A Viable Implementation of a Comparison Algorithm for Regions of Interest

John P. Heminghous
Computer Science
Clemson University
jheming@acm.org

Abstract

A fully automatic tool for the quantitative comparison of various subject's eye-gaze data would be very valuable in many eye tracking studies. In this paper we discuss our implementation of such a tool based on previous clustering and comparison techniques. The trial experiment we performed verifies that our implementation is correct and effective. The system developed will be used in further studies to compare expert/novice data.

1 Introduction

Eye tracking has shown itself to be a valuable method of providing data describing the visual attention and cognitive state of a user. An observer focuses on a certain position in a scene so he or she can see that section in high detail in order to process specific visual information most efficiently [Duchowski 2003]. Consequently, eye-fixation data supplies important information about what regions in a scene are most significant. Methods needed to cluster collected fixations into larger regions-of-interest (ROIs) have been studied and improved to an adequate level [Santella and DeCarlo 2004]. What is further needed is a robust method of comparison between multiple viewers' ROIs.

The aim of this paper is twofold: to describe the implementation of an algorithm, based on a combination previous work [Privitera and Stark 2000; Santella and DeCarlo 2004], that objectively compares multiple subjects' ROIs and to prove the validity of the algorithm through experimentation.

A robust comparison method can be used to more accurately measure results in various experimental situations. One example of previous eye tracking work that could use such analysis is aircraft inspection. The presented results from Sandsivan et al. [2005] measured only speed and accuracy to obtain comparison data between subjects. The use of the algorithm to be presented could provide more concrete results. Another example is the work done by Law et al. [2004] on surgical training. Again results were gauged by speed and accuracy. While such measures may be sufficient for any given task, direct ROI comparison will yield more detailed and specific results. Furthermore, the work could be used in most eye tracking research where multiple subjects' performance is being measured.

In section 2 relevant work concerning clustering and ROI comparison is discussed in detail. Section 3 describes the methodology used in designing an experiment to verify the comparison algorithm's correctness. The operation of the algorithm and its use in analyzing the test data is covered in depth in section 4. Section 5 presents the quantitative results obtained by applying the algorithm to the test data. A discussion of the results comprises section 6. Plans for future work are conveyed in section 7.

2 Background

An algorithm for comparing eye gaze data has already been developed [Privitera and Stark 2000]. The problem with this algorithm is that it was specifically designed to compare artificially generated point-of-regard (POR) data and it uses k-means, an ineffective (for our purposes) clustering algorithm. This section will describe first the method developed by Privitera and Stark [2000] and second the clustering method developed by Santella and DeCarlo [2004] used to replace the k-means clustering technique.

In the previous ROI comparison work, all viewable area of a scene was split up into regions using the k-means clustering method. Each region was labeled with a single character. Data collected from human viewers and generated by various algorithms was applied to the scene and fell into one of the predefined regions to form clusters. The data was then used to construct a string from the characters bound to each region representing were a viewer had looked and in what order (Figure 1). Two metrics were then used for comparison: a spatial index S_p and a sequential index S_s. The coefficients of each of these measures can be represented in a table called a Y-matrix, but because of size restrictions the data from the Y-matrices were compacted into Parsing diagrams. The Parsing diagrams contained four items (scanpaths): Repetitive (*R*), the same viewer looking at the same scene at different times; Local (*L*), different viewers looking at the same scene; Idiosyncratic (*I*), the same viewer looking at different scenes; Global (*G*) different viewers looking at different scenes. Using Parsing diagrams to analyze various subjects' results provides information on whether the subjects looked at the same areas (S_p) and whether they looked at those areas in similar order (S_s).

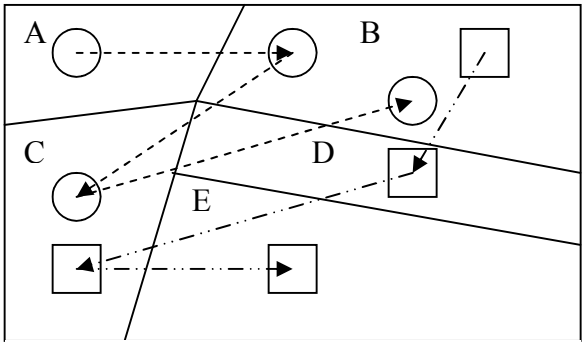

Figure 1: k-mean clustering; two viewers - viewer 1: circles (ABCB), viewer 2: boxes (BDCE)

The clustering algorithm developed by Santella and DeCarlo [2004] was chosen over k-means for this project because of its increased robustness. It was developed on the key principles of consistency, no foreknowledge, and robustness in the sense that isolated outliers do not affect clusters. The clustering algorithm uses a mean shift method to arrive at its results. The mean shift procedure repeatedly moves each point *x* to a new location s*(x)* until convergence is achieved. Then all points in the proximity of each other can be considered to be one cluster.

