data.

There is a variety of algorithms for clustering data points in 2D (see, e.g., [3] for a review). However, most of them are too slow for gaze data analysis, where we have hundreds of thousands of data points stored in a file. In addition, the requirement of knowing in advance the exact number of clusters cannot usually be met. Therefore, only a few algorithms are suitable for clustering eye gaze data.

In this paper, we review algorithms used for clustering eye gaze data. Then, we describe our implementation of two modifications of the algorithm giving the best performance. Presentation of the graphical user interface with the options available for visualization of gaze fixation clusters concludes the paper.

2. Gaze Fixation Clustering Algorithms

Algorithms relevant to clustering eye gaze data usually employ a two-level scheme that first clusters individual samples into fixations, and then clusters those fixations into "mega-fixations".

Results produced by different algorithms may vary significantly. All the algorithms provide some variables that the user can adjust to get a meaningful display. Often, selecting appropriate values for these variables is critical for successful data analysis. For instance, when the scope of fixations is too broad, or too narrow, interpretation of the data tends to be biased [2].

Scinto and Barnette [4] were the first to propose a simple algorithm based on the minimum number of fixations and the maximum distance between each pair of the fixations in a cluster. Ramakrishna et al. [1] extended the algorithm by including several methods to calculate the center of a cluster as: (1) the simple mean of fixations $\{x, y\}$, (2) the weighted (by

duration) mean of fixations $\{x, y\}$, and (3) the center of the convex polygon encompassing fixations of the cluster.

Fixation clustering algorithms giving the best results can be divided into two groups: (1) the distance-threshold algorithm, and (2) the mean-shift algorithm. Details on these two algorithms are provided in the following subsections.

2.1. Distance-Threshold Algorithm

This simple data-driven algorithm clusters fixations according to the predefined maximum distance. Two points are considered to belong to the same cluster if they are spaced closer to each other than this predefined distance.

The mechanics of the algorithm can be briefly summarized as follows.

- Take a fixation and assign it to a new cluster, if it is not yet assigned to any other cluster;
- Calculate the distances to the fixations not yet analyzed, and add to the same cluster only those that are closer than the predefined distance;
- If the fixation added already belongs to other groups, connect these groups to the current group;
- Repeat the procedure until all the fixations are analyzed.

This algorithm has the disadvantage of creating too large clusters when many fixations are scattered across the entire image, so that almost all the intermediate clusters have at least one common fixation.

2.2. Mean-Shift Algorithm

This algorithm does not have the disadvantage mentioned above. There are two modifications of the algorithm. The first of them clusters all the fixations closely spaced to each other regardless of the sequence of their occurrence, whereas the other modification also takes into account the time sequence. This is a robust version of the distance-based clustering including a preprocessing stage. The algorithm involves two steps:

- Move points closer to each other until they can be easily separated by clusters;
- Apply some distance-threshold algorithm (such as the one described in the previous section).

The first step (the mean-shift procedure) is crucial, and makes the entire process robust. It moves point X_i to a new location $S(X_i)$. The result of the shifting is the weighted mean of the nearby points based on the kernel function K.

$$S(X) = \frac{\sum_j K(X - X_j) X_j}{\sum_j K(X - X_j)}. \qquad (1)$$

The kernel function is usually a multivariate Gaussian with zero mean and covariance σ. Robustness can be achieved by limiting the support of the kernel (setting it to zero) for distances greater than 2σ. The covariance describes the spatial extent of the weighted mean computed by S. The user can control the size of clusters through controlling this value. Usually σ is below the distance between two image (stimulus) features.

Then, the kernel function can be transformed to support clustering of points in 2D using the following equation.

$$K(X_i, Y_i) = \exp\left(-\frac{X_i^2 + Y_i^2}{\sigma_s^2}\right). \qquad (2)$$

The mean shifting procedure is applied iteratively to move all the points toward a location of higher density until they are relatively close to the weighted centers of the clusters. The cluster centers are the points of moving convergence. The number of iterations is typically 5 to 10 that allow achieving convergence to within 0.1% change across the iterations.

Since the input to the algorithm is fixations F, better clustering may be achieved through applying the weights W for each fixation. The fixation's duration can be used as the weight. The weights may reduce the number of iterations as much as twice.

$$S_{FIX}(F) = \frac{\sum_j W_j K(F - F_j) F_j}{\sum_j W_j K(F - F_j)}. \qquad (3)$$

With all the points collected at the nodes, the distance-threshold clustering method is quite safe. The same covariance σ can be used as the distance threshold for the final clustering. Finally, clusters having small total fixation duration can be discarded, as they typically are outliers.

3. Visualizing Gaze Data Clusters

The visualization shows a gaze path separated and grouped into several clusters (Figure 1). This display is most effective when working with several recordings simultaneously, as the visualization techniques used allow easy evaluation of the contribution of each recording.

