Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Testing the accuracy of the neural network based eye gaze estimation using an eye tracker

Bachelor Thesis

2023
Matej Budoš

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Testing the accuracy of the neural network based eye gaze estimation using an eye tracker

Bachelor Thesis

Study Programme:	Informatics
Field of Study:	Applied Informatics
Department:	Department of Computer Science
Supervisor:	prof. Ing. Igor Farkaš, Dr.

Bratislava, 2023
Matej Budoš

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Matej Budoš
Študijný program: aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Testing the accuracy of the neural network based eye gaze estimation using an eye tracker
Testovanie presnosti odhadu očného pohľadu založeného na neurónovej sieti pomocou sledovača očí

Anotácia: Plynulosť interakcie medzi človekom a robotom výrazne závisí od schopnosti robota sledovať pohľad očí človeka, ktorý je dobrým indikátorom jeho úmyslov. Takýto systém môže využiť priamo vstupy z kamier v očiach robota, ktoré snímajú scénu a vedia vyextrahovať hlavu a oči človeka sediaceho pred robotom.

Cieľ: 1. Nainštalujte a otestujte fungujúci systém (na báze neurónovej siete) na odhad pozície pohľadu človeka (Herashchenko, 2023) u polohumaniodného robota NICO.
2. Pripravte merač pohľadu očí ktorý bude merať súradnice pohľadu participantov počas experimentu.
3. S využitím dotykovej obrazovky umiestnenej horizontlálne na stole medzi robotm a participantom navrhnite, realizujte a vyhodnoťte experiment, zameraný na testovanie presnosti predikovaného smeru pohľadu na obrazovku.

Literatúra: Admoni H., Scassellati B. (2017) Social Eye Gaze in Human-Robot Interaction: A Review. Journal of Human-Robot Interaction, 25–63. https://doi 10.5898/JHRI.6.1.Admoni
Herashchenko D. (2023). Robot reading human head pose and gaze direction. Bakalárska práca. FMFI UK Bratislava

Vedúci: prof. Ing. Igor Farkaš, Dr.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: doc. RNDr. Tatiana Jajcayová, PhD.

Dátum zadania: 12.10.2023

Dátum schválenia: 16.10.2023 doc. RNDr. Damas Gruska, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Comenius University Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Matej Budoš
Study programme:	Applied Computer Science (Single degree study, bachelor I. deg., full time form)
Field of Study:	Computer Science
Type of Thesis:	Bachelor´s thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: Testing the accuracy of the neural network based eye gaze estimation using an eye tracker

Annotation: The smoothness of the human-robot interaction greatly depends on the robot's ability to follow the human eye gaze, which is a good indicator of human intentions. Such a system can take an input directly from the cameras in the robot's eyes, which capture the scene and can extract the head and eyes of a person sitting in front of the robot.

Aim:
1. Install and test a working system (based on neural network) for estimation of human gaze position (Herashchenko, 2023) for semi-humanoid robot NICO.
2. Prepare an eye tracker that will measure the participants' gaze coordinates during the experiment.
3. Using a touchscreen placed horizontally on the table between the robot and the participant, design, implement and evaluate an experiment aimed at testing the accuracy of the predicted direction of looking at the screen.

Literature:
Herashchenko D. (2023). Robot reading human head pose and gaze direction. Bachelor's thesis. FMFI UK Bratislava
Admoni H., Scassellati B. (2017) Social Eye Gaze in Human-Robot Interaction: A Review. Journal of Human-Robot Interaction, 25–63. https://doi 10.5898/JHRI.6.1.Admoni

Supervisor:	prof. Ing. Igor Farkaš, Dr.
Department:	FMFI.KAI - Department of Applied Informatics
Head of department:	doc. RNDr. Tatiana Jajcayová, PhD.
Assigned:	12.10.2023
Approved:	16.10.2023

doc. RNDr. Damas Gruska, PhD.
Guarantor of Study Programme

.....
Student

.....
Supervisor

Acknowledgments: Tu môžete poďakovať školiteľovi, prípadne ďalším osobám, ktoré vám s prácou nejako pomohli, poradili, poskytli dáta a podobne.

