Performance Assessment Framework for Computational Models of Visual Attention

Bharathi Murugaraj(✉) and J. Amudha

Department of Computer Science and Engineering, Amrita University, Bengaluru, India
bharathi02.subra@gmail.com

Abstract. This paper presents performance framework for computational model of visual attention, a software package, written using python scripting language, developed for the real-time comparison of computational model with human fixations. The performance framework was developed for real-time processing of eye trackers recorded data, analyzing them to generate fixation map, and comparing the fixation map to a saliency model got by running a configured computational model either in bottom-up or top-down mode. The framework is designed such that added modules can be extended for various experiment processing as required by the researcher. The framework encompasses the main connection to eye tracker to collect the raw data that will have observers eye coordinates and duration, it has analysis model to analyze the model and providing methods of visualization like fixation, heatmap and scanpath, it also has a computational model that predicts the fixation on the given image stimulus, finally the platform compares the fixation and saliency map to assess the accuracy of the prediction. All the functions of the framework can be controlled by using the graphical user interfaces.

Keywords: Eye tracker · Eye movement analysis · Fixation · Heatmap · Computational model · Saliency map · Python · Visual attention

1 Introduction

Visual attention mechanisms are developed that is derived from the study of human visual systems. Its process enables machine vision systems to select the most relevant regions from a scene Eye tracking is used for usability and psychology testing and used popularly in researches in visual system, cognitive process, and human-computer interaction. In psycho-physiological research, eye-tracking methodology is used to get reaction parameters from eye movement data that used to analyze cognitive processes underlying visual behavior. Researcher's use eye movement data to study cognitive influence in learning, memory and attention. To perform eye tracking experiments many commercial eye-trackers e.g. EyeTribe [6], SMI [9], and open-source solutions e.g. Gaze Tracker [10] are available in market. They provide strong algorithms for gaze tracking, analysis and visualizations. Such tools provide valuable insights into the recorded gaze behavior. On the other hand some researchers have focused on the model-based analysis tools e.g. GazeAlyze [2], The

Psychtoolbox for Matlab [11], PychoPy [12], for gaze reading in experimental studies in vision research. These tools come up with various detecting events in the gaze data, such as algorithms for blink detection, fixation detection etc.

Modeling visual saliency has attracted much interest recently and there are now several frameworks and computational approaches available. Some are inspired by cognitive findings, some are purely computational, and others are in between. Using wide variety of approaches many computational models of visual attention have been developed to see in a free-viewing condition how to predict where people look in images. The objective of these models is to improve artificial vision systems by computing, a numerical value of the likelihood of attending to, or the saliency of every location in an image. The performance of a model is measured by how will it predicts where people look in images in a free-viewing condition. So far, researchers validate prediction of attention models by direct comparison between eye movements recorded from humans watching the stimuli and model output. The fixations from the humans and the saliency map from the models is compared to assess the performance of the models.

To combine the eye tracking device, analysis of gaze data, saliency model and assessment of the model for its performance we developed a performance assessment framework. The framework provides 4 module: experimental module that helps to control experiment with the eye tracking device and record the eye movement data called gaze data of the observer, analysis module which generates a fixation map by analyzing the gaze data, saliency module generates a saliency map from a computational model (bottom-up/top-down), and performance comparator that compares the fixation map and saliency map using 3 popular metrics Normalized Scanpath Saliency (NSS), Pearson's Correlation Coefficient (CC) and Similarity.

2 Motivation

Eye tracking is process of finding out where the user is looking at for the given stimulus, by usage of eye tracking device. All eye tracking system function with a common principle, identifying the same eye features across the multiple images. And the results are correlated to a particular eye. Salient objects generally appear visually different from the other displayed objects in the scene. Eye tracking devices records eye coordinates and duration about a position of an eye within an eye's image as registered by a camera. This raw data is translated into a gaze point. For the computational tasks of recorded data analysis: fixation detection, heatmap and scanpath are considered. The most fundamental translation from raw data is the fixation detection, which is true for one eye tracker and for a specific dataset.

