HPCGen: Hierarchical K-Means Clustering and Level Based Principal Components for Scan Path Genaration

WOLFGANG FUHL and ENKELEJDA KASNECI,
University Tübingen, Germany

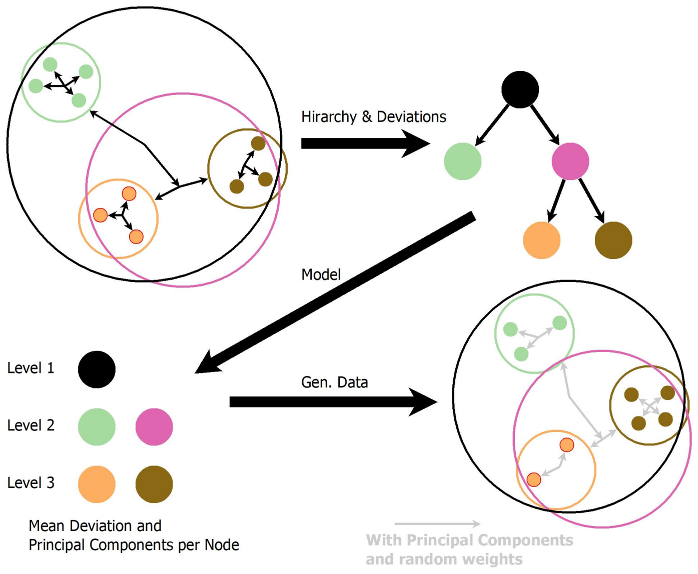

Fig. 1. The general concept of our proposed approach. We use multiple gaze recordings and cluster them hierarchically. In each cluster, we compute the shift vectors to the cluster center of the data points and compute the next level with the clustering of the deviations. Based on the shifts we compute the principal components which we can afterwards use to generate a scan path. The shown example uses only two clusters per level, our approach can use different amounts of clusters on each level.

In this paper, we present a new approach for decomposing scan paths and its utility for generating new scan paths. For this purpose, we use the K-Means clustering procedure to the raw gaze data and subsequently iteratively to find more clusters in the found clusters. The found clusters are grouped for each level in the hierarchy, and the most important principal components are computed from the data contained in them. Using this tree hierarchy and the principal components, new scan paths can be generated that match the human behavior of the original data. We show that this generated data is very useful for generating new data for scan path classification but can also be used to generate fake scan paths. Code can be downloaded here https://atreus.informatik.uni-tuebingen.de/seafile/d/8e2ab8c3fdd444e1a135/?p=%2FHPCGen&mode=list.

CCS Concepts: • **Applied computing** → *Mathematics and statistics*; *Personal computers and PC applications*; • **Computing methodologies** → *Machine learning*; • **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**.

Additional Key Words and Phrases: Eye Tracking, Gaze Simulator, Gaze Behaviour, Gaze, Fixed Stimulus, Dynamic Stimulus, Classification, Scan path, Gaze generation, Scan path generation

ACM Reference Format:
Wolfgang Fuhl and Enkelejda Kasneci. 2022. HPCGen: Hierarchical K-Means Clustering and Level Based Principal Components for Scan Path Genaration. In *2022 Symposium on Eye Tracking Research and Applications (ETRA '22), June 8–11, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3517031.3529625

1 INTRODUCTION

In today's world, eye tracking can be found in many fields. This includes human-machine interaction [1, 23], computer graphics [33, 47], self-diagnosis systems [25, 28, 44], measurement of eye misalignment [15, 34], detection of neurological diseases [10, 37], behavioral research [7, 29], learning systems [24, 36], expertise determination [5, 40], driver observation [30, 41], market research [4, 42], and many more.

In computer graphics, for example, the main interest is to render only the 2 degrees of the fovea with a high resolution [33, 47]. In the rest of the image it is sufficient to render roughly, which would result in a significant reduction of computational effort [33, 47]. One problem that exists here is the need for many scan paths, which is necessary for testing the system. In the area of market research, one evaluates product placements or the online presence of manufacturers and sellers [4, 42]. Here, many scan paths are also needed to make a general statement or a statement about specific customer groups. In self-diagnostic systems, a disease or defect is diagnosed based on the user's behavior [10, 25, 28, 37, 44], which can then later be treated or examined by a doctor. For these systems to work reliably, it is also necessary to use a lot of data. In the area of driver observation, the same applies. Here it is necessary to detect inattentive behavior or drowsiness and warn the driver [30, 41]. These systems also need to be tested on a lot of data to ensure their reliability and optimize their usability for the driver. In the field of human-machine interaction, many data are also needed, which are used to test interactions and make the final algorithm as reliable as possible [1, 23]. Thus, it is clear that data is becoming increasingly important in the machine learning era. Since the recording and annotation of data is very time-consuming, and this data is subject to data protection, science has long been concerned with the simulation of data and its use for the validation of systems or for training machine learning methods.