Abstrakt

Slovenský abstrakt v rozsahu 100-500 slov, jeden odstavec. Abstrakt stručne sumarizuje výsledky práce. Mal by byť pochopiteľný pre bežného informatika. Nemal by teda využívať skratky, termíny alebo označenie zavedené v práci, okrem tých, ktoré sú všeobecne známe.

Kľúčové slová: jedno, druhé, tretie (prípadne štvrté, piate)

Abstract

Abstract in the English language (translation of the abstract in the Slovak language).

Keywords:

Contents

List of Figures

Chapter 1

Theoretical Background

Eye tracking technology has emerged as a powerful tool in cognitive science and human-robot interaction. By precisely monitoring eye movements, it provides insights into attentional processes, decision-making, and cognitive workload. In cognitive science, eye tracking helps researchers understand how humans perceive and process information, while in human-robot interaction, it informs the design of intuitive interfaces and socially engaging robots.

This thesis builds upon the work of Dmytro Herashchenko, who developed a neural network model that predicts gaze direction in form of pitch and yaw angles, without using any additional hardware apart from RGB web camera. This research aims to explore different gaze mapping techniques, their implementations and comparison in terms of accuracy and robustness to find the most suitable method for eye tracking on screen using this model. To objectively assess the accuracy of different mapping approaches, we will be comparing the predicted gaze data from our models with the ground truth data on the screen, we can then visualize the strengths and weaknesses of each mapping technique using error heat maps. Following the evaluation, we will integrate the most accurate mapping technique in an human-robot interaction experiment featuring robot Nico and thus test the real-world applicability of our model. This final stage will determine whether this approach of eye tracking is suitable for further experiments and development.

1.1 Regression

Regression is a method used to find correlation between independent and dependent variable. This relationship is expressed through mathematical function which represents line or curve that describes this relationship. To find to best fitting function, the least squares method is used. This method evaluates the sum of the squared vertical distances between the line and the points and tries to minimize it.

1.1.1 Linear regression

Linear regression, being the simplest and most direct method, finds correlation between a target and one or more independent variables by simply fitting a line to the data. The main advantage is it's simplicity and computational efficiency. However, it is very sensitive to outliers and can easily overfit on smaller dataset.

One way to prevent overfitting on smaller data is to use ridge regression. By introducing a small bias to how line is fit to the data. We can get predictions with significantly less variance long term.

1.1.2 Polynomial regression

This method is an extension of linear regression where instead of fitting line to the data we fit nth degree polynomial. This allows capturing non-linear relationships between variables while being able to fit wider range of data patterns. However, high degrees of polynomial can be prone to overfitting.

1.1.3 Outliers

Outliers or anomalies are data points that deviate significantly from the overall pattern observed in the data. These points can significantly influence the result of a regression analysis especially on smaller datasets. There are many techniques that remove outlier points but for this paper we will be using Isolation Forest technique from library scikit-learn[10]. This method allows us to locate different regions - forests of points, in our case groupings of yaw and pitch angles that correspond to certain coordinates. We can then remove outliers within that group of points to obtain subset of the dataset containing more accurate data without anomalies as shown in 1.1.

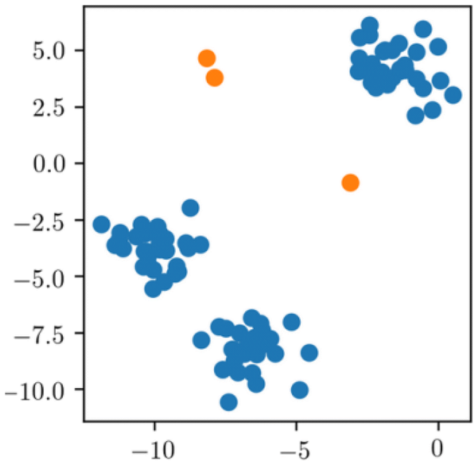

Figure 1.1: Outlier detection [11]

1.2 Gaze mapping methods

1.2.1 2D regression

Regression is one of the most widely used technique when it comes to mapping gaze to screen or world coordinates in general. In context of screen mapping, various gaze locations and their corresponding ground truth points on screen are collected in order to fit a function that predicts screen coordinates from gaze direction. The choice of regression type mainly depends on the complexity of relationship between dependent and independent variable.

