

# ScanGraph: A Novel Scanpath Comparison Method Using Visualisation of Graph Cliques

Jitka Dolezalova  
Palacký University Olomouc,  
Olomouc, Czech Republic

Stanislav Popelka  
Palacký University Olomouc,  
Olomouc, Czech Republic

The article describes a new tool for analysing eye-movement data. Many different approaches to scanpath comparison exist. One of the most frequently used approaches is String Edit Distance, where gaze trajectories are replaced by sequences of visited Areas of Interest. In cartographic literature, the most commonly used software for scanpath comparison is eyePatterns. During an analysis of eyePatterns functionality, we found that the tree-graph visualisation of its results is not reliable. Thus, we decided to develop a new tool called ScanGraph. Its computational algorithms are modified to work better with sequences of different length. The output is visualised as a simple graph, and similar groups of sequences are displayed as cliques of this graph. This article describes ScanGraph's functionality on a simple cartographic eye-tracking study example. Differences in the reading strategy of a simple map between cartographic experts and novices were investigated. The paper should aid researchers who would like to analyse the differences between groups of participants, and who would like to use our tool, available at [www.eyetracking.upol.cz/scangraph](http://www.eyetracking.upol.cz/scangraph).


**Keywords:** eye movement, eye tracking, scanpath, scanpath comparison, string edit distance, usability, cartography, application

## Introduction

This article describes a new tool for scanpath comparison using visualisation of graph cliques. This tool will allow to find similarities between participants' process of presented stimuli observation. With information about their personal characteristics (age, sex, knowledge, etc.) it is possible to reveal if these groups are using a similar strategy. The output of the tool is a simple graph. In this graph cliques are identified. A clique is a subset of vertices in a graph where all vertices are connected by an edge with all of the others from that subset. These cliques represents participants with similar sequences of visited Areas of Interest (similar approaches to observing stimuli). The advantage over other scanpath comparison techniques is

that the visualisation highlights only those participants that are similar according to the user-defined value of the degree of similarity.

In the introduction, the history and background of scanpath comparison is described with an emphasis on the most frequently used method – String Edit Distance. Also, the software eyePatterns is mentioned. During analysis of the eyePatterns outputs, it was found that results were not reliable. Its weaknesses are described in the first part of the methods section. Based on these findings, we decided to develop a new tool called ScanGraph, which calculates the similarities between scanpaths and visualises results in the form of graph cliques. The basic theory of simple graphs and cliques is also described in the methods section. The results section contains detailed information about ScanGraph. The functionality of the application is presented practically on an example of a model case study from the field of cartography. The results provide brief information about the model case study, and then an example of practical use of ScanGraph is presented. In the discussion section, the limitations of ScanGraph are described together with future proposals how to eliminate them.

Received April 4, 2016; Published August 5, 2016.  
Citation: Dolezalova, J. & Popelka, S. (2016). ScanGraph: A novel scanpath comparison method using visualisation of graph cliques. Journal of Eye Movement Research, 9(4):5, 1-13.  
Digital Object Identifier: 10.16910/jemr.9.4.5  
ISSN: 1995-8692  
This article is licensed under a [Creative Commons Attribution 4.0 International license](https://creativecommons.org/licenses/by/4.0/). 

During analysis of eye-tracking data using an average of eye-movement measures as fixation counts and durations, eye-movement behaviour unfolding a particular sequence over time is ignored. This sequence is a rich source of information (Anderson, Anderson, Kingstone, & Bischof, 2014). To analyse sequences of eye-movements, a large number of methods comparing scanpaths has been developed. These methods are collectively known as scanpath comparison.

The beginnings of interest in distinctive scanning patterns can be found in the study of Noton and Stark (1971), who reported a qualitative similarity in eye-movements when people viewed line drawings on multiple occasions. This observation was used to support the “Scanpath Theory”, which proposed that visual features were encoded and stored alongside a motor memory of the scanpath made during perception. When a picture is seen again, it is recognised by executing the stored scanpath and matching the sequential features (Foulsham et al., 2012). The scanpath comprises sequences of alternating saccades and fixations that repeat themselves when a respondent is viewing stimuli. Scanpath comparison methods can be divided into six groups (String Edit Distance, ScanMatch, Sample-based Measures, Linear Distance, MultiMatch and Cross-recurrence Quantification Analysis). An overview of these methods and their comparison is described in Anderson et al. (2014).

One of the most frequently used methods is String Edit Distance, which is used to measure the dissimilarity of character strings. As Duchowski et al. (2010) mentions, scanpath comparison based on the String Edit Distance introduced by Privitera and Stark (2000) was one of the first methods to quantitatively compare not only the loci of fixations but also their order.

When using String Edit Distance, the grid or Areas of Interest (AOI) have to be marked in the stimulus. The gaze trajectory (scanpath) is then replaced by a character string representing the sequence of fixations with characters for AOIs they hit. Only 10 percent of the scanpath duration is taken up by the collective duration of saccadic eye-movements. Fixations in the created Areas of Interest took 90 percent of the total viewing period (Bahill & Stark, 1979). A sequence of transformations (insertions, deletions, and substitutions) is used to transform one string to another. Their similarity is represented as the number of transformation steps between two analysed strings (Anderson

et al., 2014). Foulsham et al. (2012) pointed to the disadvantage of String Edit Distance, which is reducing distances to binary classification (because of the necessity of dividing stimuli on a grid or creating Areas of Interest). For some applications, as in cartography, this disadvantage can be turned into an advantage – for example, when analysing the behaviour of respondents to map composition elements.

One of the most used metrics calculating the distance between sequences is called Levenshtein distance, named after the Russian scientist Vladimir Levenshtein (Levenshtein, 1966). The Levenshtein distance between two strings  $a = a_1a_2\dots a_{|a|}$ ;  $b = b_1b_2\dots b_{|b|}$  of length  $|a|$   $|b|$  (let us denote  $Lev(a, b)$ ) is the lowest number of deletions, insertions, or substitutions required to transform the source string into a target string. For example, the distance between sequences “gravitation” and “gravity” is equal to 5 (three changes and two deletions). Hence,  $Lev(a, b) \in \mathbb{N}_0$ ,  $Lev(a, b) = 0$  if and only if the strings are equal and  $Lev(a, b) = \max\{|a|, |b|\}$  if and only if there is any correspondence between the strings. The value of the Levenshtein distance increases with larger differences between the strings. The Levenshtein method was the first used for searchpath and scanpath analysis in the study of Choi, Mosley, and Stark (1995) and Stark and Choi (1996).

