Power-efficient and shift-robust eye-tracking sensor for portable VR headsets

Dmytro Katrychuk
Department of Computer Science,
Texas State University
San Marcos, Texas
d_k139@txstate.edu

Henry K. Griffith
Department of Computer Science,
Texas State University
San Marcos, Texas
h_g169@txstate.edu

Oleg V. Komogortsev
Department of Computer Science,
Texas State University
San Marcos, Texas
ok@txstate.edu

ABSTRACT

Photosensor oculography (PSOG) is a promising solution for reducing the computational requirements of eye tracking sensors in wireless virtual and augmented reality platforms. This paper proposes a novel machine learning-based solution for addressing the known performance degradation of PSOG devices in the presence of sensor shifts. Namely, we introduce a convolutional neural network model capable of providing shift-robust end-to-end gaze estimates from the PSOG array output. Moreover, we propose a transfer-learning strategy for reducing model training time. Using a simulated workflow with improved realism, we show that the proposed convolutional model offers improved accuracy over a previously considered multilayer perceptron approach. In addition, we demonstrate that the transfer of initialization weights from pre-trained models can substantially reduce training time for new users. In the end, we provide the discussion regarding the design trade-offs between accuracy, training time, and power consumption among the considered models.

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; • **Hardware** → *Power estimation and optimization*; *Simulation and emulation*.

KEYWORDS

eye-tracking, virtual reality, VR, photo-sensor oculography, PSOG, machine learning, ML

ACM Reference Format:
Dmytro Katrychuk, Henry K. Griffith, and Oleg V. Komogortsev. 2019. Power-efficient and shift-robust eye-tracking sensor for portable VR headsets. In *2019 Symposium on Eye Tracking Research and Applications (ETRA '19), June 25–28, 2019, Denver , CO, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3314111.3319821

1 INTRODUCTION

Evidenced by increasing adoption within commercial headsets, eye-tracking (ET) technology offers notable potential for improving the user-experience in virtual and augmented reality (VR/AR) environments. The visual system is one of our main tools for interacting with the surrounding environment, so ET empowers a natural dimension of human computer interaction (HCI). This capability is crucial to provide accessibility for disabled individuals who are unable to use traditional input modalities. Real-time ET may also be used to deploy foveated rendering which produces high-quality content only near the point of gaze [Guenter et al. 2012]. This strategy can decrease power consumption or preserve the same amount of it but improve the rendering quality.

The specific traits of each subject's oculomotor system, eye shape, periocular features etc., were proved to have great potential for biometrics and health assessment applications. In the lab environment, it is possible to build an eye movement-based biometrics system with an equal-error rate (EER) of less than 3% [Friedman et al. 2017]. The main advantage of such a system is that there is no known way to spoof it, which is not the case for a fingerprint sensor or facial recognition system. With respect to health assessment, the detection of various diseases and states, including schizophrenia, mild traumatic brain injury, intoxication and fatigue, have been demonstrated using eye movement signals.

ET sensors in current VR headsets utilize video-based oculography (VOG) technology. VOG illuminates the eye using a set of infrared (IR) emitters, captures reflections using an IR camera, and estimates gaze location using image processing techniques. VOG sensors offer a robust, non-invasive ET solution, and are integrated within the majority of stand-alone ET devices. High-quality VOG systems for laboratory use may be engineered to achieve tremendous performance, including sampling rates up to $2000Hz$, along with spatial accuracy and precision scores of less than $0.5°$ and $0.05°$ of the visual angle, respectively (EyeLink 1000 Plus, [Research 2016]). VOG sensors within current VR headsets, such as FOVE VR and HTC Vive with third-party add-ons from SMI or PupilLabs, are restricted in sampling rate to $250Hz$, and have less than $1.0°$ spatial accuracy. Moreover, requirements of form factor and available processing power, prohibits using VOG in wireless headsets like Google Daydream or Samsung Gear VR.

Photosensor oculography (PSOG) [Torok et al. 1951] offers a promising solution for addressing the aforementioned limitations of VOG devices. By capturing the IR reflection from the eye using only a sparse grid of IR detectors, PSOG dramatically reduces computational requirements and promotes increased sampling rate. Unfortunately, the performance sensitivity of PSOG devices to sensors shift poses new challenges. As mentioned in [Rigas et al. 2017], the shift of more than $0.5mm$ result in more than $1.0°$ spatial accuracy degradation. Usual VR applications, such as gaming, social

ETRA '19, June 25–28, 2019, Denver , CO, USA

ACM ISBN 978-1-4503-6709-7/19/06…$15.00
https://doi.org/10.1145/3314111.3319821

interaction or learning in a simulated environment, will not be possible without changing facial expressions, repositioning the head, or moving freely. During any of these activities, slight headset shifts are likely to occur, so viable ET solutions must exhibit robustness to sensors shift.