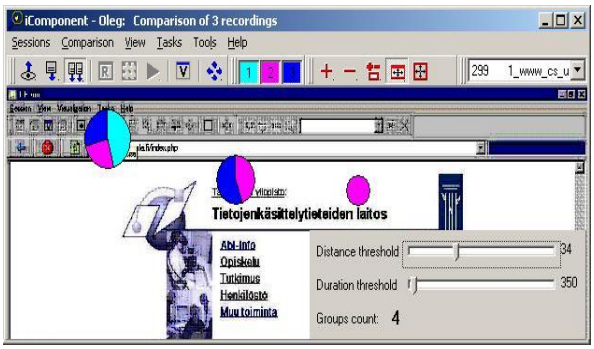

Figure 1. Visualization *Clustered* with appropriate panel

The clustering algorithm requires a value that defines the maximum inter-fixation distance in a cluster.

This value is known as *Distance threshold* and can be adjusted on the *Clustering panel*. This panel has also a slider to adjust the minimal duration of a visible cluster.

While creating the visualization, the clustering algorithm forms groups (clusters) of fixations. The distance between any two fixations in a group does not exceed the *Distance threshold* value defined in the *Clustering panel*. To get a more robust clustering, the *"Mean shift"* procedure may be applied to the fixation points before clustering the data. This procedure requires σ for the Gaussian distribution, and a convergence threshold to stop the iterations when this value is reached. The values mentioned can be adjusted using the visualization options dialog window (see Figure 6).

Each cluster is represented by the coordinates of its center, ordinal number, and the list of fixations included. In the display, a cluster may appear as a single randomly colored circle at the estimated position (weighted or simple average of the positions of the fixations in the list). Alternatively, a cluster may be shown as a group of fixations represented by small circles of the same color and size. It the latter case, the user may select the option for drawing a bounding rectangle or ellipse (Figure 2). Bounding by ellipse is more preferable as it appears properly allocated in space to minimize the area occupied. The radius of a cluster depends on the total duration of the fixations in the cluster.

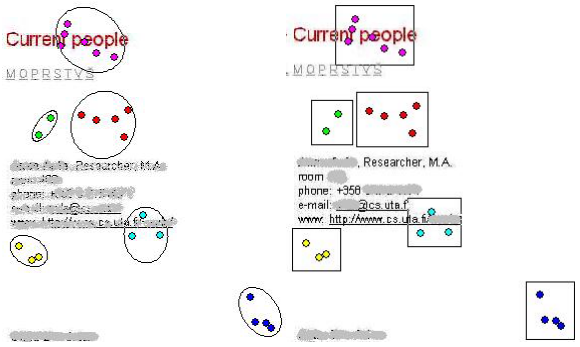

Figure 2. Clusters of fixations bounded by ellipses, or rectangles

Clusters consisting of one fixation may be hidden, as well as the clusters with the total duration of fixations less than the *Duration threshold* in the *Clustering panel*.

If several recordings are loaded into the analysis window, clusters are represented by pie charts showing the contribution of each recording in terms of duration (Figure 3). Colors of the pie's parts are the same as those assigned to the recordings. The left-most cluster includes fixations from all three recordings loaded. The middle cluster has fixations from two recordings. The right-most cluster has fixations from one recording only.

When the mouse cursor moves over a pie, a hint pops up to show the numerical data of the total cluster duration and the contribution of each recording (Figure 4). The contribution is presented by both absolute (milliseconds) and relative (percentage from total duration) values.

Figure 3. Clusters appearing as pie charts when fixations from several recordings are contained

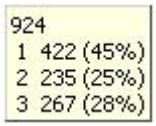

Figure 4. Hint over solid-circle cluster

The user may click on a pie to get the full statistics on it. This is shown in the *Clustering statistics* window (Figure 5).

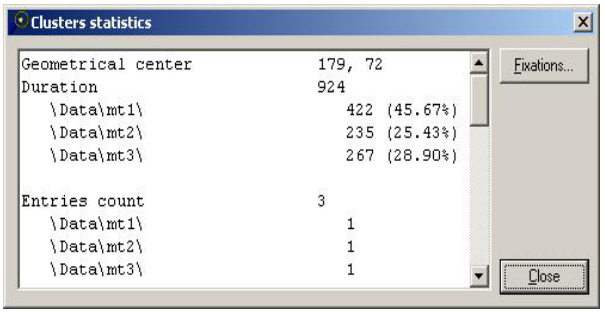

Figure 5. Statistics on cluster

The window displays the following data: (1) *geometrical center* of the cluster; (2) *duration* – the total duration of all fixations in the cluster; (3) *entries count* – the number of the gaze entries in the cluster (in other words, the count of uninterrupted segments of the gaze path); (4) *fixation count* – the count of fixations in the cluster; (5) *average, shortest, and longest fixation durations*; and (6) *standard deviation* of fixation durations.

4. GUI Options for Visualization

The group *Algorithm options* in the dialog window of the visualization *Clustered* (Figure 6) contains values for the clustering algorithm. If the flag *Apply mean shift* is checked, the mean-shift algorithm is applied before detecting clusters. The sliders *Sigma* and *Convergence threshold* allow adjusting the corresponding parameters of this algorithm.