More specific information detailing the operation of these mentioned methods will be covered later in the operation section which describes their combination and implementation.

3 Methodology

The goal of the experiment was to verify the correctness of the comparison algorithm. It was hypothesized that the repetitive and local measures of subjects instructed to view an intuitive scene should be exceptionally high.

3.1 Apparatus

The experiment was performed with a Tobii 1750 eye tracker. The Tobii 1750 is a 17 inch flat screen monitor with an incorporated eye tracker. The resolution of the monitor is 1280 x 1024. The eye tracker is capable of binocular tracking at a 50Hz sampling rate within 0.5 degrees of accuracy. An AMD 64 PC running Windows XP and software provided with the Tobii interprets the eye tracking data and exports it via TCP/IP. The display, data collection, and analysis programs were run on a PC with a AMD Opteron processor running Fedora Core Linux. The Linux box used TCP/IP to collect the data from the Windows box. The display and analysis applications were developed in OpenGL using C. The data collection application was developed in C++.

Figure 2: Sample (aggregated) numerical stimulus

3.2 Experimental Design

Subjects consisted of six college students (all male). Ages of the participants ranged from 21 to 42 years old. The number of subjects was selected relative to past studies and availability. Subjects were screened based on their ability to calibrate well with the eye tracker.

The visual stimuli consisted of three blank black screens with randomly placed numbers 1 through 4 (Figure 2) and a more complex computer generated (CG) image produced by a ray tracer (Figure 3). In the first three numerical images each number was flashed onto the screen one at a time. The individual numbers were displayed for 500ms to allow for an initial orientation time and one long fixation because fixation durations generally range from 150-500ms [Duchowski 2003]. The first stimulus (all four numbers) was repeated in order to obtain repetitive measures. The last stimulus was displayed for 5 seconds because of its vastly increased complexity.

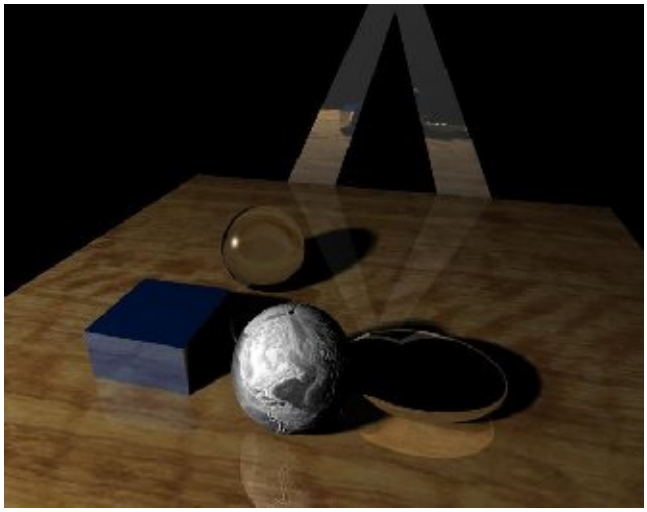

Figure 3: Ray tracing stimulus

The last image was included because it was expected that data collected on the image would have far lower local measures between subjects than the other data. Because no task was provided in viewing the CG image each subject should inspect the image differently and therefore produce widely varying eye-gaze (fixation) data.

3.3 Procedures

The subjects were placed in directly in front of the Tobii at approximately 60cm distance. Calibration was performed by displaying nine blue circles evenly spaced throughout the display. The circles were displayed independently and shrank down from a diameter of 30 pixels to a diameter of 2 pixels. The eye tracker collected 22 calibration samples at each circle location. The calibration accuracy was stored after each collection. The test subjects were instructed to look at each circle as it appeared before the calibration began. Average precision variance was computed to ensure each calibration was within acceptable limits.

Before each trial began the test subjects were instructed to fixate on each number as it appeared and to freely inspect the last image. They were informed of how much time they would have to look at each number and the final CG image. For each run all the POR data (x, y, and time stamp) was collected. The data was

exported along with headers indicating the current viewer and image to a log file to be analyzed at a later time.