Generally a computational model handles several features and then computes them in parallel. The resulting value is fused in a representation called saliency map. The saliency map is visualized as a grey-scale image. In this map more the brightness of a pixel, it is the most salient region. Where human look in an image is based on two factors; a bottom-up and top-down approach. Bottom-up models are task-free and is stimulus-driven. Thus they do not require to learn, train or tune to open-ended task. They can be used for prediction on any image dataset, the output of it can be verified against experimental data collected from humans. Top-down models are descriptive, task-specific and can be implemented computationally. But with lot of work going on this area the difference between bottom-up and top-down models is diminishing and there are algorithms of computational models that uses both the approaches for prediction. The paper [5] explores if existing object information can be used to for the next object recognition, it attempts to combine top-down and bottom-up model. Saliency maps produced by different algorithms are often evaluated by comparing output fixated image locations appearing in human tracking data. The inherent ambiguity in how saliency and ground truth are represented leads to different choices of metrics for reporting performance. Here, the performance comparator uses saliency metrics, that is, functions that take two inputs representing eye fixations and predictions and then output a number assessing the similarity or dissimilarity between them.

We presents a software performance assessment framework which is developed using python tool for comparing human fixations and the saliency map from the computational model for a given image stimulus. This will help in choosing appropriate models for the given application in hand. It can use various computational algorithm according to choice of the researcher and is an open-source software. Our intent was to developing an open-source, in-line python application that allows the complete management of entire processes of collecting data from eye tracker, post analyzing the collected data, predicting the salient region using either bottom-up or top-down models and finally compare them, that is, the fixation map and the saliency map.

3 Literature Survey

Even though, there are some well performing commercial eye trackers available, they have two disadvantages compared to the open-source solutions: they are either available at very high costs and thus becomes unaffordable for many academic research or clinical studies. They also provide a tightly-coupled environment that it is difficult to customize the way they need it for their application. Mostly all the solutions provide are black-box ensuring that there no access to image processing routines or modules. Many researchers explore different computational model to measure the accuracy of their prediction. The experiments conducted on 39 observers free-viewing 300 images and compare 10 popular recent modules [3]. It explores which models perform poorly and which ones are better. The performance of saliency models is measured using three different metrics ROC, similarity and EMD. Lot of paper explore on different evaluation metrics. Properties of the inputs affect metrics differently: how the ground truth is represented; whether the prediction includes dataset bias; whether the inputs are probabilistic; whether spatial deviation exist between the prediction and ground truth. Knowing how these properties affect metrics, and which properties are most important for a given application can help with metric selection for saliency model evaluation [4]. In this paper we use Normalized Scanpath Saliency (NSS), Pearson's Correlation Coefficient (CC) and similarity for fairest comparison of fixation map and saliency map. An open-source integrated framework like Visual Search Examination Tool (Vishnoo) [1] combines configurable search tasks with gaze tracking capabilities, thus enabling the analysis of both, visual field and visual attention. This offers easily adaptable stimulus presentation, eye-tracking and evaluation of the visual scan path combined in a single platform.

4 System Architecture of the Framework

The frameworks block diagram is shown in Fig. 1.

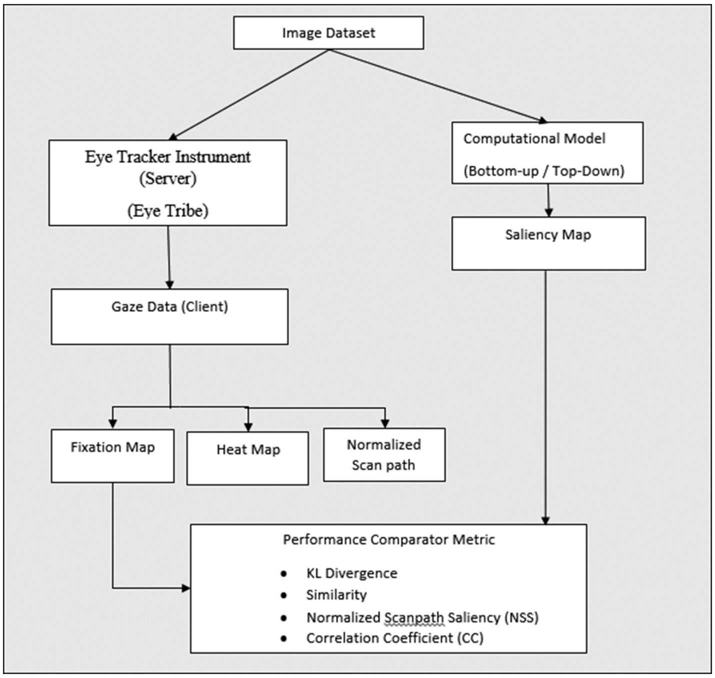


Image Dataset
Eye Tracker Instrument (Server) (Eye Tribe)
Computational Model (Bottom-up / Top-Down)
Saliency Map
Gaze Data (Client)
Fixation Map
Heat Map
Normalized Scan path
Performance Comparator Metric
KL Divergence
Similarity
Normalized Scanpath Saliency (NSS)
Correlation Coefficient (CC)