The other possible metric is called the Needleman-Wunsch algorithm with its scoring system. The Needleman-Wunsch algorithm (let us denote its value  $NW(a, b)$ ) searches for concordant elements between two strings  $a = a_1a_2\dots a_{|a|}$ ;  $b = b_1b_2\dots b_{|b|}$  of the length  $|a|$   $|b|$ . The basic scoring system used for our needs is given by Match reward = 1, Gap cost = 0 and Mismatch penalty = -1. For example, the distance between “gravitation” and “gravity” is equal to 6 (six matches). Hence,  $NW(a, b) \in \mathbb{N}_0$ ,  $NW(a, b) = \min\{|a|, |b|\}$ , when  $a$  is a subset of  $b$  or  $b$  is a subset of  $a$ . The value of  $NW(a, b)$  increases with the similarity between the strings.

In our issue, we want to count the distances between each pair of sequences from a certain set. Both of these metrics properly work when all of the compared strings have the same length. But when the lengths of the sequences are not equal, a serious problem arises. Let us show an example with Levenshtein distance. Let  $a = ABC$ ,  $b = DEF$ ,  $c = ABCDEFGHIJKLM$ ,  $d = ABDGHJKNOP$ . The distances are  $Lev(a, b) = 3$ ,  $Lev(c, d) = 7$ , thus  $Lev(c, d) >$

$Lev(a, b)$ . But whereas the sequences  $c$  and  $d$  have similar parts, the sequences  $a$  and  $b$  are totally different.

String Edit Distance measurement was used for the evaluation of web page design (Josephson & Holmes, 2002). Areas of Interest were marked on the web page, and alphabetic code was assigned to each of them. Then, the eye-path sequence for each subject's viewing of each web page by recording the sequence of fixations was created. Sequences were compared with the use of the Optimal Matching Analysis tool (Chan, 1995). Fabrikant, Rebich-Hespanha, Andrienko, Andrienko, and Montello (2008) analysed eye-movement data recorded in controlled experiments on small-multiple map (a series of similar maps using the same scale, allowing them to be easily compared) displays with the use of ClustalG software (Wilson, Harvey, & Thompson, 1999). Clustal software packages are widely used for analysing gene sequences in DNA and proteins. ClustalG was developed specifically to analyse social-science data. Based on the results of visual geoanalytical approaches with sequence alignment analysis techniques, it was found that small-multiple displays cannot generally be computationally or informationally equivalent to non-interactive animations (animations which cannot be controlled – merely the playback of the video).

In 2006, West, Haake, Rozanski, and Karn (2006) introduced the software eyePatterns – software that uses well-established sequence analysis algorithms designed primarily to aid eye-movement researchers in comparing sequence patterns within and across experimental groups of subjects. Apart from String Edit Distance, eyePatterns also integrates transition frequency analysis, clustering, sequence alignment, and pattern discovery.

For research in the field of cognitive cartography, String Edit Distance is the most important feature. Based on eye-movement data, this method can answer questions such as “How is it possible that one person orientates themselves in a map very quickly, while it takes others a long time?”, “Is there a difference in map reading between men and women?”, or “Do all people look at maps the same way?” In some cartographic studies, sequence alignment methods were also used for non-eye-movement data. For example Joh, Arentze, Hofman, and Timmermans (2002) developed a new measure for similarity between activity patterns in activity-travel patterns data. Shoval and Isaacson (2007) used sequence alignment for analysing sequential aspects within the temporal and spatial dimensions of human activities.

String Edit Distance in eyePatterns was used, for example, in the study by Coltekin, Fabrikant, and Lacayo (2010), who analysed dynamic visual analytics displays. Levenshtein (Levenshtein, 1966) and Needleman-Wunsch (Needleman & Wunsch, 1970) algorithms implemented in eyePatterns were used to generate a distance matrix. Data were visualised in eyePatterns with a tree-graph constructed by a hierarchical clustering algorithm. In this tree-graph, clusters of participants were identified visually. Together with Path Similarity Analysis (Andrienko, Andrienko, Burch, & Weiskopf, 2012) eyePatterns was also used in the author's study by Popelka, Dvorsky, Brychtova & Hanzelka (2013). The aim of the study was to identify the typology of map readers (common behavioural characteristics identical or similar between more individuals) based on their eye-movements while solving geographical tasks with the use of a map.

## Methods

### *eyePatterns and its disadvantages*

West et al. (2006) states that eyePatterns uses hierarchical clustering for calculating sequence similarity. Clustering partitions data into subsets of items that share similar traits. Agglomerative hierarchical clustering builds a hierarchy of clusters, beginning with the two most closely related sequences, and ending with the most distant sequence or cluster. The hierarchy tree can be visualised, exposing outlying and the most similar sequences (West et al., 2006).

When we analysed outputs from eyePatterns, we found that sequences with the lowest value of Levenshtein distance have (correctly) the closest possible distance (two edges between them), because the algorithm starts with them. These two sequences now make a cluster and distances in the matrix are recalculated using this cluster instead of the original nodes (sequences). eyePatterns uses the average of distances between the pair of sequences making a new cluster. Due to this clustering, the distances between nodes in the tree-graph are distorted. The distance is now calculated for the average value for the whole cluster. The problem is that the distance between particular nodes inside the cluster towards the other node can be lower than the distance between this node and the average value for the whole cluster. From the tree-graph, it is not possible to distinguish in which cases the distances between original sequences were used and where the average

for the cluster was used. Hence, tree-graph visualisation doesn't correspond to the statement used in the eyePatterns interface – "The fewer branches that are between two sequences, the more similar those sequences are", as is illustrated below in Figure 1.

Twenty scanpath strings (non-collapsed, marked as S1 – S20) from the stimulus used in model case study (see section "Model Case Study") were used to highlight the inaccuracy/errors of similarity calculation in eyePatterns. The tree-graph in Figure 1 displays the output of Levenshtein distance ("default scoring scheme") calculation from eyePatterns. Blue labels S1-S10 display participants GIS1-GIS10 (cartographers). Red labels S11-S20 stand for participants NOGIS1-NOGIS10 (non-cartographers).

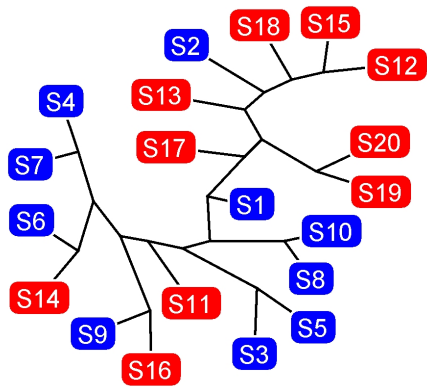


Figure 1. Output of eyePatterns – a tree-graph constructed by a hierarchical clustering algorithm

Figure 2 displays the tree-graph from Figure 1 with four highlighted sequences (participants). The closest highlighted pair in the tree-graph contains sequences S1 and S17. This should mean that the sequences are very similar. However, the Levenshtein distance between these two sequences was 13. Compare with the pair S12 and S14, lying on the opposite sides of the tree-graph. This should mean that the sequences differ a lot. The Levenshtein distance of these two sequences is only 4 – which means that only four changes are necessary to change one sequence to another. A similar situation is visible from the dendrogram (Figure 3) displaying the same data.