This study advances the development of shift-robust PSOG ET sensors with accuracy on par with existing VOG solutions for wired VR headsets. We expand the literature by evaluation of a convolutional-neural-network (CNN) utilized together with transfer learning to build a calibration mapping in the presence of simulated sensor shifts. Moreover, our work improves upon previous studies based on synthetic data by using actual eye images of multiple subjects obtained from a custom eye-tracker.

2 PRIOR WORK

Eye tracking sensors for wireless VR headsets must exhibit sufficient sampling frequency, spatial accuracy, etc., to support emerging applications, while minimizing power consumption. Although alternative sensing modalities have recently been suggested for use in wireless VR headsets, such as a smartphone camera sensor [Ahuja et al. 2018], PSOG is especially promising due to its ability to provide quality tracking at high sampling frequencies. Traditional PSOG systems illuminate the eye using IR emitters, and then capture reflections using an array of IR receiving sensors. Li et.al demonstrated a prototype which eliminates the emitter requirement by utilizing the screen light, has $10Hz$ sampling frequency and achieved 6.3° within-subject accuracy. While promising, the very limited performance of the system restricts applicability to basic touchless interaction [Li et al. 2017].

PSOG exploits the varying IR reflectivity of different eye components, such as the pupil, iris, sclera, etc. As the eye rotates within the globe during gaze shifts, intensities of reflections captured by IR sensors are perturbed, thereby providing the source for estimating gaze location. PSOG designs are distinguished according to both physical parameters of the IR transceiver hardware, along with the signal processing techniques used to map sensors output to the estimated gaze location. The recent review of the various PSOG array designs can be found in [Rigas et al. 2018]. For purposes of our current work, we focus solely on signal mapping techniques using a simulated workflow. This section is restricted to describing the limitations of two recent publications which have motivated our current work.

While PSOG offers a promising alternative for meeting the performance requirements of next-generation VR solutions, sensitivity to hardware shifts limits practical viability. Two recent approaches have been proposed to address this limitation. Rigas et al. suggested a fused sensing modality known as Hybrid PS-V [Rigas et al. 2017]. In this approach, gaze estimates formed using a high-speed ($1000Hz$) PSOG component are fused with low-speed VOG ($5Hz$) sensor output to improve spatial accuracy in the presence of sensor shifts. The authors simulated a $2x2$ grid of IR receivers using reflectivity estimates obtained from synthetic images of a close eye region. Eye position was inferred from simulated PSOG outputs by a simple deterministic mapping. Due to the decreased VOG sampling rate, the proposed architecture was estimated to consume only $15mW$ of power, a reduction of several orders of magnitude versus commercially-available VOG-only systems operating at the same sampling frequency (e.g. EyeLink 1000). Spatial accuracy of less than 1.0° of the visual angle was achieved in the presence of sensor shifts in the range of $\pm 2mm$. The main drawbacks of this approach are brief spike artifacts in the output eye signal during sensor movement phase and up to $200ms$ latency of the correction system. Additionally, the reliance upon an additional tracking modality may prove infeasible for resource-constrained environments. Finally, the work uses synthetically generated eye images for a single artificial subject, which greatly inhibits inference regarding generalization.

To eliminate the VOG requirement, Zemblys et al. utilized machine learning to achieve shift-robustness [Zemblys and Komogortsev 2018]. MLP was used to build the mapping between the PSOG output and gaze location. Model hyper-parameters were determined empirically using a standard grid-search procedure. The authors evaluated different sensor placements and found that $3x5$ rectangular grid works best. The best setup found has spatial accuracy of 0.48° of the visual angle for sensor shifts in the range of $\pm 1.75mm$. However, it is characterized by the same limitations of the previous work with respect to the utilization of a single-model synthetic data source. It is still not clear how well those approaches will work when applied to real images and diverse eye traits. Moreover, the machine learning models explored are not well-suited to capture the inherent spatial dependencies of PSOG outputs, compared to the convolutional architectures considered herein.

Our work addresses the limitations of these two studies by considering more advanced machine learning techniques on data gathered from real recordings of different subjects.

3 METHODOLOGY

3.1 Equipment

A custom ET solution introduced in [Abdulin and Komogortsev 2017] was used for data collection. This system was used in place of standard commercial solutions, such as EyeLink 1000, due to the failure of that equipment to save images obtained by the VOG sensor. The custom solution offers an additional advantage in a research environment, as it is highly reconfigurable (i.e.: easily replaceable IR emitters, camera, and chin-rest) and uses open-source software. To collect data used in this work, the system uses an 850 nm IR illuminator, ThorLabs DCC1545M camera, and a hot mirror and chin-rest from EyeLink 1000. The camera recorded monocular images with a resolution of $348x640$ pixels at 120 fps. The screen to display stimulus was installed $500mm$ away from the chin-rest. It has a resolution of $1280x1024$, and physical size of $374x300mm$, affording a target range of ±19.1° (horizontal) and ±16.7° (vertical) degrees of visual angle. All computations were performed using a workstation with the following specifications: Intel Core i7-6700K CPU, 16GB RAM, Nvidia GTX 970 GPU.