1.2.2 3D geometric model

This method is generally used in head mounted headsets, where infrared cameras capture eye images and map various positions of pupil to 3D model of the eyeballs. One of the examples of mapping this 3D model to screen coordinates can be found in this paper [8]. By adding two RGB cameras to capture both eyes, they acquired corneal image of the scene which could be then mapped to world image from front camera of the headset. Gaze point of the user was found by image matching between the corneal image and the world camera image using a mapping transformation between the cameras.

1.2.3 Interpolation-based

It is a very simple method which does not require any additional hardware. It requires a simple calibration choreography where user records their gaze direction for the outermost bounds of planar surface (monitor) in four directions. Then by interpolating between maximum and minimum yaw and pitch we are able to find corresponding x and y coordinate respectively. However, this method does not take into account various factors like eye physiology, and the geometric relationship between eye, screen and camera which makes this model extremely sensitive to head movement.[5]

1.3 Herashchenko's model

This appearance-based model is trained on combination of two datasets. The Columbia gaze dataset [12] which consists of real-world data and synthetic Metahuman dataset [4] created by Herashchenko. This addition expanded overall training data and managed to improve the accuracy on test data as shown in figure 1.8.

The whole process of eye tracking starts with preprocessing the image in form of face detection using RetinaFace library. This model allows us to locate face landmarks

and thus detect both eyes in image in order to crop them as an input into the network. Another input, although used only in last layer, are head positions produced by another pretrained model called SixDrepNet[2].

The model itself operates in a multiple-step process. First, it takes cropped images of the user's eyes which are then fed into a convolutional neural network which is responsible for learning various features of the eyes, most likely, pupil positions and eye contours. In the final layer of the network, the head position information is incorporated. This combined approach allows the model to take into account both the visual appearance of the eyes and the user's head orientation. Finally, the model outputs the pitch and yaw angles of the user's gaze, essentially pinpointing the direction where they are looking.

Trained on \ Tested on	Metahuman Dataset	Columbia Gaze Dataset	Combined Dataset
Metahuman Dataset	MAE ≈ 0.59 σ ≈ 0.52	MAE ≈ 8.12 σ ≈ 5.81	MAE ≈ 1.9 σ ≈ 3.79
Columbia Gaze Dataset	MAE ≈ 8.42 σ ≈ 4.83	MAE ≈ 1.93 σ ≈ 1.5	MAE ≈ 7.55 σ ≈ 5.17
Combined Dataset	MAE ≈ 0.65 σ ≈ 0.56	MAE ≈ 2.88 σ ≈ 1.94	MAE ≈ 1.04 σ ≈ 1.25

Figure 1.2: Model evaluation [3]

1.3.1 Issues with the model

Despite its promising performance on test dataset containing both real-world and synthetic data, Herashchenko's model indicates a notable lack of robustness. This model demonstrates inconsistency when presented with the same images. Images shows in figure1.3 are nine consecutive frames from camera in great light conditions and eye visibility. Despite this, the model often yields different yaw and pitch values ranging between [-10.51, -6.49] and [11.58, 14.7] respectively.