Fig. 1. Flow chart of performance framework

Researchers can use this framework tool with no specific programming skill, as it is graphically driven and all system parameters and all functionalities are controllable from graphical user interface (GUI). All system parameters of specific experiments are saved in a configuration script. This allows comfortable handling of the analysis of different experiments. The performance framework is entirely developed using python scripting language as it is open-source and can be easily portable. It runs on pretty much everything. All functions are written in python and are separate from the GUI components. The GUI is developed using Java programming language and is used explicitly for adding, updating and deleting configuration parameters. The performance framework consists of Experimental module, Analysis module, Saliency module and Comparator module.

Experimental Module

Eye tribe [11], an eye tracking device is used in the framework. By using the person's face and eyes the device can calculate the exact location as to where the observers is looking at. The gaze coordinates are represented by a pair of (x, y) coordinates that is taken from the screen coordinate system, and it is calculated with respect to the screen the person is observing. Before using the eye tracker device the user need to do calibration process. Because the eye characteristics of individual is different, and this need to be modeled by the eye tracking software in order to estimate gaze accurately. Pygaze [8] an open-source python package is used in the framework. Pygaze acts as a wrapper around several existing eye tracking packages. It is used to create complicated experiments. Pygaze module connects to the eye tribe and records eye movement information like the eye coordination and duration into a tab separated file.

Analysis Module

Further the gaze data is analyzed by the Analyzer module which provides valuable insights into the recorded gaze behavior. In eye tracking data analysis fixation detection is considered. The velocity at each gaze point is calculated when the eye gaze travels from the previous gaze point to the current one. If calculated velocity is smaller than the threshold velocity then that gaze point is tagged as fixation. After completing tagging all the gaze points, consecutive fixation points are segregated into fixation groups <x,y,t,d> where x, y are center coordinates of the fixation group, t is the timestamp of the initial fixation point, and d is the duration. A fixation group is ignored for which the duration threshold is greater than the duration of a group.

Saliency Module

The first computational model of visual attention using the bottom-up approach was given by Koch and Ullman [13]. Here the visual features used were color, intensity or orientation. These feature maps are weighed and summed up to a saliency map.

Performance Module

If fixation map (FMap) and saliency map (SMap) are passed two inputs to a metric function, then the following metric are applied on them for comparison.

Normalized Scanpath Saliency (NSS): Measuring the normalized saliency at the region of fixations

$$NSS(SMap, FMap) = \frac{1}{N} \sum_i \overline{SMap_i} \times FMap_i$$

where $N = \sum_i FMap$ and $\overline{SMap} = \dfrac{SMap - \mu(SMap)}{\sigma(SMap)}$

Here ɩ is the i^{th} pixel, and N is the total fixated pixels in fixation map. A positive NSS score indicates a good correspondence for a fixation located on the model predicted saliency map and the scanpath of the observer's.

Similarity (SIM): Measuring the intersection between distributions

$$SIM(SMap, FMap) = \sum_i \min(SMap_i, FMap_i)$$

where $\sum \boldsymbol{SMap} = \sum \boldsymbol{FMap} = 1$

SIM measures the amount of similarity that exists between two distributions. A SIM of 1 means the FMap distribution and SMap distributions are same and SIM of 0 means there is no overlap.

Pearson's Correlation Coefficient (CC): Evaluating the linear relationship between distributions.