3.2 Data

Random horizontal and vertical step-stimulus saccade task was used to collect data from 23 subjects. Every subject's record consists of the set of images and respective eye-movement signal file. Each line of this file contains a timestamp, and corresponding horizontal and vertical gaze estimates expressed in degrees of the visual angle.

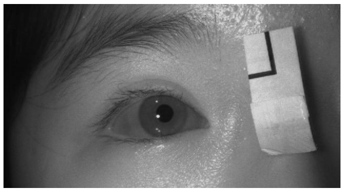


Figure 1: One IR frame from the data set

If for some reason eye-tracker was not able to infer eye gaze at the specific timestamp, (i.e: during blinks, etc.), the corresponding line is filled with NaNs. Moreover, by inspecting frequency histograms of eye gaze positions, it was detected that sometimes eye-tracker reports values far outside of the screen range (probably, just before or right after a blink). To account for that, every eye gaze position outside of the ±20° range either horizontally or vertically is filtered out.

3.3 Preprocessing

To begin preprocessing, potential head-movements were estimated by tracking the position of the angular marker attached to nasal bone area (as shown in Figure 1) throughout each image sequence on a per-subject basis. The marker's position was manually found in the first frame. Then, the location of it was updated by searching in a nearby region centered around the previous point. This estimated shift is hereby denoted as $\Delta head$.

In prior work using synthetic eye images, sensor shifts were simulated by moving the position of a virtual camera within Blender. As our data was collected using a real IR camera with a fixed position, this approach was not possible. To simulate shifting, a cropping strategy was employed on the captured image as follows. By taking a picture of a ruler it was determined that $0.5mm$ corresponds to about 4 pixel shift for our hardware setup. In this study, we test the range of $\pm 2mm$ shift with 0.5 mm step which results in ±16 pixels shift with 4 pixels step. Let Δcam denote the simulated shift. Then for each image, the top-left corner of a cropping rectangle is positioned at $start + \Delta head + \Delta cam$, where $start$ is determined manually for each recording so that for no shift and neutral eye position case the eye in the cropped image will be located in the center of PSOG sensor receptive field.

PSOG output is simulated using the exact same approach as in [Zemblys and Komogortsev 2018]. The sensor simulated by a set of receptive windows aligned in a $3x5$ rectangular grid. Each window consists of $121x121$ pixels , with mirror padding employed if it runs out of the image. To reflect the angular reception loss observed in a photosensor, each window is convolved with a Gaussian kernel with 0 mean and 1/4 of the window size as standard deviation. The average pixel intensity over the whole window after convolution is considered as the corresponding photosensor's raw output.

The entire preprocessing pipeline is depicted in Figure 2.

3.4 Machine learning

The prior machine learning study used a multilayer perceptron (MLP) model to learn the mapping from raw photosensor outputs to eye gaze position. The two-dimensional array of outputs (in our case, $3x5$) is flattened to a one-dimensional vector to form the MLP input. This representation does not preserve spatial information, which may limit the power of this approach. CNN is a natural solution for addressing this limitation, as spatial information is maintained throughout the initial convolutional layers. For comparison purposes, we benchmark a CNN model against a comparable MLP architecture.

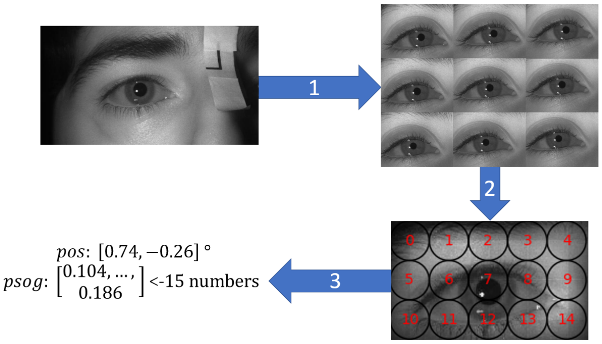

1
2
3
pos: $[0.74, -0.26]$ °
$psog$: $\begin{bmatrix}0.104, ..., \\ 0.186\end{bmatrix}$ <-15 numbers
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

Figure 2: Preprocessing pipeline:

1. **Account for head movements, then shift and crop.**
2. **For each cropped image, simulate PSOG sensor output (for each detection window one standard deviation of the gaussian kernel is depicted).**
3. **Save raw sensor output with corresponding eye gaze position.**

In practice, the amount of training data available will be restricted to what can be obtained during calibration, which may limit the performance of the learned map. To expand the pool of available training data, we propose a transfer learning approach, where network weights are initialized through training over a large pool of other subjects. We hope that this approach (hereby denoted as the "fine-tune" approach) will allow layers of the pre-trained network to learn features that are general for eye movements across subjects. By fine-tuning this model for the specific subject using calibration data, we expect faster convergence and improved accuracy versus training using randomly initialized weights (hereby denoted as the "from-scratch" approach). Note that for both architectures only fully-connected layers were fine-tuned .