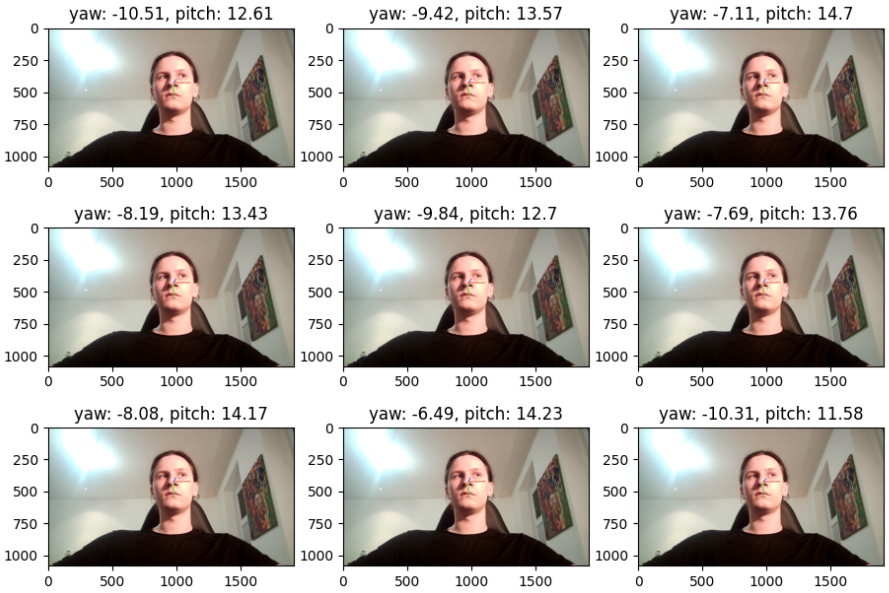

Figure 1.3: Yaw and Pitch values for 9 same images

Moreover, its reliability diminishes when the eyes undergo slight movements, suggesting that its accuracy is compromised in scenarios where precision is required. In a test where user sits 50cm from the 27" screen, nine points are shown successively and for each point eye locations are collected. These eye data should form nine different groups, each corresponding to one point on the screen as in 1.5. Despite there being some signs of groups, many of them tend to blend together as shown1.4.

These inconsistencies raise concerns about the model's reliability in practical applications, highlighting the need for further refinement to enhance its robustness and accuracy.

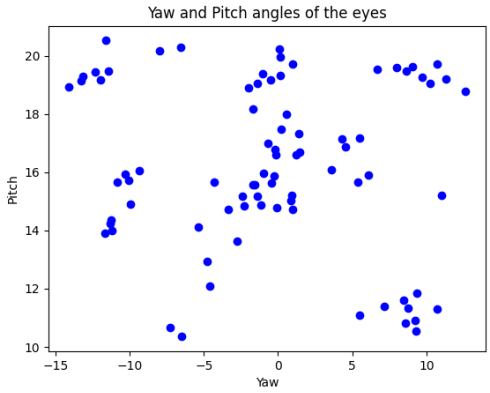

Figure 1.4: Model's values

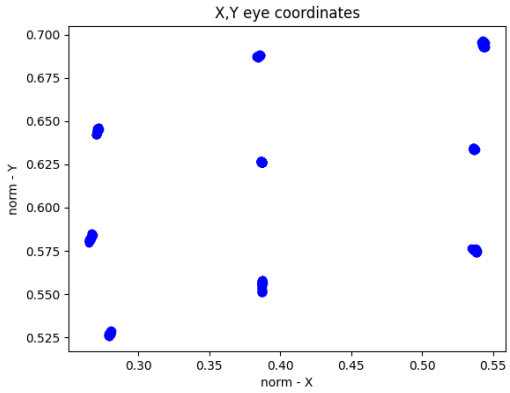

Figure 1.5: Pupil Core values

1.4 WebGazer

WebGazer is an eye tracking library written in JavaScript that uses RGB web cameras to infer the eye-gaze locations in real time. Intead of mapping location of the pupil they represent each eye as a 6 X 10 image patch to capture more characteristics of the eyes. The thing that sets this system apart from other appearance based systems is it's self-calibration. Research [1] shows that there is correlation between cursor and gaze during web browsing, especially in the regions of the page where user is focused. It also suggests this relationship can be used for gaze mapping on screen coordinates. WebGazer utilizes this facts and trains various [9]

1.5 Pupil core

Pupil Core by Pupil Labs is a head-mounted eye tracking device, equipped with two IR cameras 1.6 to track movement of both eyes and world camera to capture user's field of view. This device uses the dark pupil technique and 3D model to accurately track eye movements in three dimensions. The dark pupil technique is essentially capturing eye movements using IR camera, creating strong contrast between pupil and iris. It also allows to precisely track pupil dilatation. Tracking using 3D model 1.7, although being less accurate than tracking using 2D gaze positions, it retains it's accuracy during subtle head movements and device slippage.