In the literature, there are of course a variety of algorithms that have dealt with the simulation and generation of data in different data domains [39, 46]. However, simulating human behavior remains a challenge to this day. The latest methods use Biological Reference Values, which are used to generate scan paths [19]. However, the focus of this work is to generate eye movement types as realistically as possible based on the velocity characteristics. Another approach deals with the generation of scan paths via a variational autoencoder, which learns distribution parameters [20]. New data can then be generated using these distribution parameters, but these were only used as an add-on for data augmentation. To hide specific data in a scan path, such as the Personal Features, a reinforcement learning based approach was presented [17]. Here, the data in the middle part of the autoencoder is manipulated to manipulate the scan path. The focus of this work was to create scan paths that contain only selected information. Another work uses autoencoders together with data augmentation in the target layer of the autoencoder to obtain a representation of a scan path that can be classified with linear models [3]. All the work mentioned here generates scan paths, but never with the focus to simulate realistic human behavior.

In this work, we present an algorithm that allows to learn a scan path model hierarchically from given human scan path images. This means that based on the given scan paths, new scan paths can be generated that reflect exactly this behavior. An example would be the search for items in an online store. If some data from humans is given here, a model can be generated with our algorithm and with this model new scan paths for the search of articles. This can be used to generate data for scan path classification as well as data to validate an online presence. Of course, there are limitations to the application of our algorithm, but these are described in the limitations' section of this paper.

Our contributions to the state of the art are:

(1) A novel approach for scan path generation which is totally data driven.
(2) Code for the generation of scan path, as well as code to train new models.
(3) Comparison to other state of the art approaches, as well as the combination with other approaches to generate training data.
(4) Evaluation of fake scan path generation

2 RELATED WORK

The first part of the related work focuses on the simulation of eye movements. In 2002 an approach on image rendering was proposed in [27]. The authors focused on the generation of realistic saccade movements, and it was also possible to simulate smooth pursuits as well as different vergence angels. Another approach which focuses on the generation of eye movements as well as head rotations was proposed in [31]. Here the head movements are simulated additionally, but the simulator automatically triggers an eye movement, if the head motion starts. The next improvement was proposed in [26]. Here, head movements do not trigger eye movements anymore and in addition to the standard eye movement types, blinks are simulated as well. For the generation of the eye movements, the method in [26] uses sound files. There exists also an approach which is motivated by the eye muscle [45] movement. The approach is described in [13] and focuses especially on the eye rotation. This approach uses predefined sequences of eye movements to generate the eye rotations, for which it is not a real eye movement nor a gaze data generation approach. The last approach which is based on image rendering is [48]. This approach randomly generates eye images with different textures as well as different orientations. The main purpose of this approach is to generate training data for computer vision algorithms, since no eye movement types nor realistic gaze data is generated. The so far mentioned approaches are usually used to generate images or sequences of images corresponding to special eye movement types. Their main use is in animation of characters or the interaction with humans [38]. Therefore, the generated eye movement sequences do not correspond to real viewing behavior of humans, since no saliency information nor task specific knowledge is integrated.

An approach which focuses on a realistic simulation of gaze sequences based on saliency information is described in [6]. The approach is based on a statistical model and uses saliency information together with random number generators to simulate gaze data. In [11] the before mentioned approach was refined by adding different distributions for noise generation, which is a part in any real world data recording of gaze data. Additionally, the authors in [14] added jitter of the gaze signal itself based on a normal distribution. An extension of those approaches which supports different sampling rates as well as dynamic sampling rates, which is usually the case if the cameras have no fixed exposure time, is proposed in [19]. The approach generates random sequences of eye movement types together with the gaze information and is able to map them to static and dynamic stimuli as well as to other gaze data. The approach itself uses different distributions as well as scientifically based equations for the calculation of the gaze signal and eye movements.