We use data whitening followed by PCA that keeps components which explain more than 99% of variance for MLP architecture. To preserve spatial information, only data whitening is used for CNN architecture.

To assess the validity of the "fine-tune" approach, we partitioned available data into testing and training sets using an idea similar to k-fold cross-validation. Namely, the whole set is split into following 6 batches (subject's ids are listed): [1, 2, 3, 4]; [5, 6, 7, 8]; [10, 11, 12, 13]; [14, 15, 16, 17]; [18, 19, 20]; [21, 22, 23];. Then, each batch is separately considered as a testing set, with the remaining batches used for training and validation purposes. For the "from-scratch" approach, training and testing sets are obtained by partitioning data for each individual subject

Hyper-parameters for both MLP and CNN architectures were determined using a grid search approach. Two setups were investigated reflecting the varying use-cases of the sensor. Namely, while we are most interested in embedded VR applications, we also seek to investigate unconstrained performance for stand-alone implementations. We hereby denote these two scenarios of interest as low-power and high-power setup, respectively.

According to [Bengio 2012], a network with the same size of layers shows comparable performance to the case of a network with layers of varying size but the same overall number of parameters. We use this finding to simplify the grid search procedure. Instead of having a separate number of neurons for each fully-connected layer or number of filters for each convolution layer , we can describe the whole MLP network with two parameters: $L-$ number of layers with N neurons in each; and the whole CNN network with four values: $L_{conv}-$ number of convolution layers, each of depth D, $L_{fc}-$ number of fully-connected layers with N neurons in each. Note that for all CNN layers, a $3x3$ kernel size with a stride of 1 is used, with a zero padding to preserve filters size ('same' in Keras terms). The same designations and assumptions about a convolutional layer will be used throughout this section. Each combination of parameters listed below composes the preliminary grid search space for the specific setup:

- Low-power setup, MLP architecture:

```
$L$ from [3,4,5,6]:
  $N$ from [16,20,24,28,32]
```

- Low-power setup, CNN architecture:

```
$L_{conv}$ from [1,2,4]:
  $D$ from [4,8,16]:
    $L_{fc}$ from [3,4,5]:
      $N$ from [16,20,24,28,32]
```

- High-power setup, MLP architecture:

```
$L$ from [3,4,5,6]:
  $N$ from [16,32,48,64,96]
```

- High-power setup, CNN architecture:

```
$L_{conv}$ from [1,2,4]:
  $D$ from [4,8,16]:
    $L_{fc}$ from [3,4,5]:
      $N$ from [16,32,48,64,96]
```

The number of operations performed during the inference directly influences the power consumption of the system. As an additional restriction, the upper-limit for architecture complexity was set in accordance to resources available on the destination hardware. We use the same estimation of the power consumption of a neural network, as in [Zemblys and Komogortsev 2018] - the number of floating point operations per second (FLOPS) needed to do the forward pass. Multiplications only are taken into account. We use following formulas to approximate FLOPS complexity for:

- MLP architecture:

$$MLP_{flops} = f(1, n_{in}, n_1) + f(n_{in}, n_1, n_2) + ...+ f(n_{L-1}, n_L, n_{out}) \tag{1}$$

where n_i- number of neurons in i-th layer, $(n_{in}, 1), (n_{out}, 1)$ - dimensions of the input and output vectors respectively, and:

$$f(x, y, z) = 2xyz - xz$$

In our case, when the size of the input vector depends on the PCA result, the number of neurons n is the same for all layers and $L > 2$, this simplifies to:

$$MLP_{flops} = f(1, n_{PCA}, n) + f(n_{PCA}, n, n) + (L-2) * f(n, n, n) + f(n, n, 2) \tag{2}$$

- CNN architecture:

$$CNN_{flops} = m * (1 + D_1 + ... + D_{L_{conv}-1}) + f(1, n_{flatten}, n_1) + f(n_{flatten}, n_1, n_2) + ...+ f(n_{L_{fc}-1}, n_{L_{fc}}, n_{out}) \tag{3}$$

where $m-$ multiplications needed for convolution of one filter, D_i depth (number of filters) of $i-$th convolution layer and last convolution layer flattened to $n_{flatten}$ neurons. f and n_i declared the same as above.
In our case, when input vector is $3x5$, depth D is the same for all convolution layers, the number of neurons n is the same for all fully-connected layers and $L_{fc} > 2$, this simplifies to:

$$CNN_{flops} = 135 + 135 * D * (L_{conv} - 1) + f(1, 15 * D, n) + f(15 * D, n, n) + (L_{fc} - 2) * f(n, n, n) + f(n, n, 2) \tag{4}$$

Raspberry Pi3 is chosen as the baseline for the low-power setup, as it is a widely available and low-cost embedded solution. It is able to deliver 0.084 to 6.165 GFLOPS, and has power consumption of $16 - 1200$ MFLOPS per watt, depending on the task. We pick the highest restriction, i.e. the lower-bound of the performance range. Taking into account that the system should be able to operate on $1000Hz$, for the low-power setup the limitation on the neural network complexity is set to 0.084 MFLOPS. This translates to an estimated power consumption range from $70mW$ to $5.25W$ for the low-power setup. Knowing this limitation and using 2, 4, we can determine whether a specific architecture from the grid-search is inside our range of interest. Modern nVidia GPU, such as nVidia GTX 970, is able to deliver 122 GFLOPS. It is far more than required by the most complex network from the grid search space, so we do not put any additional restriction on the high-power setup.