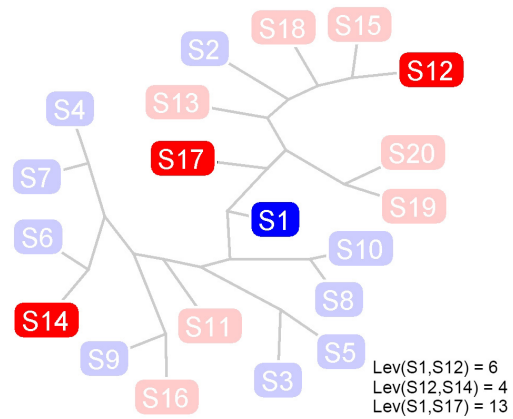


Figure 2. Tree-graph from eyePatterns with highlighted inaccuracies

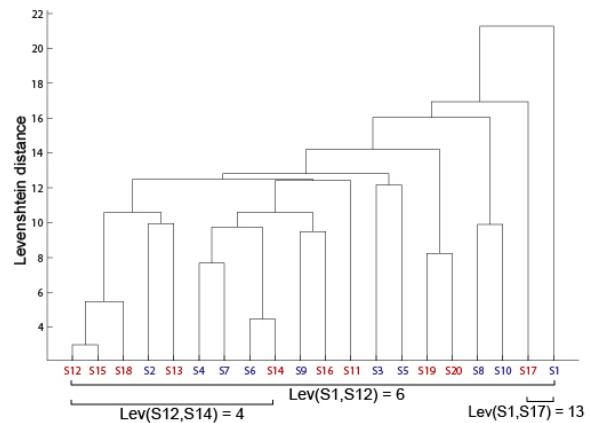


Figure 3. Tree-graph from eyePatterns redrawn to a dendrogram

Trajectories represented by sequences S1 and S17 were displayed in an OGAMA Scanpath module (Figure 4). It is evident that these two trajectories are very different as it is also obvious from their sequences (S1=EAAAAA AAAAAAAAAAAAAACCC, S17=AAAAACCBCCCC AAAA). Participant S1 (blue line) performed fewer fixations in the map. He also visited AOI E (Map title) and B (Map of Alaska), while no fixations from participant S17 (red line) were recorded there.



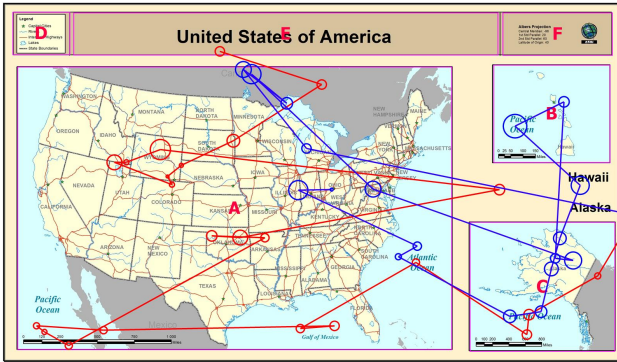


Figure 4. Comparison of trajectories S1 and S17, which were selected by eyePatterns as very similar

After discovering the inaccuracy of eyePatterns, we decided to develop our own application called ScanGraph for finding similar sequences in eye-tracking data. Compared with eyePatterns, where all sequences were included in the tree-graph, ScanGraph highlights only those sequences that are similar according to pre-set parameters. The result is displayed as a simple graph and similar groups are displayed as cliques of this graph.

### Novel approach – ScanGraph

Our aim was to create a new tool that will work on the principle of binary relation. The task of finding groups of similar elements is equivalent to the task of seeking tolerance classes of a tolerance relation. This is also equivalent to the problem of finding cliques in a simple graph and can be easily and clearly visualised. The necessary terms are defined below.

#### Binary relations

A binary relation between two sets  $A$  and  $B$  is a subset of the Cartesian product  $A \times B$ . A binary relation on set  $A$  is a subset of  $A \times A$ .

When an element  $a \in A$  is in a relation to an element  $b \in B$  we write  $aRb$ .

Given a binary relation  $R$  on a set  $A$  we have the following definitions:

A relation  $R$  on a set  $A$  is called reflexive if and only if  $aRa$  for every element  $a \in A$ .

A binary relation  $R$  on a set  $A$  is called symmetric if and only if for any  $a$  and  $b$  in  $A$ , whenever  $aRb$ , then  $bRa$ .

A binary relation  $R$  on a set  $A$  is called transitive, if and only if for any  $a, b$  and  $c$  in  $A$ , whenever  $aRb, bRc$ , then  $aRc$ .

A binary relation  $R$  on  $A$  is set to be a relation of equivalence if it is reflexive, symmetric, and transitive.

A partition of a set  $A$  is by definition a union of subsets  $A_i$  that cover  $A$  but do not intersect each other:  $A = \bigcup_{i=1}^n A_i, A_i \cap A_j = \emptyset, \forall i, j \in \{1, \dots, n\}$ . Given a relation of equivalence, we denote by  $[[a]]$  the class of equivalence of an element  $a$ :  $[[a]] = \{b \in A : aRb\}$ . Two elements have the same class if and only if they are in relation:  $[[a]] = [[b]] \Leftrightarrow aRb$ . This is a direct consequence of transitivity and symmetry. Hence, given a relation of equivalence on a set  $A$ , its classes of equivalence form a partition of the set  $A$ .

A binary relation  $R$  on a set  $A$  is set to be a relation of tolerance if it is reflexive and symmetric.

The notion of tolerance relation is an explication of similarity or closeness.

As an analogy of equivalence classes and partitions, here we have tolerance classes and coverings. A set  $B \subset A$  is called a tolerance preclass if it holds that for all  $a, b \in B$ ,  $a$  and  $b$  are tolerant, i.e.  $aRb$ . A maximum preclass is called a tolerance class. So two tolerance classes can have common elements.

Given a non-empty set  $A$ , a collection  $\prod_{i=1}^n A_i$  of non-empty subsets of  $A$  such that  $\bigcup_{B \in \Pi} B$  is called a covering of  $A$ . Given a tolerance relation on a set  $A$ , the collection of its tolerance classes forms a covering of  $A$ .

Every partition is a covering; not every covering is a partition (Chajda, 2005).