With the upcoming of deep neural networks and the recent improvements in machine learning, other novel approaches have been proposed recently. A deep recurrent neural network to generate gaze heatmaps was proposed in [43]. It is applied on static images and tries to generate a realistic gaze heatmap. While this was one of the first approaches, it has the limitation, that it only works on already seen images during training. An approach using generative adversarial neural networks (GAN) was proposed in [2]. Here, the input consists of static images and saliency maps out of which a scan path is generated. An approache for scan path manipulation using an autoencoder and reinforcement learning is described in [17]. The authors did not try to compute a realistic scan path, but to manipulate the scan path in a way that some information is removed while other information is still contained in the scan path. An approach using a variational autoencoder is proposed in [20]. Here, the autoencoder learns parameters of distributions in the central block of the autoencoder. With a random sequence of numbers and those parameters, a new scan path can be generated. The authors used this simulation technique only for data augmentation, which helped to improve the classification accuracy. In [3] an autoencoder which combines different feature extractors (Convolutions with different sizes) was proposed. The authors did not try to generate a realistic sequence but use the central part of the autoencoder as general metric for scan path classification.

3 METHOD

Algorithm 1 Model Generation

```
1:  procedure GenerateModel($GazeData, MaxLevel, NumClusters, DynCluster$)
2:      $Model = 0$
3:      $Clusters[1] = GazeData$        ▷ Multiple recordings with two or three dimensions (X,Y,time)
4:      for $l = 1$ to $MaxLevel$ do
5:          if $DynCluster = True$ then
6:              $NumClusters = UpdateRule(NumClusters, l)$
7:          for $c = 1$ to $Size(Clusters)$ do
8:              $Model[l][c].m = Mean(Clusters[c])$        ▷ First level is position other levels are deviations
9:              $SubClusters[c] = KMeans(Clusters[c], NumClusters)$        ▷ SubClusters has multiple dimensions
10:                 ▷ The clustering will assigne each gaze point a cluster if there are to few gaze points.
11:                 ▷ The shifts to those gaze points will be computed and used in the PCA.
12:             $CombineCloseSubClusters(SubClusters)$        ▷ Assign clusters between multiple recordings
13:             $Model[l][c].p = PCA(Mean(SubClusters[c]) - Model[l][c].m)$
14:         $Clusters = ToOneDimension(SubClusters)$
15:     return $Model$
```

The general idea of our approach is to extract the structure of gaze data and reuse this structure to generate new gaze data (Figure 1). Gaze data generally consists of eye movements such as fixations and saccades [12]. These fixations and saccades form clusters. If one now observes a longer time interval, several fixations form new clusters, which are interesting or important image areas or objects in fixed stimuli or in dynamic scenes. If we follow this formation of clusters over the whole image, a hierarchy of clusters is formed, which becomes deeper and finer. In our case, we go the opposite way and try to find the rough clusters from the whole recordings in the first step and then calculate finer and finer divisions for each cluster.

If we now look at several images of people, we notice that the same areas are often looked at. These areas are clusters in our hierarchy. Thus, our idea is to assign the clusters according to the hierarchy and position (the position can be two or three-dimensional. Thus, include spatial position and time) between the images to each other. Since each cluster has further sub clusters (In the last level of the hierarchy these are viewpoints), we can calculate the displacement of these sub clusters to the parent cluster. Together with the assignment of the clusters, we obtain numerous possible shifts. We consider these displacements as shapes, which can be approximated via an active shape model [8]. This active shape model uses the most important principal components, which can then be used to compute new sub clusters. Once we have calculated the general hierarchy over all images, we can use this hierarchy to generate new gaze data. To do this, we run through the hierarchy from above and select random clusters in each layer, which are positioned based on their parent cluster or the principal components they contain. The leaves of the randomly generated tree are then the gaze points that are generated.