3.5 Statistical analysis

In the following statistical analysis, we use the term "architecture" to denote neural network architecture tested: either MLP or CNN; and "approach" to talk about whether corresponding architecture is fine-tuned or trained from scratch.

We perform statistical analysis to determine if observed spatial accuracies could have occurred by chance. To accomplish this goal, the effect of architecture and approach was tested using Analysis of Variance (ANOVA). Mean estimates for each combination of architecture and approach were calculated. The significant interaction effect was followed up by a series of post-hoc paired comparisons, which were adjusted for multiple comparisons using the method of Scheffe [Scheffe 1967]. A conventional alpha level of 0.05 was used to determine statistical significance.

3.6 Implementation

We use Keras 2.2.4 framework for Python 3.6.7 with TensorFlow 1.12 as the backend. All error bars are created with Plotly 3.4, statistical analysis of the results is done with SAS University Edition. Neural networks visualization is done with 'NN-SVG' [LeNail 2019]. The data set and codebase snapshot are available on TxState Digital Library [1]. The up-to-date codebase is available on GitHub. [2]

4 RESULTS

4.1 Model selection

To utilize available data during the grid search as much as possible, it was split into 75% of train and 25% validation sets, without test part. Architectures with the best performance on the validation set are:

- Low-power, MLP: $Dense(\,24\,)\,x\,4$, ≤ 0.074 MFLOPS [3]
- Low-power CNN: $Conv2D(\,4\,)\,x\,2\ +\ Dense(\,20\,)\,x\,4$, 0.083 MFLOPS
- High-power MLP: $Dense(\,96\,)\,x\,5$, ≤ 5.595 MFLOPS [3]
- High-power CNN: $Conv2D(\,16\,)\,x\,2\ +\ Dense(\,96\,)\,x\,5$, 9.766 MFLOPS

Layers names follow Keras' naming convention, here:
$Conv2D(\,D\,)\,x\,L$ means L convolution layers in a row, each with D filters; $Dense(\,N\,)\,x\,L$ means L fully-connected layers in a row, each with N neurons. Each architecture is visualized in Figure 3

4.2 Training process details

To explore variability in training performance with respect to initial condition, training was repeated 10 times for random initial values. This was done since Keras does not allow users to achieve reproducible results by fixing a seed value. We use Adam with Nesterov momentum [Dozat 2016] as the optimization algorithm. The learning rate is set to 0.001, epsilon to 10^{-8}, and all other parameters to the default values. Eye-tracker output is used as a ground-truth. The discrepancy between the model output and ground-truth is measured with mean-squared loss. To prevent overfitting, the early-stopping [Caruana et al. 2001] technique is used. The training process ends as soon as the lowest validation loss have not improved for the specified number of steps (patience parameter in Keras). For the "fine-tune" approach, models are pre-trained with batch size of 2000 and patience of 100. Data is split into several batches as described in the Machine learning subsection. Every batch is individually considered as the test set, with remaining data split into 80%/20% for train and validation sets, respectively. For both the "fine-tune" approach and for the training "from scratch", batch size is set to 200, and patience to 50. The only available data to build a map between photosensors output and eye gaze of the specific subject is gathered during the calibration procedure. To resemble the small amount of such data, a 24%/6%/70% train, validation and test split is used for both approaches. Note that the spatial distribution of gaze locations for the train split is different from calibration data. The attempt to address this issue is made in the Discussion section.

4.3 Testing results analysis

For both setups, Figure 4 shows error bars that represents the mean and one standard deviation of the spatial accuracy on the test set obtained for 10 repetitions of training for each combination of architecture and approach for all subjects.

As can be seen, the best spatial performance is 0.67° and 0.55° for the low-power and high-power setup respectively. This observation is supported by the paired comparisons in Table 1b (high-power: Table 1d). For high-power setup, it is achieved with CNN architecture retrained from scratch for each subject. Nevertheless, for low-power setup of CNN architecture the performance of the "from scratch" approach isn't significantly different from the "fine-tune" one ($p = 0.29$). Additional evaluation is provided in the Discussion section to pick the best approach.

