Fixation detection in eye-tracking data

Umar Bin Qushem
University of Eastern Finland
Joensuu, Finland
umarbin@student.uef.fi

Silvia Fontfreda Casals
University of Eastern Finland
Joensuu, Finland
silviaf@student.uef.fi

Peter Gazdík
University of Eastern Finland
Joensuu, Finland
peterg@student.uef.fi

Roman Oravec
University of Eastern Finland
Joensuu, Finland
romano@student.uef.fi

Essi Kirveskari
University of Eastern Finland
Joensuu, Finland
essikir@student.uef.fi

ABSTRACT

One of the main challenges in eye-tracking analysis and data collection is to identify the saccades and the fixations of a subject's gaze. Many algorithms have been designed to achieve a successful identification and classification for saccades and fixations. For this work, we present a detailed description, design, implementation and evaluation of the outcome results for the I-VT algorithm.

KEYWORDS

eye-tracking algorithm, fixation detection, gaze-tracking algorithm, data analysis, I-VT

INTRODUCTION

Many uses for eye tracking technologies exist, in any task where one's gaze is used, an application of a technology of this nature can be found. In the existing literature regarding eye tracking, multiple fields of application can be found, for instance, in Andrew T. Duchowski's *A breadth-first survey of eye-tracking applications* [1] a long list of uses in different scenarios can be found. In his work, Duchowski describes the use of eye-tracking methodologies in fields such as neuroscience, psychology, industrial engineering and human factors, marketing/advertising and computer science. Articles dedicated to a more specific sector can also be found, some of them including study cases: D. Riby and P. J. B. Hancock's study [6] compares where two groups of subjects focus their attention to. The first group of individuals has autism (a developmental disorder which one of its syndromes is the lack of desire for social interaction) and subjects from the second group have Williams syndrome (a genetic disorder that causes, among other symptoms, a higher interest in social interaction) and concludes than those with Williams syndrome are more likely to focus on human face's while those with autism do not pay as much attention to it.

When it comes to the technology itself it is composed by hardware and software. Currently, the eye tracking devices can be fit in one of three categories: contact lens / eye-attachable devices, optical video-based tracking and electrooculogram measure around the eyes. It is important to mention that the apparatus used to gather data from eyesight have been evolving and adapting to more modern technologies and techniques. In Robert Gabriel Lupu and Florina Ungureanu [4] work, a summary of the evolution of eye tracking apparatus are presented. Depending on the type of device that is being used for the data collection (and if there are any additional censors) different software and algorithms to process and analyse the data are going to be chosen. There are two main parts related to the eye-tracking data: Obtaining and processing. The software used for gathering the data strictly depends on the type of device that is being used; for example, in case of using a video-based apparatus, the software is going to work with reflections caused in the eyes, however, if the test is being carried with electric sensors there will be no reflection but electric impulses. An example of an algorithm for a video-based eye tracking device is the Starburst algorithm [3] designed by Dongheng Li and Derrick J. Parkhurst. The processing part should be divided in two sub-parts: pre-processing and classification. The main task of pre-processing it so de-noise the data to obtain information easier to work with. Once the data has been denoised it is possible to start the identification of fixations and saccades, which is the main objective of eye tracking applications. A few algorithms have been designed to carry out this task. Dario D. Salvucci and Joseph H. Goldberg5 present a taxonomy to describe, evaluate and compare five of the most popular algorithms to label saccades and fixations: two velocity based algorithms (Velocity-Threshold Identification and Hidden Markov model fixation identification), two dispersion based algorithms (Dispersion-Threshold Identification and Minimum Spanning Trees Identification) and finally a are based algorithm, Area of Interest Identification.

DATA

For our data analysis, we used a dataset consisting of raw gaze locations. The data was recorded in real time from participant's activities with a sampling rate of 1000Hz. Participants have been shown multiple images and were supposed to analyse and possibly recognise their contents. In order to perform the eye's fixation detection using I-VT algorithm, our data is structured into sid (subject id), known, x1, y1 variables in a single csv file. "Known" is a boolean variable indicating whether the subject recognised the contents in a particular image. Coordinate system, where the (0,0) point is located in the centre of the screen, has been used.

ALGORITHM: VELOCITY-THRESHOLD IDENTIFICATION

I-VT is a velocity-based algorithm and its main idea is to classify some velocities between two given points as fixations or saccades. The pseudocode of the algorithm is in Listing 1. In the presented implementation there are two inputs for the algorithms: the threshold value and the data provided by the eye tracking device. The mentioned data must include a list of points the eye sight was being directed to and temporal information such as the frequency the points in the data were being recorded in for further calculations of velocities.

The first step is to calculate the velocities; it is a simple task since the algorithm has the data of the points and the time in between them (or some other data from which the time can be calculated out of). Depending on possible peculiarities of the scenario it might be the case where additional calculations are needed, e.g: the subject is looking around in a 3D scenario with multiple elements in different distances from their position. The velocity between point A to B is calculated by dividing the Euclidean Distance by the time difference between A and B. Once all velocities in between each point are calculated the labelling/classification step can start. To decide if a point is a fixation or saccade the velocity is compared to the threshold and if it is lower it is classified as fixation and if it is higher the point is considered a saccade. Finally, once the fixations are identified, a centroid point for each of them is calculated.