Algorithm 2 Scan path Generation

```
1:  procedure GenerateScanPath($Model, MaxClusters, MaxSubClusters, DynCluster$)
2:      $GazeData = 0$
3:      $ParentClusterMeans[1] = 0$
4:      for $l = 1$ to $MaxLevel$ do
5:          if $DynCluster = True$ then
6:              $MaxClusters = UpdateRule(MaxClusters, l)$
7:              $MaxSubClusters = UpdateRule(MaxSubClusters, l)$
8:          for $lc = 1$ to $ParentClusterMeans$ do
9:              $LastMean = ParentClusterMeans[lc]$
10:             $NumClusters = Rand(0, MaxClusters)$
11:             if $NumClusters = 0$ then
12:                 $NewParentClusterMeans[lc][1] = LastMean$    ▷ Clusters can end before the max level is reached
13:                                                              ▷ They won't be touched in the next iteration
14:             else
15:                 for $c = 1$ to $NumClusters$ do
16:                     $IDX = Rand(1, size(Model[l]))$
17:                     $m = Model[l][IDX].m$    ▷ Mean shift of the model, first level it is the position.
18:                     $p = Model[l][IDX].p$    ▷ Principal components to compute new shifts.
19:                     $NumSubClusters = Rand(1, MaxSubClusters)$
20:                     $RandWeights = RandMatrix(NumSubClusters, ColumnSize(p))$
21:                     $NewParentClusterMeans[lc][c] = LastMean + m + RandWeights * p$
22:         $ParentClusterMeans = ToOneDimension(NewParentClusterMeans)$
23:         if $l = MaxLevel$ then
24:             $GazeData = ParentClusterMeans$
25:             if $TimeDependent = True$ then    ▷ Estimated automatically on the dimensionality of the means
26:                 $GazeData = NormalizeTime(GazeData)$    ▷ Necessary if the time was used for clustering
27:             else
28:                 $GazeData = AddTime(GazeData)$    ▷ Necessary if there is no time given
29:     return $GazeData$
```

In Algorithm 1 the conceptual procedure for generating a model according to our approach is described. As input, it needs the gaze data of several images (GazeData), a maximum depth (MaxLevel), from which the calculation is stopped, an initial number of clusters (NumClusters), and whether the number of clusters should change over the depth or not (DynClusters). The gaze data can be passed both as simple x,y coordinates or additionally with the time, which is then considered with the KMeans procedure for the computation of the clusters. In the top level, an initial mean position is determined, from which the displacements for further clusters are then calculated. As soon as the clusters are calculated on all data, these are assigned between the images in order to be able to calculate general information for possible displacements via the sub clusters. In the following step, these possible shifts are divided into the most important subcomponents using the principal component analysis. These principal components are stored in the model together with the mean displacement (in the first layer the mean position). For each further layer in the model, this procedure is repeated until the last layer is reached or there is too little data in each cluster to form new clusters.

Our procedure to use a model to generate new gaze data, is described in Algorithm 2. As input, this algorithm has, the Model (Model), the maximum number of clusters (MaxClusters), the maximum number of subclusters per cluster (MaxSubCluster), and whether the maximum values for the clusters should change based on the current level (DynCluster). Our algorithm starts at the first level and randomly determines how many clusters to create. For each cluster to be created in this level, the number of subclusters is determined randomly. The displacement of these subclusters is calculated using the principal components and a random matrix. Together with the mean displacement or mean position for the first layer from the previous cluster, the positioning of the subclusters is determined. Then, the current clusters are replaced with the new clusters and for each new subcluster all the above operations are performed again. In the end, the leaves of this tree are obtained, which are the created gaze points. In case no time has been used in the data, the time is added linearly from the beginning to the end. If time has been given in the data, the data is sorted based on the created gaze points and normalized from zero to one.

4 EVALUATION

Table 1. The results of our approach in combination with others to improve the classification accuracy of participants and stimuli on two public data sets. Abbreviations: NN is neural network, ET is ensemble of bagged trees.

					ETRA2019Challenge Data [32, 35]		Hollywood Data [9]	
	Training Data				Accuracy		Accuracy	
Method	Sim [22]	VAE [20]	Proposed	Real Data	Stimulus	Participant	Stimulus	Participant
Encodji [16]					52%	23%	35%	23%
NN&Heatmap [21]	X				60%	29%	31%	28%
ET&Heatmap [21]					61%	29%	30%	31%
Encodji [16]					51%	24%	36%	22%
NN&Heatmap [21]		X			61%	28%	37%	34%
ET&Heatmap [21]					60%	31%	38%	33%
Encodji [16]					58%	28%	39%	37%
NN&Heatmap [21]			X		64%	35%	40%	35%
ET&Heatmap [21]					62%	37%	42%	33%
Encodji [16]					89%	81%	73%	67%
NN&Heatmap [21]				X	81%	73%	59%	41%
ET&Heatmap [21]					82%	77%	62%	51%
Encodji [16]					61%	43%	41%	36%
NN&Heatmap [21]	X	X	X		59%	34%	38%	37%
ET&Heatmap [21]					60%	37%	38%	38%
Encodji [16]					90%	82%	75%	71%
NN&Heatmap [21]	X			X	83%	73%	70%	43%
ET&Heatmap [21]					88%	77%	77%	51%
Encodji [16]					91%	81%	78%	70%
NN&Heatmap [21]		X		X	83%	74%	73%	48%
ET&Heatmap [21]					87%	78%	76%	55%
Encodji [16]					91%	83%	79%	74%
NN&Heatmap [21]			X	X	84%	76%	73%	57%
ET&Heatmap [21]					88%	81%	77%	62%
Encodji [16]					92%	85%	81%	76%
NN&Heatmap [21]	X	X	X	X	86%	79%	76%	58%
ET&Heatmap [21]					89%	83%	78%	63%
Chance					25.0%	12.5%	0.48%	1.58%