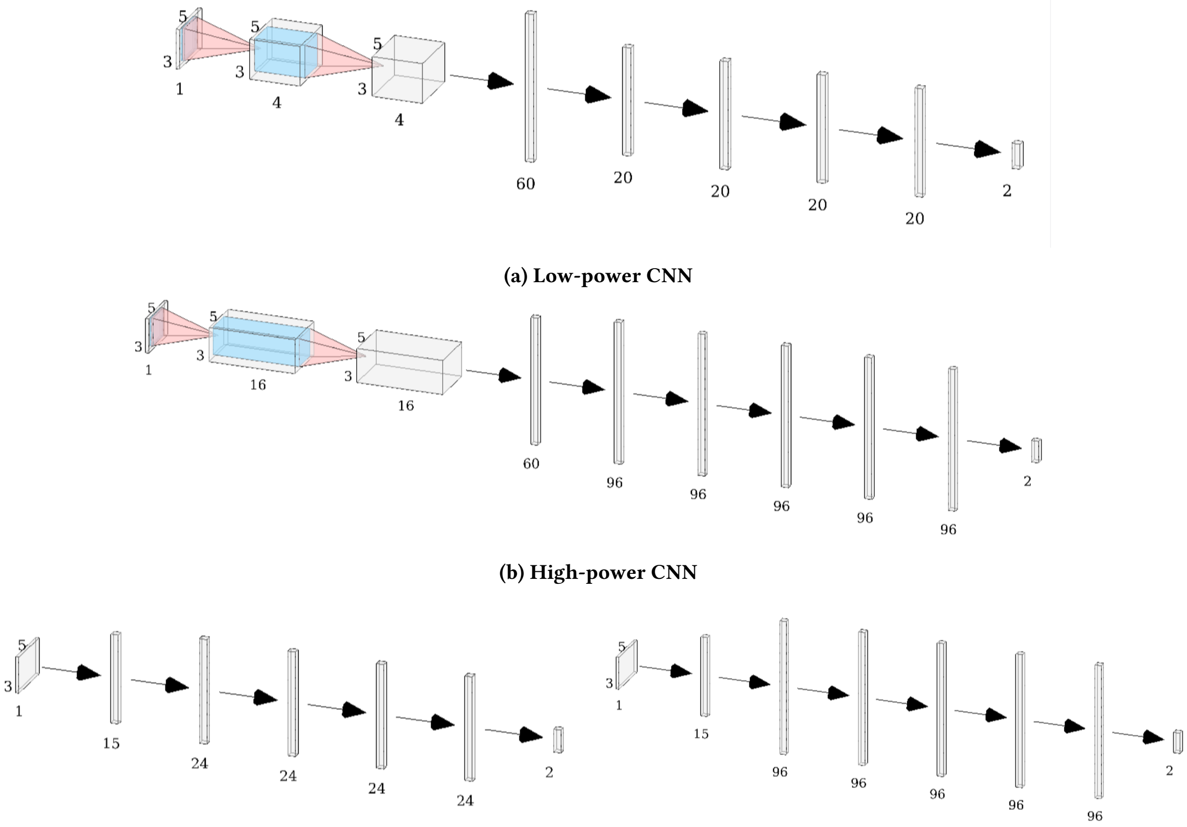
In Table 1a we present F-tests of the effect of architecture, approach, and their interaction for the low-power setup (high-power: Table 1c). The statistically significant interaction means that the effect of approach depends on the architecture and vice versa. Table 1b presents the post-hoc paired comparisons for all combinations of architecture and approach for the low-power setup (high-power: Table 1d).

5 DISCUSSION

The time needed to enroll a new user is of critical importance for consumer-grade devices. The most time consuming step for our approach is model training using calibration data. While fine-tuning model weights should reduce training time, results from the main evaluation demonstrated negligible difference. One possible explanation for this effect was the use of a small batch size. To analyze this, we trained CNN architecture of the low-power setup only one time for each subject with two batch sizes: 200 and 2000. Means (± one standard deviation) of resulting time and accuracy for both values are listed in Table 2. Though for batch size of 200 two approaches seems indistinguishable, it can be seen that a higher batch size can drastically improve training time but on the cost of decreased spatial accuracy. Despite training from scratch being more accurate, it is up to 1.9 times slower, so **we pick "fine-tune" approach as the best** overall one. Moreover, this approach combined with a bigger batch size seems much more promising in terms of getting reasonable training time on the embedded platform. The proper evaluation of time complexity on actual hardware with possible optimization applied should be done in the future work.

Our work has following limitations: sensor shifts simulated by cropping, simulated photosensors output, and a random split of the whole recording into training and testing set. While the first two should give a close approximation to results obtained using actual hardware, the later does not reflect a real-case scenario where only calibration data is available to train on. Unfortunately, our eye-tracker did not provide the map between frames and eye-position signal during calibration. Fortunately, the grid of points used as stimuli for the random saccades task contains a subset similar to the calibration grid. We exploited this observation, training the model only on samples that have euclidean distance no more

[1] https://digital.library.txstate.edu/handle/10877/7955
[2] https://github.com/pseudowolfvn/psog_nn/tree/etra2019
[3] The upper limit is provided due to the different number of components left after PCA for different subjects