After the test another calibration was performed and the average precision variance was again collected. The variances from the two calibrations were compared in order to factor out any slippage of the eye tracker.

4 Operation

The implement algorithm consisted of two parts: clustering and comparing. First the clustering logic will be covered and then the comparing logic.

Before the clustering was performed velocity based saccade detection was used to filter out any data points considered as part of a saccade. Velocity was defined as difference in position over difference in time.

$$v = \frac{\sqrt{(x_{t+1} - x_t)^2 + (y_{t+1} - y_t)^2}}{dt}$$

Any velocity above 130°/sec [Duchowski 2003] identified data as part of a saccade, and resulted in its exclusion from the usable data set.

4.1 Clustering

The first step in the algorithm defined by Santella and DeCarlo [2004] is called the mean shift procedure. The process starts with a set of n points:

$$\{x_j \mid j \in 1..n\},$$

and repeatedly relocates each point x_j to a new locality $s(x_j)$ which is the weighted mean of nearby data points.

$$s(x) = \frac{\sum_j k(x - x_j)x_j}{\sum_j k(x - x_j)}$$

The symbol k is the kernel function that defines the effects of data points on each other.

$$k_{spatial}([x_i, y_i]) = \exp\left(-\frac{x_i^2 + y_i^2}{\sigma_s^2}\right)$$

The parameter σ_s defines the spatial extent of the kernel. Specifically it guarantees that no clusters exist closer in locality than σ_s. As suggested kernel support was limited to $2\sigma_s$ in order to eliminate the effects of distant outliers.

The implementation keeps two lists: one containing the original data and the other the mean shifted data. The mean shifted list originally contains a copy of the original data but is repeatedly operated on (s(x) is calculated) until convergence. Convergence is detected by the condition that no data point moves more than ε pixels in a single mean shift step. For this experiment ε was set to five pixels for all trials. Varying ε a small amount such as ten pixels would produce negligible effects. A data point is then grouped into a cluster with all points less than σ_s pixels away from itself. The mean shifted list contains a reference back to the original data list that it was copied from, and therefore, the original data can be classified into clusters. Clusters with less than five members are considered outliers and discarded from further use. All viewer data for each image is clustered together for comparing.

4.2 Comparing

The comparison methods developed by Privitera and Stark [2000] used string based comparison to exhibit differences in scan paths. In this implementation each cluster is labeled with a character. The cluster ordering is defined by the first viewers gaze data and is kept the same for all other viewers. Strings for each viewer (for each image) are then constructed by concatenating the character, from each cluster visited by the viewers gaze data, to the current end of the string.

After the strings are constructed S_p and S_s are then computed. Given the strings *a* and *b*, S_p is computed by dividing the number of characters that appear in both strings by the number of characters in *a*. Consider the strings *a* = ABACED and *b* = ABACD. Then S_p, the location similarity between *a* and *b* is (5/6) = 0.83. Again given the strings *a* and *b*, S_s is computed by subtracting the Levenshtein distance between *a* and *b* divided by the number of characters in *a*. The Levenshtein distance between two strings is based on the cost of three operations: insertion, deletion, and substitution used two transform the second string (*b*) into the first (*a*). Using the strings supplied above S_s, the sequential similarity between *a* and *b* is (1-1/6) = 0.83.

While Privitera and Stark [2000] used four similarity metrics: repetitive, local, idiosyncratic, and global, for viewer comparison purposes, the idiosyncratic (same viewer, different images) and global (different viewer, different images) were not relevant, hence, not included in our implementation.

4.3 Visual Data Representation

The analysis program displays a visualization of the data for each (aggregate) image (Figure 4). Subject's original eye-gaze data is color coded to match the legend in the top right corner of the display. Bright red points represent all subject's mean shifted data. A bright blue ellipse surrounds each cluster, and a bright green character towards the center of the cluster is its label.