#### Simple graphs

A graph  $G = (V, E)$  is defined as a structure of two finite sets  $V$  and  $E$ . The elements of  $V$  are called vertices, and the elements of  $E$  are called edges. Each edge has a set of one or two vertices associated with it, which are called its endpoints.

An edge is said to join its endpoints. A vertex joined by an edge to a vertex  $v$  is said to be a neighbour of  $v$ .

A proper edge is an edge that joins two distinct vertices.

A self-loop is an edge that joins a single endpoint to itself.

A multi-edge is a collection of two or more edges having identical endpoints.

A simple graph has neither self-loops nor multi-edges (Gross & Yellen, 2005).

When we use the term graph without a modifier, we mean a simple graph.

### Cliques

A subset  $S$  of  $V(G)$  is called a clique if every pair of vertices in  $S$  is joined by at least one edge, and no proper superset of  $S$  has this property.

Thus, a clique of a graph  $G$  is a maximal subset of mutually adjacent vertices in  $G$ .

The clique number of a graph  $G$  is the number  $\omega(G)$  of vertices in the largest clique in  $G$ .

A complete graph is a simple graph such that every pair of vertices is joined by an edge (Gross & Yellen, 2005).

## Results

### ScanGraph Application

The application was created using PHP and C# (Backend) and D3.js (Frontend). Its interface can be seen in Figure 5. When the web page [www.eyetracking.upol.cz/scangraph/](http://www.eyetracking.upol.cz/scangraph/) is loaded, only the left column (1) is displayed. The user can select input data or try the functionality with a predefined source of data (1a). The application works with data exported from the application OGAMA (Voßkühler, Nordmeier, Kuchinke, & Jacobs, 2008). OGAMA

(OpenGazeAndMouseAnalyzer) is an open-source application design to record and analyse eye and mouse movement data. It allows Levenshtein distances between sequences to be calculated, but the output is just a matrix with distance values. It is not possible to find groups of similar participants. Sequence similarity measures from OGAMA can be exported to a text file – and this text file can be imported directly to ScanGraph. This is one of the advantages over eyePatterns, which needs data in a specific format prepared in a text file or table processor as necessity.

Then, the user can specify the method of computation (1b) and select between collapsed or original strings (1c). In collapsed strings, there are no successive characters (AOIs) in the sequence. In the last step, the user can display an advised graph (1d) or construct a graph according to parameter  $p$  (1e) or percentage of edges (1f) (see below).

After clicking on the button “Advised graph” or “Compute graph”, elements 2 – 7 are added to the display. In element 2, the points (vertices) representing all sequences of participants from the input dataset are shown. Different colours represent the affiliation to the category (e.g. male/female, expert/novice, etc.) The table on the left

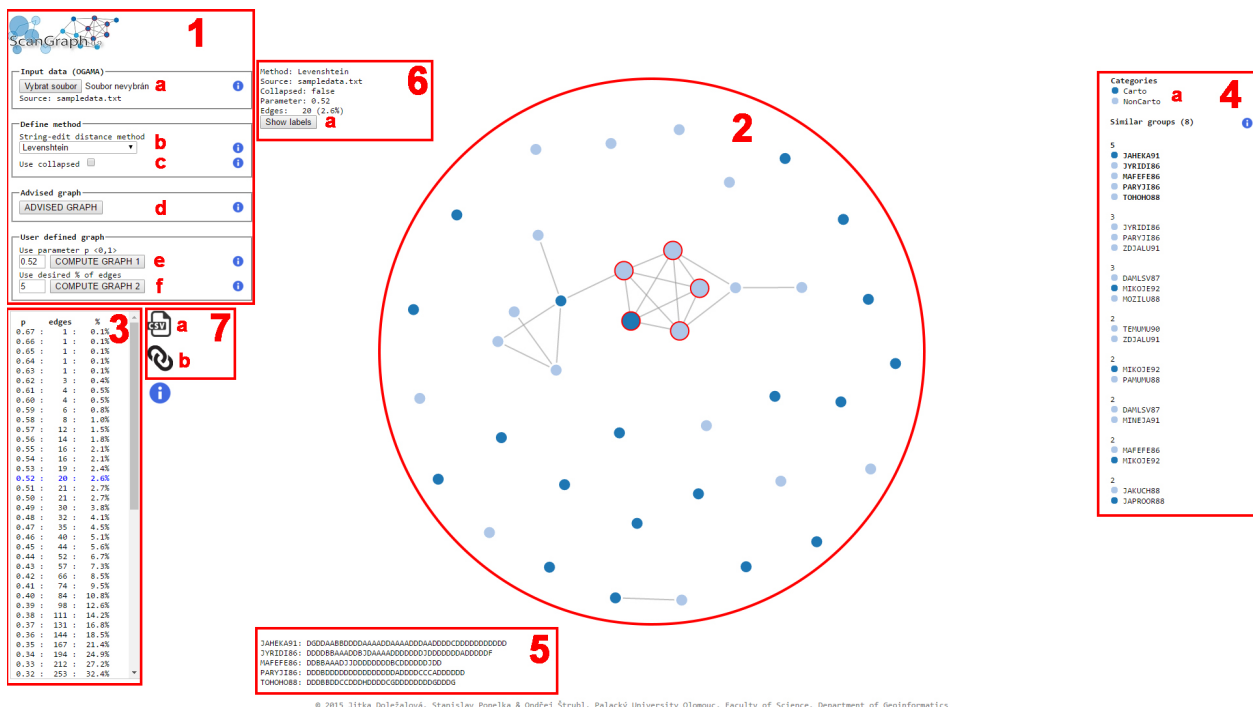


Figure 5. Interface of the ScanGraph application

(3) contains 100 pre-computed values of parameter  $p$ . In addition to parameter values, the number of edges is visible. The last column contains the percentage of edges from the complete graph. The user can click into the table to display the particular graph.

Cliques with two and more vertices found in these graphs are listed on the right side of the page (4). Colour points in front of the participant's name represents their group. An explanation of colours can be found in the upper part (4a). After clicking on any group of similar sequences in this section of the page, the clique is highlighted in the graph, and strings are also shown in the bottom part of the page (5). The user can visually inspect the sequence of the characters in the string. The overview of settings is shown in the upper part of the page (6), and the user can add labels with subject names to the graph (6a).

By clicking on the icon (7a), the user can download the table with all matrices (original matrix, modified matrix, and adjacency matrix), listed similarity groups with their character strings, input data, and overview of the settings. Clicking on (7b) allows a permanent link to the displayed graph to be created.