Listing 1: Pseudocode of IV-T algorithm.

```
IVT(potocol, threshoold)
{
        velocities[]
        values[]
        for each element in protocol
        {
                velocities[element] = vel(protocol[element],
                                          protocol[element+1])
        }

        for each element in velocities
        {
                if (velocities[element] > threshold)
                        values[element] = saccade
                else
                        values[element] = fixation
        }
return values
}
```

Specification of the threshold

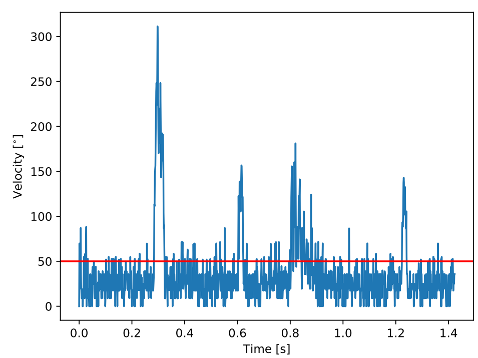
The I-VT algorithm is one of the simplest fixations and saccades labelling algorithm, however, the specification of the threshold is a complex part. Usually the value of the threshold varies significantly depending on the input data, for example, it should be considered a lower value for the threshold in a scenario where the subject is reading that if the subject is driving a car. Kenneth Holmqvist [2] suggests that the value of the threshold should be between to 10 to 50 $^{\circ}/s$.

ANALYSIS

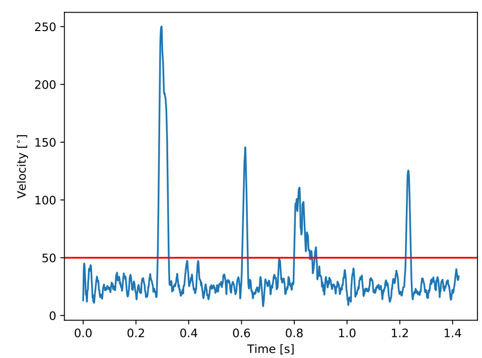
To perform the analysis of the data Python 3.7 was used. Firstly, we performed conversion from the unit-coordinates to the real world coordinates, otherwise we would not be able calculate distances using the Euclidean distance. Secondly, we implemented the naive I-VT implementation mentioned in section 3. Once the results were obtained, we observed that the algorithm was producing an inaccurate classification, therefore it was necessary to perform some additional processing on the input data and the output results. After experimenting with different thresholds we established the threshold for our I-VT implementation would be 50°/s.

Pre-processing of the input data

Pre-processing on the input data is necessary since the measured data contains a lot of noise. Since we are calculating the velocities from point to point the noise in the input can have a major impact on the analysis and classification. We used the Savitzky-Golay Smoothing Filter [5] to filter the data from the calculated velocities and remove the noise since it is a technique used on high-frequency data. We filtered out some point velocities that would be considered saccades which are probably just noise. The impact of filtering can be seen in Figure 1.



(a) Original velocities before pre-procesing.



(b) Velocities after filtering using Savitzky-Golay filter with the 3rd polynomial order and window length 11.

Figure 1: Velocities before and after pre-processing.

Post-processing the results

After looking at the results given by the pre-processing and the I-VT algorithm we found out that the pre-processing step was not enough for accurate classification. Therefore, we decided to perform a post-processing on the classification output. Firstly, we trimmed the fixations by excluding all the points outside of 2° radius with respect to the time sequence, the used value was determined according to Nyström, M. & Holmqvist work [5]. The goal of this step is to eliminate smooth pursuits. Secondly, we discarded short fixations which duration was shorter than 50 ms. Finally, those fixations which were closer than $0.5°$ to each other were merged.

RESULTS

The results are shown in Figure 2. The values for both mean saccade amplitudes and fixations duration did not differ too much for the "known" and "unknown" instances, therefore we can conclude that the fact if the subject recognises given image does not have major impact on the fixations duration, nor the saccade amplitudes. The standard error bars overlap a lot in both charts, which implies there was no significant difference between the subjects.

When we closer examine the MSA values, it is apparent that the saccade amplitudes are slightly higher when the subject did not recognise the image. This probably means that the subject need to look around the image more, when it is unknown to them. The difference for known and unknown instances is even smaller in the MFD chart. However, we can still recognise a small trend where the fixations duration is a bit higher for the unknown images. This result was kind of expected, since people usually need more time to analyse an image they are not familiar with.