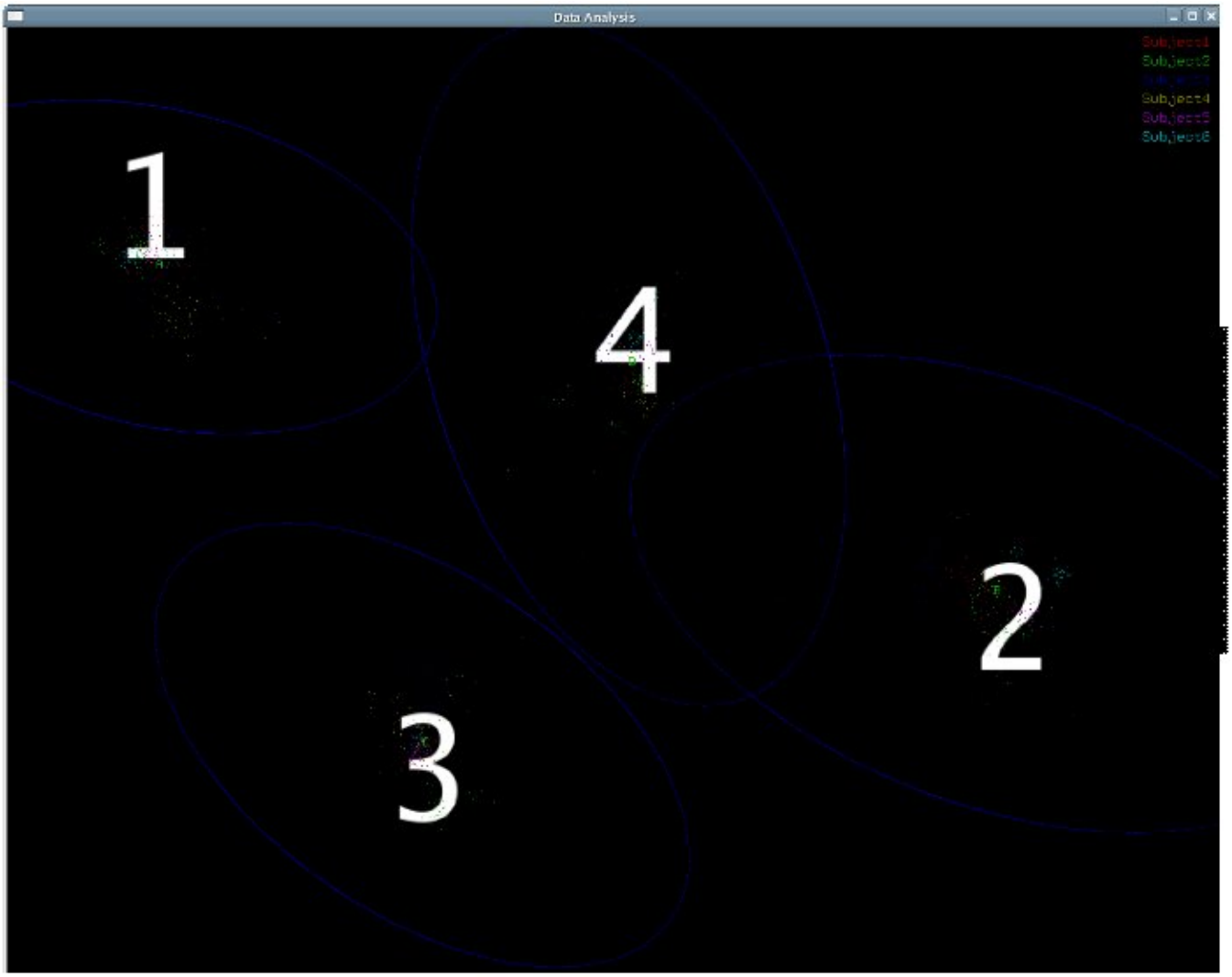

Figure 4: Sample data visualization ($\sigma_s = 100$)

5 Results

The parafoveal range that can be seen by humans in high detail is approximately 5° [Duchowski 2003]. With a viewer at 60cm away from the display, a 17 inch monitor, and a resolution of 1280 x 1024 pixels the parafoveal radius translates to approximately 100 pixels. Therefore, the data was analyzed with $\sigma_s = 100$ to cluster in respect to what a viewer can foveate on.

	S_p	S_s
numbers1	1.00	1.00
numbers2	1.00	0.88
numbers3	1.00	0.96
numbers1 (second run)	1.00	0.80
raytrace	0.83	0.43

Figure 5: Local results, $\sigma_s = 100$

The local results present an average of the measures (S_p and S_s) of all subjects for each image. The S_p values demonstrate that all subjects looked at all clusters in the numerical images and most clusters in the CG image. The S_s values are high for the numerical images expressing that the view sequences between subjects were very similar; the value was lower for the CG image indicating less relation between the view sequences of the subjects.