### *Computations and Visualisation*

At first, the distance matrix (original matrix  $D = (d_{ij})$ ) is constructed according to the Levenshtein or Needleman-Wunsch algorithm. Each element of the matrix  $d_{ij}$  is a distance between sequences  $i$  and  $j$ . The distances, however, are poorly comparable between themselves because of the different lengths of the sequences, as mentioned above. According to this, the value of a distance  $d_{ij}$  is divided by the higher value of the length of sequences  $i$  and  $j$ . It is the highest distance the two sequences could have (in the case of Levenshtein) or the value of the greatest similarity the two sequences could have (in the case of Needleman-Wunsch) (modified matrix  $M = (m_{ij})$ ). Obviously, now the new elements could have a value from the interval  $(0,1)$  and still apply the higher the value is (in the case of Levenshtein), or the lower the value is (in the case of Needleman-Wunsch) the more different the strings are. The next step is up to the user.

The first option is to select the Advised graph button. This button returns a graph with 5% of the possible edges and a corresponding value of parameter  $p$  (see below). This graph is user-friendly and according to our experi-

ences has a very high interpretive value about any similarities. This option is recommended for users with no experience with ScanGraph. The second option is to construct a user-defined graph. The graph is created according to parameter  $p$  or percentage of edges.

The parameter  $p$  takes its value from the interval  $(0,1)$  and represents the degree of similarity. The higher the value of  $p$ , the higher the similarity of the given sequences. Obviously,  $p_{ij} = 1 - m_{ij}$  applies in the case of Levenshtein distance and  $p_{ij} = m_{ij}$  in the case of the Needleman-Wunsch algorithm  $\forall i, j \in \{1, \dots, n\}$  of the modified matrix, where  $n$  is the number of participants. The value of  $p$  constructs a new matrix (adjacency matrix  $A = (a_{ij})$ ) according to these conditions:

$$a_{ij} = \begin{cases} 1, & \text{if } p \geq p_{ij}, \\ 0, & \text{otherwise.} \end{cases}$$

Hence, the adjacency matrix represents a simple graph, which is displayed. The second option is to set a percentage of edges. This number takes a value from the interval  $< 0,100 >$ . The algorithm finds a value of the parameter  $p$  for which the graph will have a given percentage of edges (eventually rounded to the nearest lower value) and displays the graph and parameter  $p$ .

Besides the graph itself, a table with three columns is displayed. Parameter  $p$  with its 100 possible values and number of edges, and the percentage of the corresponding graph.

Each graph is represented by its adjacency matrix. Using the matrix, all cliques contained in the graph can be found. Each clique represents a group of sequences which has the same or higher degree of similarity than the given parameter  $p$ .

The maximal clique problem (finding all maximal cliques in a graph) is an NP-complete decision problem.

Definitions of the decision problem according to (Gross & Yellen, 2005) follows.

A decision problem is a problem that requires only a yes or no answer regarding whether some element of its domain has a particular property.

A decision problem belongs to the class P if there is a polynomial-time algorithm to solve the problem.

A decision problem belongs to the class NP if there is a way to provide evidence of the correctness of a yes answer so that it can be confirmed by a polynomial-time algorithm.

A decision problem  $R$  is polynomially reducible to  $Q$  if there is a polynomial-time transformation of each instance  $I_R$  of problem  $R$  to an instance  $I_Q$  of problem  $Q$  such that instances  $I_R$  and  $I_Q$  have the same answer.

A decision problem is NP hard if every problem in class NP is polynomially reducible to it.

An NP-hard problem  $R$  is NP-complete if  $R$  is in class NP.

The algorithm default used by ScanGraph is based on an exhaustive algorithm and finds an optimal solution with all cliques with two or more vertices in the given graph. When the computational time is too large, ScanGraph uses a heuristic algorithm.

Figure 6 displays an example of the influence of the parameter  $p$  value. On each image, the largest clique is marked. When the parameter is set to 0, the graph is always a complete graph. As the value of a parameter is increased, vertices very different from each other are dropped from the largest clique. When the value of a parameter is set to 0.5, the first vertex is out of the graph – it is not similar to any other in the dataset. The largest value of the parameter in the figure is 0.9 and only two vertices are making a clique. The character strings of these vertices were exactly the same, so in this case, it was needless to include an image with the graph with parameter 1, because it will be exactly the same as the previous image.

## Model Case Study

The functionality of the developed ScanGraph tool was presented in an example of a case study comparing different map compositions. The data were recorded as a part of work by students. The aim of the study was to reveal whether cartographers and non-cartographers perceive maps differently.

The experiment contained a total of 18 stimuli. Six types of maps were created and each of them was presented with three different map compositions. The distribution of map elements (map, legend, title, imprint, addition map) were placed at various positions in the stimuli.

A total number of 20 respondents participated in this eye-tracking study. Half of them were selected from a group of undergraduate students who had already attended a cartography course. The rest of them were selected from among students in fields of study other than cartography. The differences between cartographers and non-cartographers were investigated. For the case study, an eye-tracking device SMI RED 250 was used, and data were recorded with a frequency of 60Hz. Eye positions were recorded every 16ms. Eyes move in a number of different ways, simultaneously responding to commands from a number of different brain areas. One of the most important types of eye movement is no movement at all, but rather the ability to keep the eye trained on a fixed spot in the world. This is known as fixation. To get from one fixation to the next, the eyes make rapid, ballistic movements known as saccades (Hammoud & Mulligan, 2008). Plenty of algorithms for fixation detection exist, but the

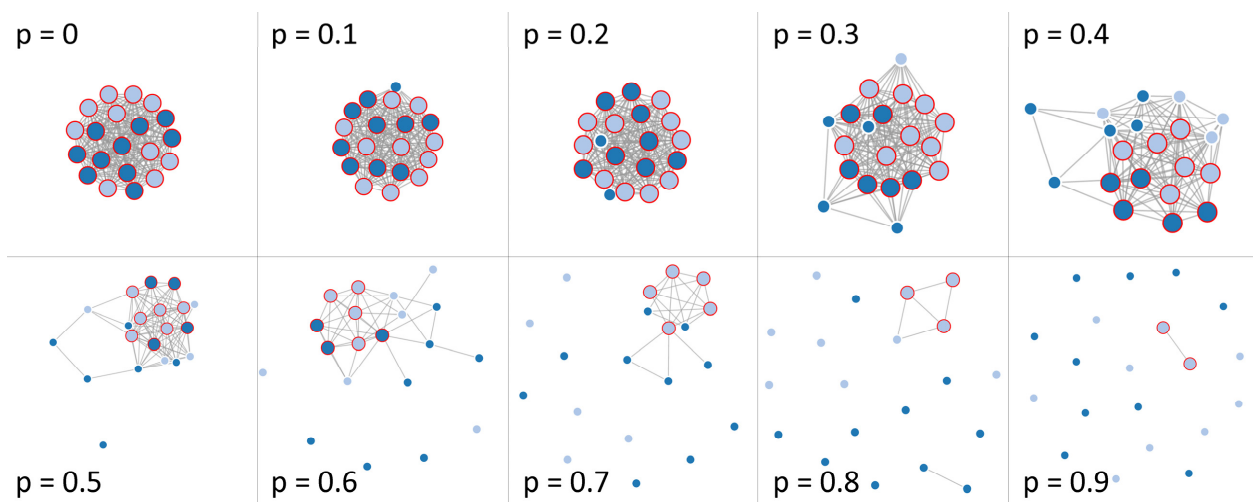


Figure 6. Example of the influence of parameter  $p$  value

most used for low-speed data (up to 250 Hz) is I-DT. I-DT takes into account the close spatial proximity of eye position points during eye movement trace (Salvucci & Goldberg, 2000).