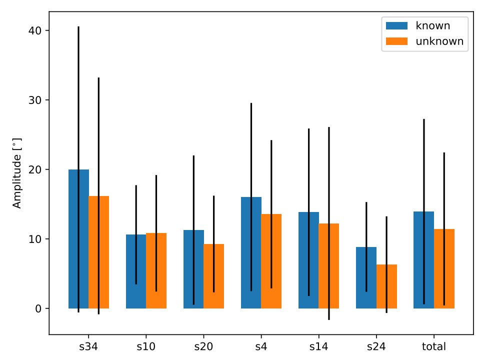
To sum it up, by processing the experimental data, we learned that there are small but evident trends in the way of how people look at known and unknown images. When the image is not known, the subject makes bigger movements around the area and spends more time fixating on the points of interest.

GROUP MEMBERS

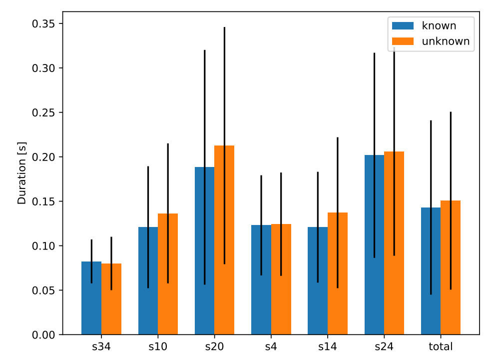
Beforehand starting to work in any type of task for this course, the group aimed to equally divide the load of work for each member. However, we found it was difficult to do so since not everyone lived in the same location. When it comes to some specific tasks like coding or the video recording, it was impossible to achieve a parallel, compatible and successful progress for the tasks that were asked to be accomplished. Members contribution is specified in Table 1.

Contribution	Homeworks	Project
Peter Gazdík	22.5 %	26 %
Roman Oravec	22.5 %	26 %
Silvia Fontfreda Casals	22.5 %	26 %
Umar Bin Qushem	22.5 %	22 %
Essi Kirveskari	10 %	0 %

Table 1: Members contribution.



(a) Mean Saccade Amplitudes (MSA).



(b) Mean Fixation Durations (MFD).

Figure 2: MSA and MFD values calculated for each subject and then aggregated for all subjects.

REFERENCES

[1] Andrew T. Duchowski. 2002. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers* 34, 4 (01 Nov 2002), 455–470. https://doi.org/10.3758/BF03195475

[2] Kenneth Holmqvist. 2017. *Eye tracking: A comprehensive guide to methods, paradigms, and measures.*

[3] Dongheng Li and Derrick J. Parkhurst. 2019. Starburst: A robust algorithm for video-based eye tracking. (05 2019).

[4] Robert Lupu and Florina Ungureanu. 2013. A survey of eye tracking methods and applications. *Buletinul Institutului Politehnic din Iaşi. Secţia Automatică şi Calculatoare* 3 (01 2013).

[5] Marcus Nyström and Kenneth Holmqvist. 2010. An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data. *Behavior Research Methods* 42, 1 (01 Feb 2010), 188–204. https://doi.org/10.3758/BRM.42.1.188

[6] Deborah Riby and Peter J. B. Hancock. 2009. Looking at movies and cartoons: eye-tracking evidence from Williams syndrome and autism. *Journal of Intellectual Disability Research* 53, 2 (2009), 169–181. https://doi.org/10.1111/j.1365-2788.2008.01142.x

APPENDIX

Source code

https://github.com/roman-oravec/eye-tracking

Output CSV file

```
subject_id   MFD_true MFD_SD_true MFD_false MFD_SD_false MSA_true MSA_SD_true
MSA_false MSA_SD_false MFD_overall MFD_overall_SD MSA_overall MSA_overall_SD

s34 0.08238709677419356 0.02471864572362308 0.08003658536585366 0.030034307668410826
19.996207729322794 20.5830271219317 16.184017628180293 17.03739626988288
0.08068141592920353 0.02869345628176178 17.31242589811847 18.242372677045772

s10 0.12082758620689654 0.06858826126844722 0.1363740740740741 0.07873432162608908
10.591077046299505 7.136746892030859 10.798375195773072 8.383530420265577
0.13486622073578597 0.0779441239025403 10.77915549979539 8.276056903897071

s20 0.18822340425531917 0.13201484074014214 0.21258536585365853 0.13336485360126057
11.25755544763716 10.75110182468221 9.260184191366369 6.955669354406164
0.20492642140468226 0.13342218532171884 9.886027184997884 8.384343655189413

s4 0.12292222222222221 0.05634028118451857 0.12423118279569892 0.05811783817086419
16.019399154851122 13.545359159193996 13.541019301047882 10.670797098275209
0.12380434782608696 0.0575475051015702 14.341329462171846 11.734026958363943

s14 0.12082051282051282 0.06232169305609437 0.137125 0.08489928960244603 13.831293891704638
```

12.05218830603043 12.214686161231263 13.883333567577756 0.13446443514644352 0.08186450300524645 12.494728445250272 13.597585658653463

s24 0.20168750000000002 0.11534606557551064 0.20609756097560977 0.11738654197970737 8.838956432411104 6.458087145254052 6.288596710721379 6.957455849794646 0.2053775510204082 0.11706718072186485 6.694335757353835 6.943381751897723