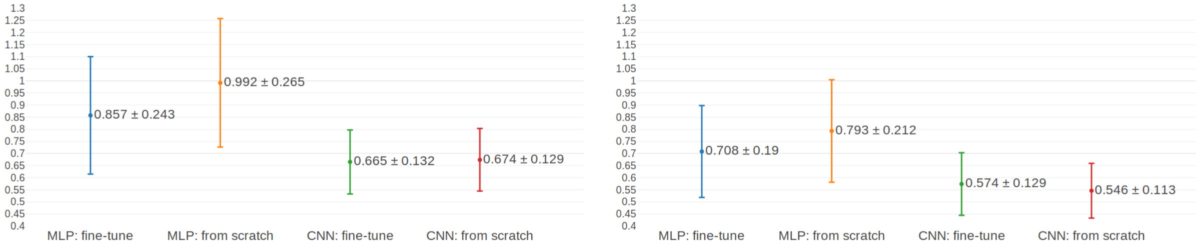
(a) Low-power CNN

(b) High-power CNN

(c) Low-power MLP

(d) High-power MLP

Figure 3: Neural network architectures used



(a) Low-power setup

(b) High-power setup

Figure 4: Spatial accuracy error bars

than 35 pixels from any point from the "calibration" grid (points 1, 3, 5, 9, 11, 13, 17, 19, 21 in Figure 5). For most subjects, the data is unevenly spread around "calibration" points, providing very limited coverage for some participants as shown in Figure 5a. The distribution is much better for some subjects as depicted in Figure 5b. This limitation cannot be fixed using an increased distance threshold, as it will include points that do not correspond to the fixations of interest. For more reliable results, proper calibration data should be collected.

Eye-tracking has numerous potential applications in VR. Our study demonstrates adequate accuracy at sufficiently low power consumption to support general touchless interaction. However, it is still not clear how well this approach preserves specific features in the eye-signal that are essential for biometrics and health assessment. To verify the feasibility of our design for all possible VR applications, the fabrication and assessment of actual hardware is required.

Table 1: Statistical analysis. Groups 1, 2, 3, 4 stands for 'MLP: fine-tune', 'MLP: from scratch', 'CNN: fine-tune', 'CNN: from scratch' combinations of 'architecture: approach' respectively.

(a) Low-power setup, Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
arch	1	33.2	18.56	0.0001
appr	1	872	519.91	<.0001
arch*appr	1	872	402.01	<.0001

(b) Low-power setup, Differences of LS Means

№	Group A	Group B	Estimate	Standard Error	DF	t Value	Adj P
1	1	2	-0.1344	0.004434	872	-30.30	<.0001
2	1	3	0.1922	0.05930	33.4	3.24	0.0151
3	1	4	0.1836	0.05930	33.4	3.10	0.0229
4	2	3	0.3266	0.05930	33.4	5.51	<.0001
5	2	4	0.3180	0.05930	33.4	5.36	<.0001
6	3	4	-0.00863	0.004434	872	-1.95	0.2863

(c) High-power setup, Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
arch	1	36.3	15.20	0.0004
appr	1	872	156.57	<.0001
arch*appr	1	872	602.26	<.0001

(d) High-power setup, Differences of LS Means

№	Group A	Group B	Estimate	Standard Error	DF	t Value	Adj P
1	1	2	-0.08497	0.003243	872	-26.20	<.0001
2	1	3	0.1345	0.04898	36.5	2.75	0.0573
3	1	4	0.1620	0.04898	36.5	3.31	0.0124
4	2	3	0.2194	0.04898	36.5	4.48	0.0002
5	2	4	0.2470	0.04898	36.5	5.04	<.0001
6	3	4	0.02758	0.003243	872	8.51	<.0001

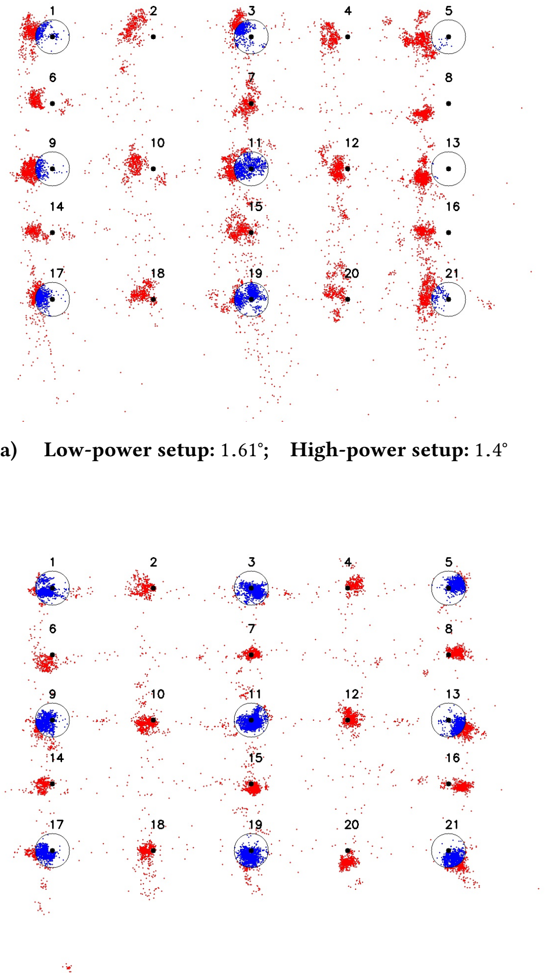
Table 2: Low-power setup, CNN training time analysis