Table 2. The results of different data generation algorithms for fake scan path generation. Abbreviations: NN is neural network, ET is ensemble of bagged trees, and HOV is histogram of oriented gradients.

		Successful deception		
Data	Method	Sim [22]	VAE [20]	Proposed
Challenge [32, 35]	Encodji [16]	32%	41%	49%
	NN&Heatmap [21]	53%	69%	75%
	NN&HOV [18]	57%	72%	79%
	ET&Heatmap [21]	44%	61%	71%
	ET&HOV [18]	44%	62%	73%
Hollywood [9]	Encodji [16]	4%	19%	31%
	NN&Heatmap [21]	11%	37%	57%
	NN&HOV [18]	13%	42%	62%
	ET&Heatmap [21]	7%	33%	49%
	ET&HOV [18]	10%	35%	55%

Our scan path generator and model creator are implemented in Matlab. The scripts are provided online and everybody can access them via the link in the abstract. For the evaluation we used the implemented machine learning approaches in Matlab as well as we used Matlab to compute the Encodji [16], heatmap [21], and histogram of oriented velocities (HOV) [18] input format. The hardware for our evaluation was a desktop PC with an AMD Ryzen 9 3950X 16-Core Processor and 64 GB DDR4 ram. We did not use a GPU, since all models are small enough to train and execute them fast on a CPU. We conducted two experiments. The first is about improving classification accuracy with simulated data, and the second is about generating scan paths that pretend to be from a person. Alle evaluations are with additional data augmentation. The augmentations used is random scan path cropping, adding random noise to the gaze points, combining random scan path of the same class, and adding up to 5% of random gaze points. For our evaluation, we performed a 5-fold cross validation. For the ensemble of bagged decision trees (ET), we performed an automatic optimal parameter search from Matlab. For the neural networks (NN), we used a hidden layer with 100 neurons and determined the optimal training parameters via a grid search. For the Encodeji procedure, we used the parameters and the model from the paper [16].

The VAE [20] simulator was trained on the training data for each class and random number vectors were used to generate 1000 simulated gaze data for each class. We used the same procedure for our approach whereby data augmentation was used during training for the VAE as well as our approach. In the case of the [22] (Sim), scan paths were generated using the gaze data and the stimulus. For each procedure, 1000 scan paths were generated for each class and used as additional training data in the first experiment. In the second experiment, the generated gaze data was used to fool the machine learning methods.

The ETRA2019Challenge data [32, 35] consists of 960 recordings with 45 seconds duration each. The data was collected from 8 subjects. The performed tasks are fixation, visual search and visual exploration. The eye tracking device used is an EyeLink II and during recording the subjects had to use a chin-rest to increase the accuracy of the eye tracker. In total, there were 4 different stimuli used. In our experiment, we use the stimulus and subject classification.

The Hollywood data [9] provides eye tracking data of 63 subjects watching Hollywood video clips. Each clip had a length of 30 seconds and in total there are 206 different video clips. Each subject watched $\approx$ 40 clips. The used Eye Tracker was an Eyelink 1000. In addition to the subject and stimuli information, there is also a rating given for each clip, as well as if the movie was seen before. In this work, we use the stimulus as well as the subject for our classification experiments.

Looking at Table 1 it becomes clear that none of the simulators alone comes close to the results of the real data. If all simulators are used together, the individual results improve, but the real data are still significantly better. If the data of the simulators are used together with the real data, the best results are obtained. Our approach provides the largest contribution, but it is not drastically different from the other simulators. The best results are obtained when all generated data and the real data are used together for training. Here there is a significant improvement compared to the results obtained with the real data only.

In our machine learning deception experiment (table 2), we additionally used histograms of oriented velocities [18]. These do not use spatial position, only orientations. This makes them perform even worse than the heatmap features in the pure classification experiment. As seen in table 2, it is noticeable that the methods based on the heatmap and HOV features are the easiest to fool. This is due to the fact that these methods quantize the data. In the case of Encodji coding, the deception becomes much more difficult. Overall, our method performs best in this experiment, which is most evident in the Hollywood dataset.