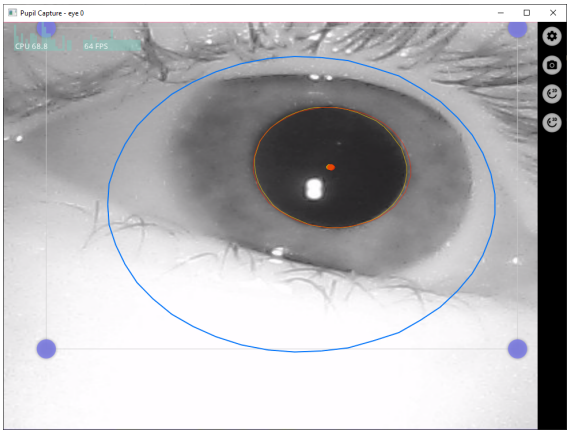

Figure 1.6: IR eye camera

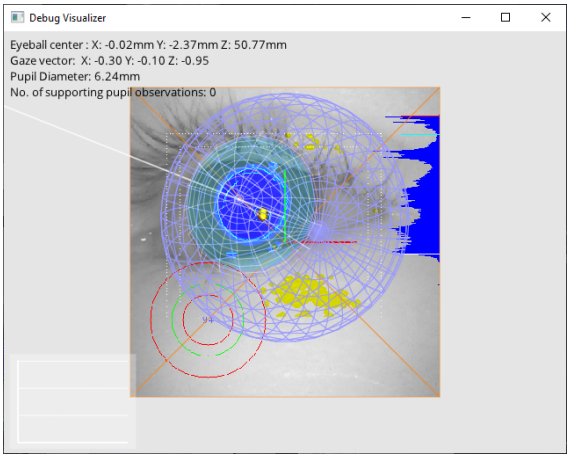

Figure 1.7: 3D eye model

1.5.1 Coordinate system

In Pupil Core, various coordinate systems play crucial roles in understanding and processing visual data. There are three main coordinate systems associated with each camera: 2D Image Space, 2D Normalized Space, and 3D Camera Space.

The 2D Image Space is defined within the captured image itself, with its origin at the top left corner and measured in pixels. It includes lens distortion and has boundaries corresponding to the width and height of the image.

The 2D Normalized Space is similar to 2D Image Space but is normalized to unit dimensions, ranging from 0 to 1 for both x and y coordinates. It still incorporates lens distortion effects and maintains the same boundaries as the image space. The main difference is the origin of the coordinates starts at bottom left corner.

Lastly, the 3D Camera Space is centered at the camera itself and with no lens distortion effects considered. It lacks predefined boundaries and utilizes a Cartesian coordinate system with x, y, and z axes. Notably, the eye model, which shares this space, describes the orientation of the eye relative to the camera. Cartesian coordinates describe the eye's position, while spherical coordinates phi and theta, provide angular information to express gaze direction. [7]

1.5.2 Pupil-core-network-client

This Python module serves as a client interface for the Pupil Core Network API, designed to facilitate interactions with Pupil Core. The module offers a range of features, including initiating and stopping recordings, retrieving version information, managing plugins like Annotation Plugin and Eye Process, estimating clock offset between client and Pupil time, and streaming video data to and from Pupil Capture. It provides a convenient way for developers and researchers to integrate Pupil Core functionalities into their Python applications or scripts.