The second group of controls (*Visibility*) allows defining the visibility of clusters. If the flag *Unite fixations* is checked, clusters appear as solid circles (Figure 3), where their radius represents the total duration of the cluster. The scheme of mapping the

duration to the radius uses a scaling value provided by the slider *Circle radius scale* on the right of this flag. The cluster's center can be either simply the mass center of the fixation composing the cluster. Alternatively, if the flag *Weighted center* is checked, the cluster's center is calculated using the duration of each fixation as a weight:

$$X_c = \frac{\sum_N X_i \cdot D_i}{\sum_N D_i}, \quad (4)$$

Here, the variables denote: X_c – the <*x*> coordinate of the cluster, N – the number of fixations in the cluster, X_i – the <*x*> of the i^{th} cluster's fixation, D_i – the fixation's duration. The expression for calculating the <*y*> coordinate is similar.

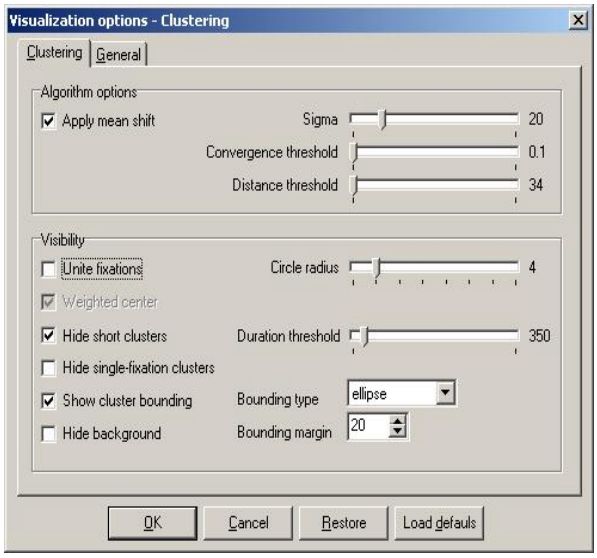

Figure 6. Options of visualization *Clustered*

If the *Unite fixations* flag is unchecked, each fixation is displayed as a circle of certain radius (now the *Circle radius* slider is visible, and can be adjusted if needed). The fixations of the same cluster have the same color.

If the *Hide short clusters* flag is checked, only the clusters having a total duration bigger than the value of the *Duration threshold* slider are visible. If the *Hide single-fixation clusters* flag is checked, the clusters consisting of a single fixation are hidden.

With the *Show cluster bounding* flag checked, each solid-circle cluster is displayed with a bound, either a rectangle or ellipse (the items in the *Bounding type* drop-down list). The bound is drawn around a cluster (Figure 2) with the offset defined in the *Bounding margin* control.

Finally, the *Hide background* flag allows controlling visibility of the background picture. Sometimes the users may wish to set it invisible. This is especially true when they need to visualize clusters as groups of related fixations, and the image is high-contrast and very detailed (i.e., it is hard to detect fixations visually).

5. Conclusions

The clustering algorithm proposed in this paper is suitable for visualization of eye gaze data. The algorithm was implemented in software that has the following main features:

- visualization of system and experimental events;
- highly adjustable visualization objects (colors, sizes, fonts, etc.);
- convenient zooming and panning of the display;
- hints containing short information over the objects or events;
- visualization of each subject's contribution to the observation using pie-diagrams;
- various visualizations of fixation clusters.

The software was successfully tested in an experiment designed to compare the observation patterns of various text fonts. The software allowed convenient visualization of the eye tracker's data superimposed upon the observed letters [5].

References

[1] **S.P. Ramakrishna, B.D. Barnette, D. Birkmire, R. Karsh.** Cluster: A program for the identification of eye-fixation-cluster characteristics. *Behavior Research Methods, Instruments & Computers* 25, 1993, 9-15.

[2] **D.D. Salvucci, J.H. Goldberg.** Identifying fixations and saccades in eye-tracking protocols. *Proc. ETRA* 2000, *ACM Press*, 2000, 71-78.

[3] **A. Santella, D. DeCarlo.** Robust clustering of eye movement recordings for quantification of visual interest. *Proc. ETRA* 2004, *ACM Press*, 2004, 27-34.

[4] **L. Scinto, B.D. Barnette.** An algorithm for determining clusters, pairs or singletons in eye-movement scan-path records. *Behavior Research Methods, Instruments & Computers* 18, 1986, 41-44.

[5] **O. Špakov, T. Evreinova, G. Evreinov.** Pseudo-graphic typeface: design and evaluation. *Proc. Nordic Symposium on Multimodal Communication,* 2003, 183-196.

[6] **D.S. Wooding, M.D. Mugglestone, K. Purdy, A.G. Gale.** Eye movements of large populations: I. Implementation and performance of an autonomous public eye tracker. *Behavior Research Methods, Instruments & Computers* 34, 2002, 509-517.

Received March 2007.