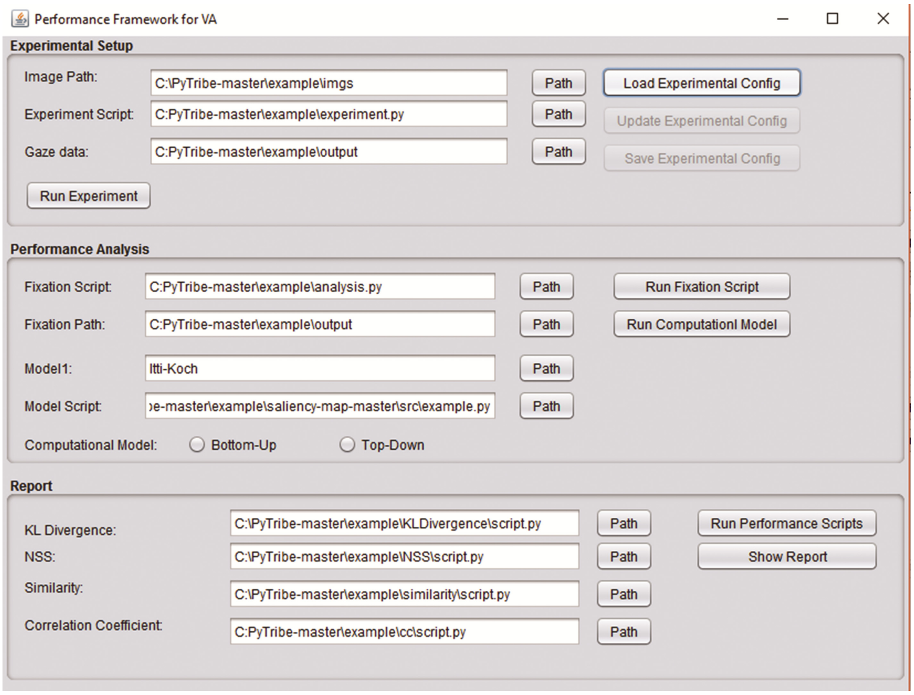
$$CC(SMap, FMap) = \frac{\sigma(SMap, FMap)}{\sigma(SMap) \times \sigma(FMap)}$$

where $\sigma(SMap, FMap)$ is the covariance of saliency map and fixation map. CC can range between -1 to 1. CC is 1 means a perfect correlation, whereas -1 also means perfect correlation but in opposite directions.

5 Implementation

The GUI panel looks like in Fig. 2. All the system parameters are loaded from the configuration script. This can be changed or updated and rewritten to the same script or to a new configuration file.

To conduct the experiment, eye tracking system Eye tribe is used. Pygaze connects to the eye tribe server. It is prerequisite that the eye tribe need to be calibrated separately. Currently it is outside the scope the performance framework. Pygaze starts loading the dataset, that is, the collection of images in the gap of 30 s each. There are 48 images in the traffic sign dataset. They are in PNG format of size 360 × 270, each of the image representing a traffic scene. In the dataset the image is categorized into 3 classes of different traffic sign template, pedestrian crossing, intersection and compulsory for bikes. The traffic sign dataset of 48 images was used to conduct the experiment to collect the eye movements from the participants who were allowed to free view each image for 2 s. Once this data is collected, the ground truth eye fixations from different participants is formatted as a column data and written to a tab separated file. The "rawx" and "rawy" data which represents the gaze coordinates. The Gaze coordinates are the point on screen that the user is currently looking. Gaze coordinates are defined as pixels in a top-left oriented 2D coordinate system and are available in both raw and smoothed forms. There are other information also in the tab separated file like "Tracking state", "Fixation" and "pupil coordinates". But for generating fixation Gaze coordinates values are used. The file content of tab separated file from the pygaze is show below in Fig. 3.