Batch size	Fine-tune (time/acc)	From scratch (time/acc)
200	69.84 ± 21.07*sec* (0.72 ± 0.17°)	65.64 ± 20.13*sec* (0.67 ± 0.13°)
2000	9.15 ± 4.5*sec* (1.07 ± 0.3°)	17.45 ± 3.48*sec* (0.77 ± 0.14°)

6 CONCLUSION

In this paper, we evaluated the PSOG-based eye-tracking sensor for portable VR headsets. Among all tested neural network architectures and approaches for calibrating the system to each new subject's eye traits, fine-tuning of convolutional neural network works best. This combination was tested as two setups: with limited power consumption for embedded devices, and with no such restriction aiming for the best spatial accuracy.

The first setup can be used to implement the system on portable hardware platforms that already exist, with high sampling frequency of 1000 Hz and spatial accuracy of 1.07° (taking into account performance and model training time trade-off). The second setup shows the potential of the proposed system for current wired hardware, because it maintains higher sampling frequency with better spatial accuracy of 0.55°. Both setups are robust to sensor shifts.



(a) **Low-power setup:** 1.61°; **High-power setup:** 1.4°

(b) **Low-power setup:** 1.03°; **High-power setup:** 0.85°

Figure 5: Spatial accuracy on testing set for: bad (a, subject 6), good (b, subject 8) split of data into blue training and red validation + testing samples that should reflect calibration procedure. Circles represents 35px area around "calibration" points

ACKNOWLEDGMENTS

We are thankful to Dr. Lee Friedman for his help with statistical analysis, and to Dr. Evgeny Abdulin for creating the hardware used in this study. This work is supported by the National Science Foundation under Grant Nos: CNS-1250718 and CNS-1714623. The work is inspired by the Google Virtual Reality Research Award and Google Global Faculty Research Award bestowed on Dr. Komogortsev in 2017 and 2019, respectively.

REFERENCES

Evgeniy R Abdulin and Oleg V Komogortsev. 2017. Study of Additional Eye-Related Features for Future Eye-Tracking Techniques. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 1457–1463.

Karan Ahuja, Rahul Islam, Varun Parashar, Kuntal Dey, Chris Harrison, and Mayank Goel. 2018. EyeSpyVR: Interactive Eye Sensing Using Off-the-Shelf, Smartphone-Based VR Headsets. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 2 (2018), 57.

Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*. Springer, 437–478.

Rich Caruana, Steve Lawrence, and C Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems*. 402–408.

Timothy Dozat. 2016. Incorporating nesterov momentum into adam. (2016).

Lee Friedman, Mark S Nixon, and Oleg V Komogortsev. 2017. Method to assess the temporal persistence of potential biometric features: Application to oculomotor, gait, face and brain structure databases. *PloS one* 12, 6 (2017), e0178501.

Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. 2012. Foveated 3D graphics. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 164.

Alexander LeNail. 2019. NN-SVG: Publication-Ready Neural Network Architecture Schematics. *Journal of Open Source Software* 4(33) (2019), 747. https://doi.org/10.21105/joss.00747

Tianxing Li, Qiang Liu, and Xia Zhou. 2017. Ultra-Low Power Gaze Tracking for Virtual Reality. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. ACM, 25.

SR Research. 2016. *The EyeLink® 1000 Plus Eye Tracker*. https://www.sr-research.com/wp-content/uploads/2017/07/EyeLink-1000-Plus-Brochure-2016-November.pdf.

Ioannis Rigas, Hayes Raffle, and Oleg V Komogortsev. 2017. Hybrid PS-V Technique: A Novel Sensor Fusion Approach for Fast Mobile Eye-Tracking with Sensor-Shift Aware Correction. *IEEE Sensors Journal* 17, 24 (2017), 8356–8366.

Ioannis Rigas, Hayes Raffle, and Oleg V Komogortsev. 2018. Photosensor Oculography: Survey and Parametric Analysis of Designs using Model-Based Simulation. *IEEE Transactions on Human-Machine Systems* 99 (2018), 1–12.

Henry Scheffe. 1967. *The analysis of variance*. Wiley.

Nicholas Torok, Victor Guillemin Jr, and JM Barnothy. 1951. LXXX Photoelectric Nystagmography. *Annals of Otology, Rhinology & Laryngology* 60, 4 (1951), 917–926.

Raimondas Zemblys and Oleg Komogortsev. 2018. Making stand-alone PS-OG technology tolerant to the equipment shifts. In *Proceedings of the 7th Workshop on Pervasive Eye Tracking and Mobile Eye-Based Interaction*. ACM, 2.