For the case study, the software OGAMA was used with an ID-T algorithm for fixation detection. In OGAMA, the most important parameters for fixation detection are “Maximum distance” and “Minimum number of samples”. Thresholds in OGAMA were set to 15px (distance) and 10 samples. More information about this setting is described in (Popelka & Doležalová, 2015).

### Results of the Model Case Study

Recorded eye-movement data were visualised using the Sequence Chart method available in SMI BeGaze. The Sequence Chart shows the temporal sequence of the visited Areas of Interest. Figure 7 shows a Sequence Chart for all respondents for three different map compositions. Areas of Interest were marked around all map composition elements. The colour of the stripes under the maps represents a distribution of respondent attention between these AOIs. From a visual analysis of the Sequence Chart, a difference between the group of cartographers and non-cartographers can be seen. The most prominent difference is in the map element title (blue).

A fixation cross preceded each stimulus, so AOI representing the map field is always viewed in the first 500ms. Beyond this time, most cartography students automatically read the title of the map, or rather, noted fixations representing it in AOI. Non-cartographers did not do so. It is evident especially in the first column, where the stimulus was an “ideal” map composition. In the following columns, the composition did not obey cartographic rules. Despite this fact, students of cartography were trying to find the title of the map.

The Sequence Chart is illustrative and easy to interpret, but a deeper analysis of differences between the strategies of participant groups needs a more sophisticated method of analysis. In this paper, data from this short study were used for demonstrating the use and possibilities of the developed ScanGraph web application. More specifically, eye-movement data recorded during observation of map composition #1 (first column of Figure 7) were used.

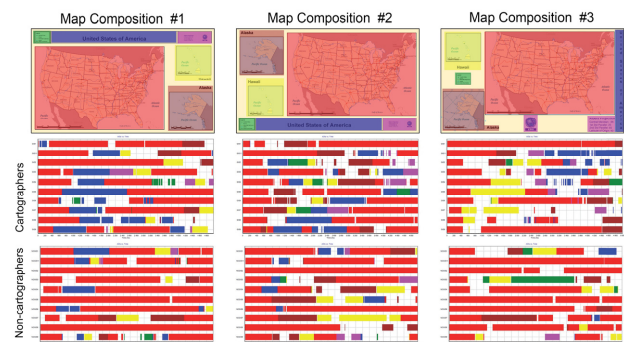


Figure 7. Sequence chart visualisation of participant reading strategy of three different map compositions. Map composition #1 was used for a model case study.

### Demonstration of using the ScanGraph Application

The ScanGraph application was designed to work with data exported from the open-source application OGAMA (Voßkühler et al., 2008) – An open source software designed to analyse eye and mouse movements in slideshow study designs. OGAMA contains a tool called “Levenshtein Distance Calculation”, which is capable of computing Levenshtein distances between the trajectories of participants. Sequence similarities can be calculated based on the regular grid or user defined Areas of Interest.

The output of this tool is a matrix of similarities between sequences and also the list of scanpath strings for each participant. When using ScanGraph, only the strings are important. The values of similarity calculated in OGAMA are not used, because the Levenshtein algorithm was modified to take into account different lengths of strings.

The first step of data analysis with ScanGraph is the creation of Areas of Interest above analysed stimuli. In our case, map composition #1 from Figure 7 was used, and Areas of Interest were marked around map composition elements (see Figure 8).





Figure 8. Areas of Interest marked in the stimulus with Map composition #1

In the next step, the Scanpath module in OGAMA was used to display the trajectories of participants and scanpath strings. The text file with sequence similarities can be exported from OGAMA and directly used as input data in ScanGraph.

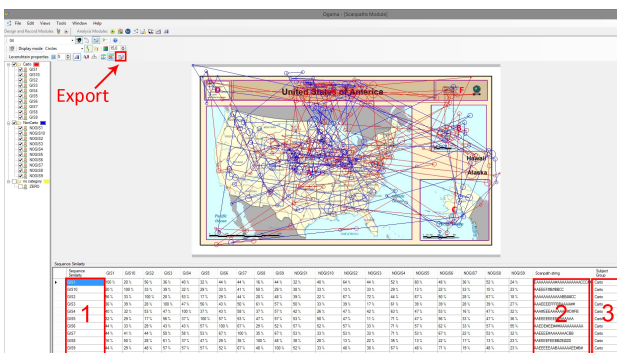


Figure 9. An environment of OGAMA's Scanpath module. Levenshtein distances between selected participants (and set of AOIs) are calculated and can be exported as a text file. Only Subject names (1), Scanpath strings (2), and affiliation to subject groups (3) is used in ScanGraph.

The exported text file was then opened with ScanGraph. The user can choose between the Levenshtein and Needleman-Wunsch algorithm and construct a graph.

In this particular example, the order of visited areas as well as their number of fixations was investigated. Therefore, the original (non-collapsed) data were analysed with the Levenshtein algorithm. The user can modify the value of parameter  $p$  or percentage of edges. In our case, we started with the parameter value 0.8, and two cliques were found (Figure 10).

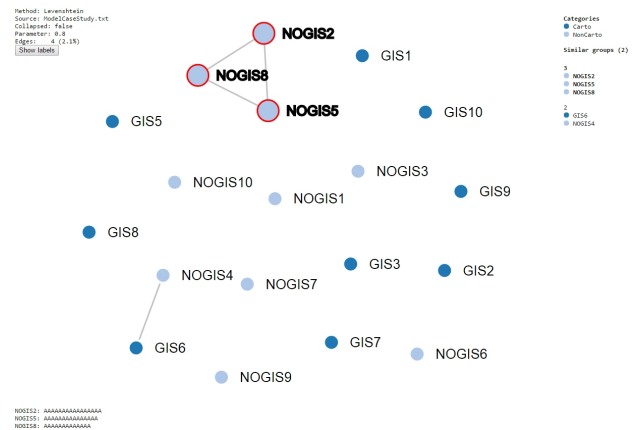


Figure 10. Output of ScanGraph showing two cliques based on parameter  $p = 0.8$ .