Experimental Setup

Image Path: C:\PyTribe-master\example\imgs

Experiment Script: C:PyTribe-master\example\experiment.py

Gaze data: C:PyTribe-master\example\output

Performance Analysis

Fixation Script: C:PyTribe-master\example\analysis.py

Fixation Path: C:PyTribe-master\example\output

Model1: Itti-Koch

Model Script:)e-master\example\saliency-map-master\src\example.py

Computational Model: ○ Bottom-Up ○ Top-Down

Report

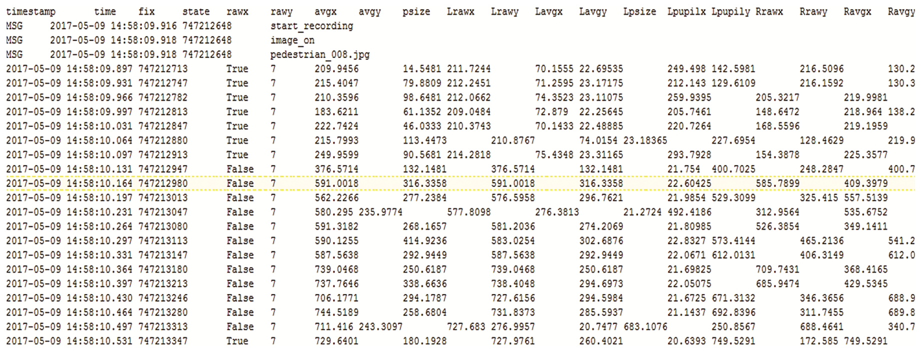
KL Divergence: C:\PyTribe-master\example\KLDivergence\script.py

NSS: C:\PyTribe-master\example\NSS\script.py

Similarity: C:\PyTribe-master\example\similarity\script.py

Correlation Coefficient: C:PyTribe-master\example\cc\script.py

Fig. 2. The GUI of the performance framework



```
timestamp       time    fix     state   rawx    rawy    avgx    avgy    psize   Lrawx   Lrawy   Lavgx   Lavgy   Lpsize  Lpupilx Lpupily Rrawx   Rrawy   Ravgx   Ravgy
MSG     2017-05-09 14:58:09.916 747212648               start_recording
MSG     2017-05-09 14:58:09.918 747212648               image_on
MSG     2017-05-09 14:58:09.918 747212648               pedestrian_008.jpg
2017-05-09 14:58:09.897 747212713               True    7       209.9456        14.5481 211.7244        70.1555 22.69535        249.498 142.5981        216.5096        130.2
2017-05-09 14:58:09.931 747212747               True    7       215.4047        79.8809 212.2451        71.2595 23.17175        212.143 129.6109        216.1592        130.3
2017-05-09 14:58:09.966 747212782               True    7       210.3596        98.6481 212.0662        74.3523 23.11075        259.9395        205.3217        219.9981
2017-05-09 14:58:09.997 747212813               True    7       183.6211        61.1352 209.0484        72.879  22.25645        205.7461        148.6472        218.964 138.2
2017-05-09 14:58:10.031 747212847               True    7       222.7424        46.0333 210.3743        70.1433 22.48885        220.7264        168.5596        219.1959
2017-05-09 14:58:10.064 747212880               True    7       215.7993        113.4473        210.8767        74.0154 23.18365        227.6954        128.4629        219.9
2017-05-09 14:58:10.097 747212913               True    7       249.9599        90.5681 214.2818        75.4348 23.31165        293.7928        154.3878        225.3577
2017-05-09 14:58:10.131 747212947               False   7       376.5714        132.1481        376.5714        132.1481        21.754  400.7025        248.2847        400.7
2017-05-09 14:58:10.164 747212980               False   7       591.0018        316.3358        591.0018        316.3358        22.60425        585.7899        409.3979
2017-05-09 14:58:10.197 747213013               False   7       562.2266        277.2384        576.5958        296.7621        21.9854 529.3099        325.415 557.5139
2017-05-09 14:58:10.231 747213047               False   7       580.295 235.9774        577.8098        276.3813        21.2724 492.4186        312.9564        535.6752
2017-05-09 14:58:10.264 747213080               False   7       591.3182        268.1657        581.2036        274.2069        21.80985        526.3854        349.1411
2017-05-09 14:58:10.297 747213113               False   7       590.1255        414.9236        583.0254        302.6876        22.8327 573.4144        465.2136        541.2
2017-05-09 14:58:10.331 747213147               False   7       587.5638        292.9449        587.5638        292.9449        22.0671 612.0131        406.3149        612.0
2017-05-09 14:58:10.364 747213180               False   7       739.0468        250.6187        739.0468        250.6187        21.69825        709.7431        368.4165
2017-05-09 14:58:10.397 747213213               False   7       737.7646        338.6636        738.4048        294.6973        22.05075        685.9474        429.5345
2017-05-09 14:58:10.430 747213246               False   7       706.1771        294.1787        727.6156        294.5984        21.6725 671.3132        346.3656        688.9
2017-05-09 14:58:10.464 747213280               False   7       744.5189        258.6804        731.8373        285.5937        21.1437 692.8396        311.7455        689.8
2017-05-09 14:58:10.497 747213313               False   7       711.416 243.3097        727.683 276.9957        20.7477 683.1076        250.8567        688.4641        340.7
2017-05-09 14:58:10.531 747213347               True    7       729.6401        180.1928        727.9761        260.4021        20.6393 749.5291        172.585 749.5291
```

Fig. 3. Tab separated file generated from the pygze module.