5 LIMITATIONS

The limitations of our algorithm are on the one hand that it is not necessarily the case that real human behavior is simulated. This is due to the fact that our algorithm learns the hierarchy of scan paths from humans and also their distribution, but combines this later randomly. The problem here is that individual parts of the generated scan path are definitely human, but the combination does not necessarily reflect human behavior. Also, nowadays, it is not possible to prove or determine what exactly in a human scan path are the characteristic features of a person and which are not. Another limitation of this work is the evaluation regarding the faking of human scan paths (Table 2). In this evaluation it can be evaluated how many similar scan paths our simulator has generated, which have faked the machine learning approach, but not whether these are good simulated human scan paths regarding the human behavior.

6 CONCLUSION

In this paper, we have presented a new fully data-based scan path generation algorithm that can learn from human scan paths. In the first step, the algorithm generates a hierarchy using K-Means clustering and then computes the principal components using the distribution of the next clusters. During generation, the hierarchy is processed, and new clusters are generated in each generated layer based on the principal components. This combines the behavior of from the learned data so that all parts of the generated scan path are human, but the combination does not necessarily result in a human scan path. In our evaluation, we looked at combining different scan path generators as well as real data to train scan path classifiers. Here we could show that the presented generator is at least as good as the state of the art. In combination with the generated data of the other generators and the real data, the best result was obtained, which means that our approach extends the previous possibilities to obtain training data. Furthermore, we conducted experiments to simulate human scan paths to fool a machine learning procedure. Here our method performed best. Overall, we hope that the presented approach is a helpful contribution to the state of the art and helps further researchers to improve scan path classification as well as scan path generation.

REFERENCES

[1] Omer Arslan, Oguz Atik, and Serkan Kahraman. 2021. Eye tracking in usability of electronic chart display and information system. *The Journal of Navigation* 74, 3 (2021), 594–604.

[2] Marc Assens, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E O'Connor. 2018. PathGAN: visual scanpath prediction with generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 0–0.

[3] Louise Gillian C Bautista and Prospero C Naval. 2021. Gazemae: general representations of eye movements using a micro-macro autoencoder. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 7004–7011.

[4] Svetlana Bialkova, Klaus G Grunert, and Hans van Trijp. 2020. From desktop to supermarket shelf: Eye-tracking exploration on consumer attention and choice. *Food Quality and Preference* 81 (2020), 103839.

[5] Tad T Brunyé, Trafton Drew, Kathleen F Kerr, Hannah Shucard, Donald L Weaver, and Joann G Elmore. 2020. Eye tracking reveals expertise-related differences in the time-course of medical image inspection and diagnosis. *Journal of Medical Imaging* 7, 5 (2020), 051203.

[6] Daniel J Campbell, Joseph Chang, Katarzyna Chawarska, and Frederick Shic. 2014. Saliency-based bayesian modeling of dynamic viewing of static scenes. In *Eye Tracking Research and Applications*. ACM, 51–58.

[7] Margot Chauliac, Leen Catrysse, David Gijbels, and Vincent Donche. 2020. It Is All in the" Surv-Eye": Can Eye Tracking Data Shed Light on the Internal Consistency in Self-Report Questionnaires on Cognitive Processing Strategies?. *Frontline Learning Research* 8, 3 (2020), 26–39.

[8] Timothy F Cootes and Christopher J Taylor. 1992. Active shape models—'smart snakes'. In *BMVC92*. Springer, 266–275.

[9] Francisco M Costela and Russell L Woods. 2019. A free database of eye movements watching "Hollywood" videoclips. *Data in brief* 25 (2019), 103991.

[10] Rebecca Davis and Alla Sikorskii. 2020. Eye tracking analysis of visual cues during wayfinding in early stage Alzheimer's disease. *Dementia and geriatric cognitive disorders* 49, 1 (2020), 91–97.

[11] Andrew Duchowski, Sophie Jörg, Aubrey Lawson, Takumi Bolte, Lech Świrski, and Krzysztof Krejtz. 2015. Eye movement synthesis with 1/f pink noise. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*. ACM, 47–56.

[12] Andrew T Duchowski and Andrew T Duchowski. 2017. *Eye tracking methodology: Theory and practice*. Springer.