Figure 11 shows the trajectories of participants making similar groups with parameter  $p = 0.8$  (Figure 10). The larger one contained three participants from the group of non-cartographers. All of them spent their whole observation time in the AOI A (containing the map field). The other clique comprised two participant sequences and is displayed in shades of blue. Both participants performed the same number of fixations (17), mainly in the map field (AOI A) and map title (AOI E). Participant GIS6 (dark blue) made an extra fixation in the map legend.

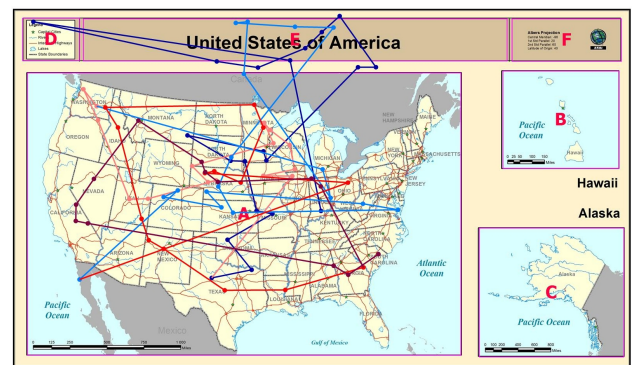


Figure 11. Trajectories of five participants making two cliques (based on parameter  $p = 0.8$ ). Three participants displayed in shades of red spent the whole time in the map field. Participants GIS6 and NOGIS4 (shades of blue) visited the map field and map title.

When the value of the parameter was decreased to 0.75, a total of six cliques was found in the data (Figure 12). The group of three similar participants was (obviously) preserved, but it was extended by participant NOGIS4 (in the case of the first clique), respectively GIS2 (in the case of the second clique). This means that participants NOGIS4

and GIS2 are both similar to the group of three participants from Figure 10 (NOGIS2, NOGIS5, and NOGIS8), but are not similar to each other.

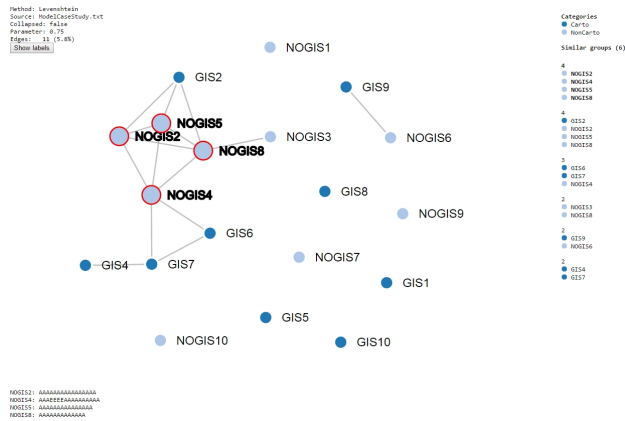


Figure 12. ScanGraph output showing two cliques based on parameter  $p = 0.75$ .

The trajectories from Figure 12 are displayed in Figure 13. Trajectories of participants NOGIS2, NOGIS5, and NOGIS8 shown in shades of red are again displayed in shades of red. The blue trajectory represents participant NOGIS4, and the green belongs to participant GIS2. Both trajectories are similar to the red ones, but they are not similar to each other.

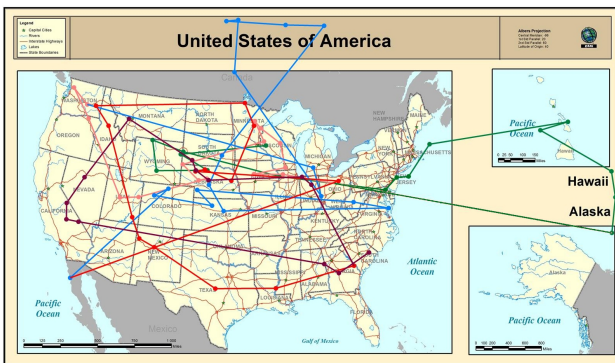


Figure 13. Trajectories of five participants making two cliques (based on parameter  $p = 0.75$ ). Participants NOGIS2, NOGIS5, and NOGIS8 are displayed in shades of red. Participants NOGIS4 (blue line) and GIS2 (green line) are both similar to the red ones, but not to each other.

## Discussion

As already mentioned in the introduction, plenty of scanpath comparison methods exist (Anderson et al., 2014). For cartographic research (and not only), there is an

advantage in using String Edit Distance based on Areas of Interest marked around map composition elements.

Until now, scanpath comparison in cartography was performed by using the eyePatterns application (West et al., 2006), which offers a variety of functions, but in cartography, only evaluating similarity between sequences was previously used. We have found that the implementation of Levenshtein and Needleman-Wunsch algorithms in eyePatterns is correct, but it is not appropriate for comparing strings with different length. Visualisation of results via tree-graphs is inaccurate and misleading, so we decided to develop our own tool – ScanGraph. We believe that our tool offers more useful results than eyePatterns in this specific functionality, but the variety of functionality (i.e. search for patterns) in eyePatterns can still be used for some applications.

The case study presented in this article demonstrated the use of the ScanGraph application. The goal of the case study was not to find anything important from the trajectories of participants but to present ScanGraph functionality. For that reason, only one stimulus observed by 20 participants was investigated.

During the development of ScanGraph, we ran into several problems, but most of them were solved or bypassed. As was mentioned above, the exhaustive algorithm is  $NP$ -complete. Due to this, with an increasing number of edges, the computational time increases non-polynomially. In that case, a heuristic algorithm is used. The heuristic algorithm might not find an optimal solution, i.e. all maximal cliques (Vecerka, 2007). However, the graphs where a solution could be found by an exhaustive algorithm have a higher interpretative value for the experiment.

ScanGraph is still under development. The next step will be to add a matrix of differences between AOIs. For some experiments, it may be important to define the cost of transitions between each pair of AOI separately. For example, transition from AOI A to AOI B (e.g. map vs. legend) could mean a more important change than transition from AOI C to AOI D (e.g. two columns of the legend), so the Levenshtein distance should be different. For this case, the user will define his own matrix of differences between AOIs.

Another possible improvement could be the computation of similarities between participants for a whole experiment. The user will upload a compressed file containing character strings from all stimuli of the experiment. The



global similarity between all participants throughout all stimuli will be again displayed as a graph.

## Conclusion

This article describes the possibilities of a newly developed application for scanpath comparison called ScanGraph. The application performs scanpath comparison based on the String Edit Distance method, and its output is a graph. Groups of similar sequences/participants are displayed as cliques of this graph.