Using the gaze coordinate and the duration value from the file is read and processed to generate fixation location and fixation map as shown in Fig. 4.

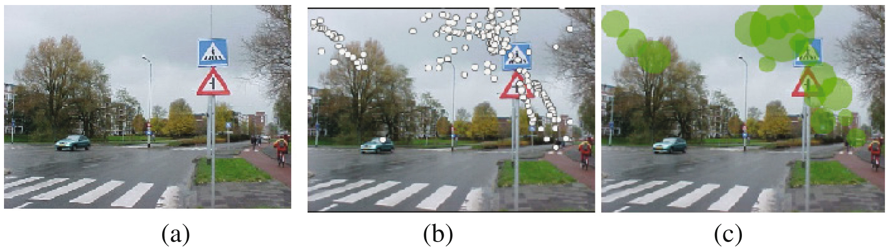


Fig. 4. (a) Traffic sign sample image from the dataset with its (b) fixation location and (c) fixation map

In implementation, free viewing task is considered since this makes it easier to use saliency models with fewest assumptions of the parameters. Two computational models of visual attention was used. Firstly basic Itti and Koch model based was used. The fixation location and fixation map as predicted by the model is displayed in Fig. 5.

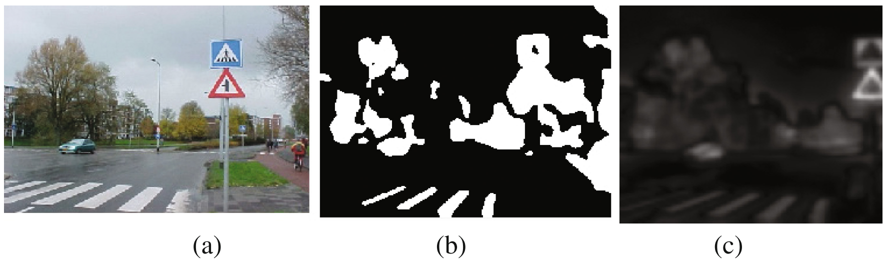


Fig. 5. (a) Traffic sign sample image from the dataset with its (b) binary image of salient regions and (c) its model saliency map using Itti-Koch

Graph Based Visual Saliency (GBVS) [16] was the second model used against the same image dataset. The saliency map is shown in Fig. 6.

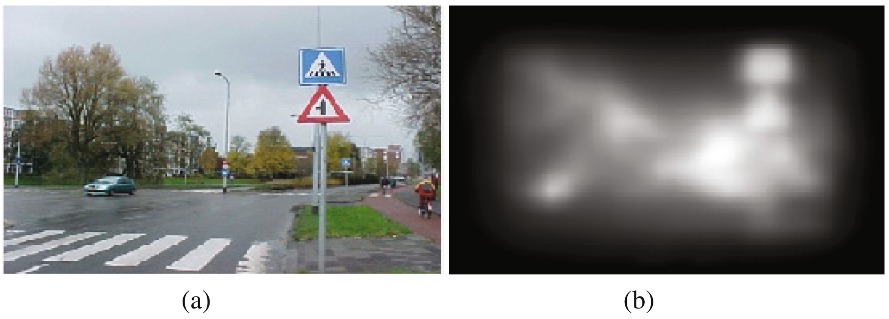
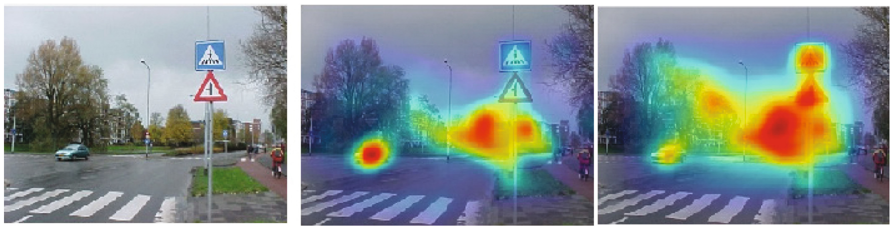


Fig. 6. The GBVS model output (a) original image (b) saliency map

By overlaying the resulting salient regions from Itti-koch and GBVS, the original image is as shown in Fig. 7.