[13] Andrew T Duchowski and Sophie Jörg. 2015. Modeling Physiologically Plausible Eye Rotations. *Proceedings of Computer Graphics International* (2015).

[14] Andrew T Duchowski, Sophie Jörg, Tyler N Allen, Ioannis Giannopoulos, and Krzysztof Krejtz. 2016. Eye movement synthesis. In *Eye Tracking Research and Applications*. ACM, 147–154.

[15] John R Economides, Daniel L Adams, Cristina M Jocson, and Jonathan C Horton. 2007. Ocular motor behavior in macaques with surgical exotropia. *Journal of neurophysiology* 98, 6 (2007), 3411–3422.

[16] Wolfgang Fuhl, Efe Bozkir, Benedikt Hosp, Nora Castner, David Geisler, Thiago C Santini, and Enkelejda Kasneci. 2019. Encodji: encoding gaze data into emoji space for an amusing scanpath classification approach. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. 1–4.

[17] Wolfgang Fuhl, Efe Bozkir, and Enkelejda Kasneci. 2021. Reinforcement learning for the privacy preservation and manipulation of eye tracking data. In *International Conference on Artificial Neural Networks*. Springer, 595–607.

[18] W. Fuhl, N. Castner, and E. Kasneci. 2018. Histogram of oriented velocities for eye movement detection. In *International Conference on Multimodal Interaction Workshops, ICMIW*.

[19] Wolfgang Fuhl and Enkelejda Kasneci. 2018. Eye movement velocity and gaze data generator for evaluation, robustness testing and assess of eye tracking software and visualization tools. *arXiv preprint arXiv:1808.09296* (2018).

[20] Wolfgang Fuhl, Yao Rong, and Enkelejda Kasneci. 2021. Fully convolutional neural networks for raw eye tracking data segmentation, generation, and reconstruction. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 142–149.

[21] W Fuhl, N Sanamrad, and E Kasneci. 2021. The Gaze and Mouse Signal as additional Source for User Fingerprints in Browser Applications. *arXiv preprint arXiv:2101.03793* (01 2021).

[22] W. Fuhl, T. Santini, T. Kuebler, N. Castner, W. Rosenstiel, and E. Kasneci. 2018. Eye movement simulation and detector creation to reduce laborious parameter adjustments. *arXiv preprint arXiv:1804.00970* (2018).

[23] Aaron L Gardony, Robert W Lindeman, and Tad T Brunyé. 2020. Eye-tracking for human-centered mixed reality: promises and challenges. In *Optical Architectures for Displays and Sensing in Augmented, Virtual, and Mixed Reality (AR, VR, MR)*, Vol. 11310. International Society for Optics and Photonics, 113100T.

[24] Patrick Jermann, Marc-Antoine Nüssli, and Weifeng Li. 2010. Using dual eye-tracking to unveil coordination and expertise in collaborative Tetris. *Proceedings of HCI 2010 24* (2010), 36–44.

[25] Antony William Joseph and Ramaswamy Murugesh. 2020. Potential eye tracking metrics and indicators to measure cognitive load in human-computer interaction research. *Journal of Scientific Research* 64, 1 (2020).

[26] Binh H Le, Xiaohan Ma, and Zhigang Deng. 2012. Live speech driven head-and-eye motion generators. *IEEE transactions on Visualization and Computer Graphics* 18, 11 (2012), 1902–1914.

[27] Sooha Park Lee, Jeremy B Badler, and Norman I Badler. 2002. Eyes alive. In *ACM Transactions on Graphics (TOG)*, Vol. 21. ACM, 637–644.

[28] Astar Lev, Yoram Braw, Tomer Elbaum, Michael Wagner, and Yuri Rassovsky. 2020. Eye Tracking During a Continuous Performance Test: Utility for Assessing ADHD Patients. *Journal of Attention Disorders* (2020), 1087054720972786.

[29] Dirk Lewandowski and Yvonne Kammerer. 2020. Factors influencing viewing behaviour on search engine results pages: A review of eye-tracking research. *Behaviour & Information Technology* (2020), 1–31.

[30] Congcong Liu, Yuying Chen, Lei Tai, Haoyang Ye, Ming Liu, and Bertram E Shi. 2019. A gaze model improves autonomous driving. In *Proceedings of the 11th ACM symposium on eye tracking research & applications*. 1–5.

[31] Xiaohan Ma and Zhigang Deng. 2009. Natural eye motion synthesis by modeling gaze-head coupling. In *IEEE Virtual Reality Conference*. IEEE, 143–150.