ScanGraph can be used for all studies where differences between gaze movements in different groups of participants are investigated. ScanGraph works with an exported file from the open-source application OGAMA. To use ScanGraph, it is necessary to create Areas of Interest around specific parts of analysed stimuli. OGAMA allows a text file containing character strings representing the order of visited AOIs to be exported. This file can be directly imported to ScanGraph, which is freely available at [www.eyetracking.upol.cz/scangraph](http://www.eyetracking.upol.cz/scangraph). In the ScanGraph web environment, the user can display groups of participants with similar character strings – participants with a similar strategy. The user can calculate an advised graph (containing 5% of edges) or a user defined graph based on parameter (percentage of similarity) or percentage of edges. Groups of similar participants are marked in this graph and the user can quickly inspect their character strings.

Until now, the eyePatterns application was commonly used for this purpose. We have found that the eyePatterns output, a tree-graph showing similarity between all sequences, does not reflect the similarity measured by the algorithms used. From the tree-graph, similar groups can only be found visually, which is very inaccurate. Our approach does not connect all the sequences (participants), but created groups of similar participants correspond to the computations. The user knows that an identified group is similar according to a given parameter – which is not possible in eyePatterns. The algorithms for calculating similarity were modified and work better with strings of different length.

The functionality of ScanGraph was presented in an example of a simple cartographic case study in detail. This paper can serve as a user manual for ScanGraph.

## Acknowledgements

This paper is supported by the projects of Operational Program Education for Competitiveness – European Social Fund (projects CZ.1.07/2.3.00/20.0170), of the Ministry of Education, Youth, and Sports of the Czech Republic and the student project IGA\_PrF\_2016\_008 of Palacky University.

The authors declare that there is no conflict of interest regarding the publication of this paper.

## References

- Anderson, N. C., Anderson, F., Kingstone, A., & Bischof, W. F. (2014). A comparison of scanpath comparison methods. *Behavior research methods*, 1-16.
- Andrienko, G., Andrienko, N., Burch, M., & Weiskopf, D. (2012). Visual Analytics Methodology for Eye Movement Studies. *Visualization and Computer Graphics, IEEE Transactions on*, 18(12), 2889-2898.
- Bahill, A. T., & Stark, L. (1979). The trajectories of saccadic eye movements. *Scientific American*, 240(1), 108-117.
- Coltekin, A., Fabrikant, S., & Lacayo, M. (2010). Exploring the efficiency of users' visual analytics strategies based on sequence analysis of eye movement recordings. *International Journal of Geographical Information Science*, 24(10), 1559-1575.
- Duchowski, A. T., Driver, J., Jolaoso, S., Tan, W., Ramey, B. N., & Robbins, A. (2010). *Scanpath comparison revisited*. Paper presented at the Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications.
- Fabrikant, S. I., Rebich-Hespanha, S., Andrienko, N., Andrienko, G., & Montello, D. R. (2008). Novel method to measure inference affordance in static small-multiple map displays representing dynamic processes. *Cartographic Journal, The*, 45(3), 201-215.
- Foulsham, T., Dewhurst, R., Nyström, M., Jarodzka, H., Johansson, R., Underwood, G., & Holmqvist, K. (2012). Comparing scanpaths during scene encoding and recognition: A multi-dimensional approach.
- Gross, J. L., & Yellen, J. (2005). *Graph theory and its applications*: CRC press.

- Hammoud, R. I., & Mulligan, J. B. (2008). Introduction to Eye Monitoring *Passive Eye Monitoring* (pp. 1-19): Springer.
- Chajda, I. (2005). *Úvod do algebry* (Vol. 1). Olomouc: Vydavatelství Univerzity Palackého.
- Chan, T. W. (1995). Optimal matching analysis: a methodological note on studying career mobility. *Work and occupations*, 22(4), 467-490.
- Choi, Y. S., Mosley, A. D., & Stark, L. W. (1995). String editing analysis of human visual-search. *Optometry and Vision Science*, 72(7), 439-451. doi:10.1097/00006324-199507000-00003
- Joh, C.-H., Arentze, T., Hofman, F., & Timmermans, H. (2002). Activity pattern similarity: a multidimensional sequence alignment method. *Transportation Research Part B: Methodological*, 36(5), 385-403.
- Josephson, S., & Holmes, M. E. (2002). *Visual attention to repeated internet images: testing the scanpath theory on the world wide web*. Paper presented at the Proceedings of the 2002 symposium on Eye tracking research & applications.
- Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8), 707-710.
- Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3), 443-453.
- Noton, D., & Stark, L. (1971). Scanpaths in saccadic eye movements while viewing and recognizing patterns. *Vision Research*, 11(9), 929-938. doi:http://dx.doi.org/10.1016/0042-6989(71)90213-6
- Popelka, S., & Doležalová, J. (2015). Non-photorealistic 3D Visualization in City Maps: An Eye-Tracking Study *Modern Trends in Cartography* (pp. 357-367): Springer.
- Popelka, S., Dvorsky, J., Brychtova, A., & Hanzelka, J. (2013). User typology based on eye-movement paths *Geoconference on Informatics, Geoinformatics and Remote Sensing - Conference Proceedings, Vol I* (pp. 1041-1048).
- Privitera, C. M., & Stark, L. W. (2000). Algorithms for defining visual regions-of-interest: Comparison with eye fixations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(9), 970-982.
- Salvucci, D. D., & Goldberg, J. H. (2000). *Identifying fixations and saccades in eye-tracking protocols*. Paper presented at the Proceedings of the 2000 symposium on Eye tracking research & applications.
- Shoval, N., & Isaacson, M. (2007). Sequence alignment as a method for human activity analysis in space and time. *Annals of the Association of American Geographers*, 97(2), 282-297.
- Stark, L. W., & Choi, Y. S. (1996). Experimental metaphysics: The scanpath as an epistemological mechanism. *Advances in psychology*, 116, 3-69.
- Vecerka, A. (2007). Grafy a grafové algoritmy. *Univerzita Palackého Přírodovědecká Fakulta Katedra Informatiky, Olomouc*.
- Voßkübler, A., Nordmeier, V., Kuchinke, L., & Jacobs, A. M. (2008). OGAMA (Open Gaze and Mouse Analyzer): open-source software designed to analyze eye and mouse movements in slideshow study designs. *Behavior research methods*, 40(4), 1150-1162.
- West, J. M., Haake, A. R., Rozanski, E. P., & Karn, K. S. (2006). *eyePatterns: software for identifying patterns and similarities across fixation sequences*. Paper presented at the Proceedings of the 2006 symposium on Eye tracking research & applications.
- Wilson, C., Harvey, A., & Thompson, J. (1999). *ClustalG: Software for analysis of activities and sequential events*. Paper presented at the IATUR Conference Proceedings.