(a) (b) (c)

Fig. 7. Salient region overlayed on the original image (a) original image (b) Itti-koch (c) GBVS

The saliency model is assessed for validation using three different metric by the perfComparator: Similarity, Normalized Scanpath Saliency (NSS) and Correlation Coefficient (CC) as displayed in Fig. 8 above. After running the pygaze for the eye tracker experiment, and pygaze analyzer to generate fixation map, a comparison report is generated. This will compare between the fixation map and saliency map as show in the performance framework UI. The metric result is also displayed in the "Performance Metrics" UI table in Fig. 9. This gives a direct picture to the researcher as to how well the model is able to predict in compare to the ground truth.

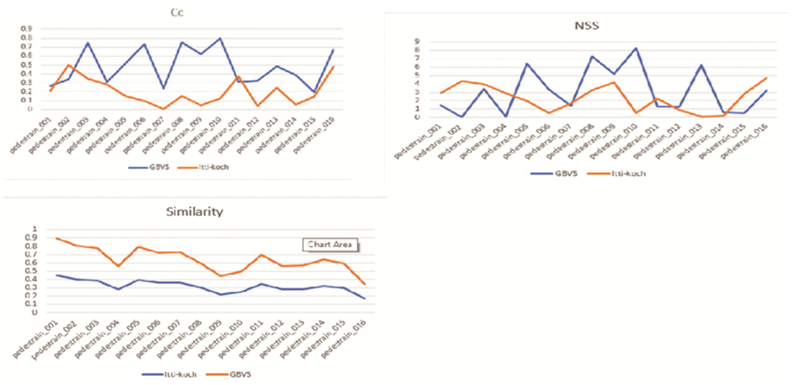
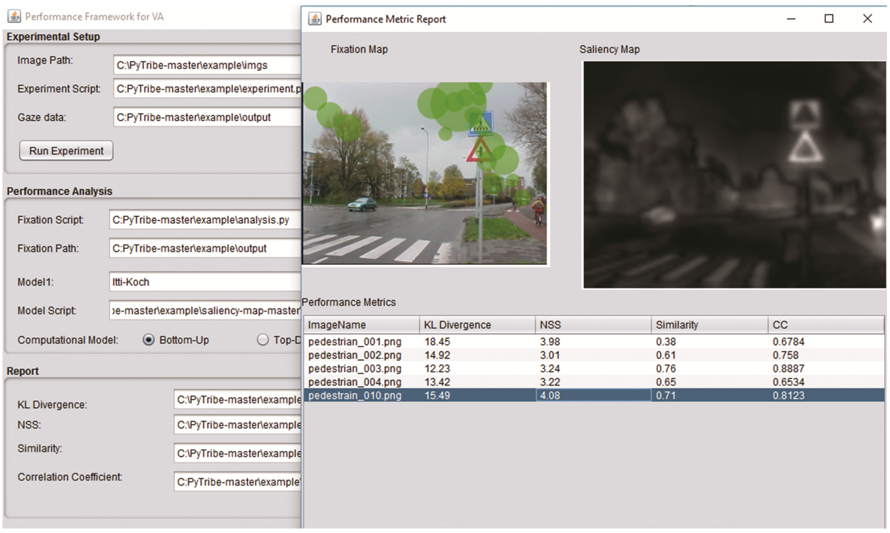


Fig. 8. Performance measure of Itti-koch and GBVS model



Experimental Setup

Image Path: C:\PyTribe-master\example\imgs

Experiment Script: C:PyTribe-master\example\experiment.p

Gaze data: C:PyTribe-master\example\output

Performance Analysis

Fixation Script: C:PyTribe-master\example\analysis.py

Fixation Path: C:PyTribe-master\example\output

Model1: Itti-Koch

Model Script:)e-master\example\saliency-map-master

Computational Model: [✓ Bottom-Up] Top-D

Report

KL Divergence: C:\PyTribe-master\example

NSS: C:\PyTribe-master\example

Similarity: C:\PyTribe-master\example

Correlation Coefficient: C:PyTribe-master\example'