[32] Michael B McCamy, Jorge Otero-Millan, Leandro Luigi Di Stasi, Stephen L Macknik, and Susana Martinez-Conde. 2014. Highly informative natural scene regions increase microsaccade production during visual scanning. *Journal of neuroscience* 34, 8 (2014), 2956–2966.

[33] Xiaoxu Meng, Ruofei Du, and Amitabh Varshney. 2020. Eye-dominance-guided foveated rendering. *IEEE transactions on visualization and computer graphics* 26, 5 (2020), 1972–1980.

[34] Nisha Nesaratnam, Peter Thomas, and Anthony Vivian. 2017. Stepping into the virtual unknown: feasibility study of a virtual reality-based test of ocular misalignment. *Eye* 31, 10 (2017), 1503–1506.

[35] Jorge Otero-Millan, Xoana G Troncoso, Stephen L Macknik, Ignacio Serrano-Pedraza, and Susana Martinez-Conde. 2008. Saccades and microsaccades during visual fixation, exploration, and search: foundations for a common saccadic generator. *Journal of vision* 8, 14 (2008), 21–21.

[36] Derek Panchuk, Samuel Vine, and Joan N Vickers. 2015. Eye tracking methods in sport expertise. In *Routledge handbook of sport expertise*. Routledge, 176–187.

[37] Ivanna M Pavisic, Yoni Pertzov, Jennifer M Nicholas, Antoinette O'Connor, Kirsty Lu, Keir XX Yong, Masud Husain, Nick C Fox, and Sebastian J Crutch. 2021. Eye-tracking indices of impaired encoding of visual short-term memory in familial Alzheimer's disease. *Scientific reports* 11, 1 (2021), 1–14.

[38] Tomislav Pejsa, Bilge Mutlu, and Michael Gleicher. 2013. Stylized and performative gaze for character animation. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 143–152.

[39] Adil Rasheed, Omer San, and Trond Kvamsdal. 2020. Digital twin: Values, challenges and enablers from a modeling perspective. *Ieee Access* 8 (2020), 21980–22012.

[40] Juliane Richter, Katharina Scheiter, Thérése Felicitas Eder, Fabian Huettig, and Constanze Keutel. 2020. How massed practice improves visual expertise in reading panoramic radiographs in dental students: An eye tracking study. *PloS one* 15, 12 (2020), e0243060.

[41] Yumiko Shinohara, Rebecca Currano, Wendy Ju, and Yukiko Nishizaki. 2017. Visual attention during simulated autonomous driving in the US and Japan. In *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. 144–153.

[42] Maria Sielicka-Różyńska, Ewa Jerzyk, and Natalia Gluza. 2021. Consumer perception of packaging: An eye-tracking study of gluten-free cookies. *International Journal of Consumer Studies* 45, 1 (2021), 14–27.

[43] Daniel Simon, Srinivas Sridharan, Shagan Sah, Raymond Ptucha, Chris Kanan, and Reynold Bailey. 2016. Automatic scanpath generation with deep recurrent neural networks. In *Proceedings of the ACM Symposium on Applied Perception*. ACM, 130–130.

[44] Sarah Snell, Daniel Bontempo, Gregory Celine, and Robert Anthonappa. 2020. Assessment of medical practitioners' knowledge about paediatric oral diagnosis and gaze patterns using eye tracking technology. *International Journal of Paediatric Dentistry* (2020).

[45] Douglas Tweed, Werner Cadera, and Tutis Vilis. 1990. Computing three-dimensional eye position quaternions and eye velocity from search coil signals. *Vision research* 30, 1 (1990), 97–110.

[46] Mireia Vilardell, Maria Buxó, Ramon Clèries, José Miguel Martínez, Gemma Garcia, Alberto Ameijide, Rebeca Font, Sergi Civit, Rafael Marcos-Gragera, Maria Loreto Vilardell, et al. 2020. Missing data imputation and synthetic data simulation through modeling graphical probabilistic dependencies between variables (ModGraProDep): An application to breast cancer survival. *Artificial intelligence in medicine* 107 (2020), 101875.

[47] David R Walton, Rafael Kuffner Dos Anjos, Sebastian Friston, David Swapp, Kaan Akşit, Anthony Steed, and Tobias Ritschel. 2021. Beyond blur: Real-time ventral metamers for foveated rendering. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–14.

[48] Erroll Wood, Tadas Baltrusaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson, and Andreas Bulling. 2015. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*. 3756–3764.