	S_p	S_s
Subject1	1.00	0.75
Subject2	1.00	0.50
Subject3	1.00	0.50
Subject4	1.00	0.75
Subject5	1.00	0.25
Subject6	1.00	0.50

Figure 6: Repetitive results (numbers1), $\sigma_s = 100$

The repetitive results present an average of the measures (S_p and S_s) for each of the viewer's two viewings of the numbers1 image. The outcome can be interpreted much in the same way as the previous results.

Since the σ_s value was selected based on human physiology it is possible that the data could have been skewed (too large/not enough clusters) relative to the image size. Furthermore the data was run with $\sigma_s = 70$ and $\sigma_s = 40$ exhibited in Figures 7-10.

	S_p	S_s
numbers1	1.00	1.00
numbers2	1.00	0.68
numbers3	090	0.83
numbers1 (second run)	1.00	0.72
raytrace	0.82	0.32

Figure 7: Local results, $\sigma_s = 70$

	S_p	S_s
Subject1	1.00	0.75
Subject2	1.00	0.50
Subject3	1.00	0.50
Subject4	1.00	0.75
Subject5	1.00	0.25
Subject6	1.00	0.25

Figure 8: Repetitive results (numbers1), $\sigma_s = 70$

	S_p	S_s
numbers1	1.00	1.00
numbers2	0.90	0.57
numbers3	0.90	0.77
numbers1 (second run)	0.87	0.57
raytrace	0.53	0.24

Figure 9: Local results, $\sigma_s = 40$

	S_p	S_s
Subject1	1.00	0.50
Subject2	1.00	0.25
Subject3	1.00	0.25
Subject4	0.75	0.50
Subject5	0.75	0.25
Subject6	1.00	0.0

Figure 10: Repetitive results (numbers1), $\sigma_s = 40$

6 Discussion

The results suggest that the algorithm works as expected. Both the location and sequential (local) measures were high for the numerical images and significantly lower for the CG image. Even when decreasing σ_s the pattern remained. The results for the repetitive measures did not reveal much significance because only two runs over the same image were conducted. This fact is not a great loss, because the same method is used to compute both local and repetitive measures (different values are averaged to come up with the repetitive outcome). Consequently, proof off the local results infers the correctness of the repetitive results barring any coding errors. Even so, further testing is warranted.

The local results for all but the first image suffered because data was collected uninterruptedly throughout each entire trial. This fact did not allow the subjects to reorient their eyes, and data from the previous image's last fixation was carried over to the next image. This problem could have been averted by briefly pausing the data collection when an image changed.

The experimentation yielded a worthwhile consideration. When testing a static scene for a short time low values should be used for σ_s to avoid too few clusters and thus possibly distorted or insignificant results.

7 Future Work

Future work will include porting the system from C to C++ for added readability and maintainability. Also a graphical interface will be designed to make program controls easier to access and display results in an easier to read format. While the idiosyncratic measure was not deemed necessary for the initial implementation, it could be useful in certain situations and will be included the future. After the tool is updated it will be used in future studies.

Acknowledgments

Thanks to Andrew Duchowski for his input, Thomas Grindinger for the ray traced image, and George Glatt IV for his participation.

References

DUCHOWSKI, A. T. 2003. *Eye Tracking Methodology: Theory and Practice*. Springer-Verlag Lodon Limited.

LAW, B., ATKINS, M. S., KIRKPATRICK, A. E., LOMAX, A. J., AND MACKENZIE, C. L. 2004. Eye Gaze Patterns Differentiate Novice and Experts in a Virtual Laparoscopic Surgery Training Environment. In *Proceedings of the Eye Tracking Research and Applications (ETRA) Symposium* 2004, 41-47.

PRIVITERA, C. M., AND STARK, L. W. 2000. Algorithms for defining visual regions-of-interest: Comparison with eye fixations. *IEEE Transactions on Pattern Analysis and Machine Intellegence* 22, 9, 970-982.

SANDASIVAN, S., GREENSTEIN, J. S., GRAMOPADHYE, A. K., AND DUCHOWSKI, A. T. 2005. Use of Eye Movements as Feedforward Training for a Synthetic Aircraft Inspection Task. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* 2005, 141-149.

SANTELLA, A., AND DECARLO, D. 2004. Robust Clustering of Eye Movement Recordings for Quantification of Visual Interest. In *Proceedings of the Eye Tracking Research and Applications (ETRA) Symposium* 2004, 27-34.

WOODING, D.S. 2002. Fixation maps: quantifying eye-movement traces. In *Proceedings of the Eye Tracking Research and Applications (ETRA) Symposium* 2002, 31-36.