1.5.3 Screen calibration

One of many calibration methods that Pupil Labs provide is screen calibration. Five points (in each corner and center of screen) are displayed in short sequence, where for each point multiple corresponding eye coordinates are collected . The data is then filtered by confidence of pupil detection. Finally, the system fits the data with a polynomial regression model and evaluates its accuracy. If needed, the process iterates, removing outliers and refitting the model to a cleaner data subset until a satisfactory accuracy is achieved.[7]

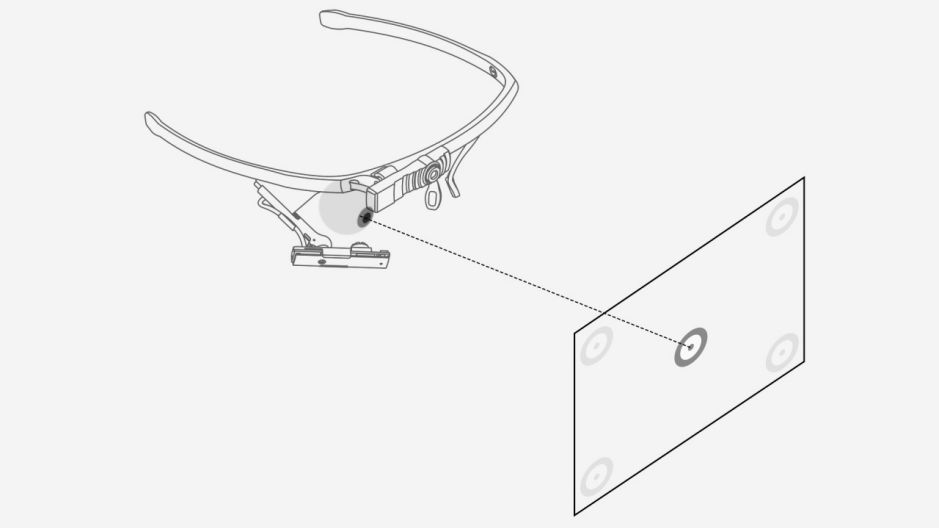

Figure 1.8: Screen calibration [6]

Bibliography

[1] Monchu Chen, John Anderson, and Myeong Sohn. What can a mouse cursor tell us more? correlation of eye/mouse movements on web browsing. *Proceedings of CHI Extended Abstracts*, pages 281–282, 03 2001.

[2] Thorsten Hempel, Ahmed A. Abdelrahman, and Ayoub Al-Hamadi. 6d rotation representation for unconstrained head pose estimation. In *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, October 2022.

[3] Dmytro Herashchenko. Robot reading human head pose and gaze direction. Technical report, 2023.

[4] MetaHuman high-fidelity digital humans made easy. `https://www.unrealengine.com/en-US/metahuman`. [Online; accessed 15-4-2024].

[5] Jia-Bin Huang, Qin Cai, Zicheng Liu, Narendra Ahuja, and Zhengyou Zhang. Towards accurate and robust cross-ratio based gaze trackers through learning from simulation. pages 75–82, 03 2014.

[6] Moritz Kassner, William Patera, and Andreas Bulling. Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction. April 2014.

[7] Pupil Labs. Pupil labs github.

[8] Moayad Mokatren, Tsvi Kuflik, and Ilan Shimshoni. 3d gaze estimation using rgb-ir cameras. *Sensors*, 23:381, 12 2022.

[9] Alexandra Papoutsaki, Patsorn Sangkloy, James Laskey, Nediyana Daskalova, Jeff Huang, and James Hays. Webgazer: Scalable webcam eye tracking using user interactions. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3839–3845. AAAI, 2016.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[11] Sandosh .S, V. Govindasamy, and Akila Gopu. Enhanced intrusion detection system via agent clustering and classification based on outlier detection. *Peer-to-Peer Networking and Applications*, 13, 05 2020.

[12] B.A. Smith, Q. Yin, S.K. Feiner, and S.K. Nayar. Gaze Locking: Passive Eye Contact Detection for Human?Object Interaction. In *ACM Symposium on User Interface Software and Technology (UIST)*, pages 271–280, Oct 2013.