Performance Metrics

ImageName	KL Divergence	NSS	Similarity	CC
pedestrian_001.png	18.45	3.98	0.38	0.6784
pedestrian_002.png	14.92	3.01	0.61	0.758
pedestrian_003.png	12.23	3.24	0.76	0.8887
pedestrian_004.png	13.42	3.22	0.65	0.6534
pedestrain_010.png	15.49	4.08	0.71	0.8123

Fig. 9. The performance metric display comparing the fixation map and saliency map

6 Experiments

From the conducted experimental analysis we see that GBVS model performs better than the Itti-kotch. The Linear Correlation Coefficient (CC) values for the images are closer to 1, that is to say, there is a linear relationship exists between fixation map and saliency map. The similarity score is also approximating to 1, which tells us that the two maps distributions are same. This might be due to the factor that GBVS model also considers is center surround along with color, intensity and orientation in computing saliency map.

7 Conclusion

The performance framework provides a new platform for the researchers to experiment with wide range of performance assessment of computational models, and as well by collecting the human fixations from the eye tracking system module available in the framework. In Vishnoo top-down model used for specific task programming is assessed against scanpath i.e., focus of study is more on gaze data analysis, while the performance framework allows to compare the saliency map and fixation map using 3 popular metrics. Since all components required to test the performance of computational models is available as a combined solution in a single framework, which makes it fastest and flexible solution for researchers. Since the framework can be easily configurable, it becomes an attractive tool to many scientific research.

References

1. Tafaj, E., Kubler, T.C., Peter, J., Rosenstiel, W., Bogdan, M.: Vishnoo-an open-source software for vision research. In: 2011 24th International Symposium on Computer-Based Medical Systems (CBMS) (2011)
2. Berger, C., Winkeles, M., Lischke, A., Hoppner, J.: GazeAlyze: a MATLAB toolbox for the analysis of eye movement data. Behav. Res. Methods **44**, 404–419 (2012)
3. Judd, T., Durand, F., Torralba, A.: A benchmark of computational models of saliency to predict human fixations. MIT Technical report (2012)
4. Riche, N., Duvinage, M., Mancas, M., Gosseling, B., Dutoit, T.: Saliency and human fixations: state-of-the-art and study of comparison metrics. In: 2013 IEEE International Conference on Computer Vision (2013)
5. Amudha, J.: Performance evaluation of bottom-up and top-down approaches in computational visual attention system, Coimbatore (2012)
6. THEEYETRIBE: http://www.theeyetribe.com
7. Radha, D., Amudha, J., Jyotsna, C.: Study of measuring dissimilarity between nodes to optimize the saliency map. Int. J. Comput. Technol. Appl. **5**(3), 993–1000 (2014)
8. Dalmaijer, E.S., Mathot, S., Van der Stigchel, S.: PyGaze: an open-source, cross-platform toolbox for minimal-effort programming of eyetracking experiments. Behav. Res. Methods **46**, 913–921 (2014)
9. SensoMotoric Instrument GmbH (2011). http://www.smivision.com
10. http://gazegroup.org/downloads/23-gazetracker/
11. Brainard, D.H.: The psychophysics toolbox. Spat. Vis. **10**(4), 433–436 (1997)
12. Peirce, J.W.: PsychoPy-psychophysics software in Python. J. Neurosci. Methods **162**(1–2), 8–13 (2007)
13. Koch, C., Ullman, S.: Shifts in selective visual attention: towards the underlying neural circuitry. Hum. Neurobiol. **4**, 219–227 (1985)
14. Bylinskii, Z., Judd, T., Borji, A., Itti, L., Durand, F., Oliva, A., Torralba, A. MIT saliency benchmark. http://saliency.mit.edu/
15. Amudha, J., Radha, D., Deepa, A.S.: Comparative study of visual attention models with human eye gaze in remote sensing images. In: Proceedings of the Third International Symposium on Women in Computing and Informatics (2015)
16. Harel, J., Koch, C., Perona, P.: Graph-based visual saliency. In: Proceedings of Advances in Neural Information Processing Systems (NIPS), vol. 19, pp. 545–552 (2007)