Generating Realistic Synthetic Head Rotation Data for Extended Reality using Deep Learning

Jakob Struye
jakob.struye@uantwerpen.be
University of Antwerp - imec
Antwerp, Belgium

Filip Lemic
filip.lemic@upc.edu
Universitat Politècnica de Catalunya
Barcelona, Spain

Jeroen Famaey
jeroen.famaey@uantwerpen.be
University of Antwerp - imec
Antwerp, Belgium

ABSTRACT

Extended Reality is a revolutionary method of delivering multimedia content to users. A large contributor to its popularity is the sense of immersion and interactivity enabled by having real-world motion reflected in the virtual experience accurately and immediately. This user motion, mainly caused by head rotations, induces several technical challenges. For instance, which content is generated and transmitted depends heavily on where the user is looking. Seamless systems, taking user motion into account proactively, will therefore require accurate predictions of upcoming rotations. Training and evaluating such predictors requires vast amounts of orientational input data, which is expensive to gather, as it requires human test subjects. A more feasible approach is to gather a modest dataset through test subjects, and then extend it to a more sizeable set using synthetic data generation methods. In this work, we present a head rotation time series generator based on TimeGAN, an extension of the well-known Generative Adversarial Network, designed specifically for generating time series. This approach is able to extend a dataset of head rotations with new samples closely matching the distribution of the measured time series.

CCS CONCEPTS

• **Mathematics of computing** → **Time series analysis**; • **Computing methodologies** → **Adversarial learning**; • **Human-centered computing** → *Virtual reality*.

KEYWORDS

Synthetic data, Data Generation, Extended Reality, Generative Adversarial Networks

ACM Reference Format:
Jakob Struye, Filip Lemic, and Jeroen Famaey. 2022. Generating Realistic Synthetic Head Rotation Data for Extended Reality using Deep Learning. In *Proceedings of ACM International Conference on Multimedia (ACMMM '22).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACMMM '22, October 10–14, 2022, Lisbon, Portugal

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Extended Reality (XR), encompassing Virtual, Mixed and Augmented Reality, has proven to be a major revolution in media consumption. In addition to its widespread use for recreational purposes [30], XR has enabled novel approaches for other tasks, including training [21], remote operation [28], and architecture and construction [2, 11]. A key enabler of the XR experience is how it can reflect the user's real-world motion accurately and immediately within the experience [47]. This enables the user to seamlessly and intuitively change their gaze direction, and can also serve as a source of input for virtual experiences.

The user freedom in XR induces a number of challenging demands on the system. When the user rotates their head, the displayed content must adapt to this at a moment's notice. More specifically, the *motion-to-photon latency* dictates that the effect of any user motion must be visible on-screen within 20 ms as to avoid nauseating the user [15]. Several algorithms aid in fulfilling this latency requirement. Generated content is often warped right before display through algorithms such as Asynchronous Time-Warp, using the most recent measurements of user pose [38]. When displaying pre-recorded 360° content, viewport-dependent encoding ensures that only the content expected to be within the user's field of view is transmitted to reduce transmission latency [9, 20]. Furthermore, video away from the user's expected centre of gaze may be encoded at lower quality, further reducing data size [25]. Overall, algorithms aiming at satisfying the motion-to-photon latency often include Deep Learning components converting users' orientational data to useful outputs, such as how to compress visual data. Training and testing these Deep Learning algorithms is a notoriously data-hungry process [6, 14]. Furthermore, extensive evaluation of full algorithms again requires massive amounts of orientational data.

While the aforementioned algorithms are the most ubiquitous consumers of orientational data in the field of XR, needs for substantial orientational data sources arise in other situations as well. For truly wireless interactive XR, where content is generated off-device and streamed in real-time over the air, frequencies in the millimetre-wave band (30 GHz to 300 GHz) or higher are needed to stream content in extremely high quality [15, 34]. To guarantee sufficient signal strength in these frequency ranges, communication must be *beamformed* between the sender and receiver, rather than being sent and received omnidirectionally [39]. In addition, in the field of Redirected Walking, non-deterministic mappings between real-world motion and virtual motion are applied to avoid real-world collisions without restricting virtual freedom [33]. High-performance solutions in both of these cases again require enormous amounts of orientational data for training and evaluation.

Over the past years, many datasets of XR orientational measurements have been published. Commonly, these datasets consist of logs of timestamped orientations, measured at a regular interval and represented in the yaw-pitch-roll format. In this format, an orientation is deconstructed into three subsequent rotations starting from a reference orientation: yaw represents turning one's head left or right, pitch represents tilting up or down, and roll represents tilting sideways. Collecting these datasets usually involves tens to hundreds of test subjects, who are each shown several minutes to several tens of minutes of XR content [10, 12, 17, 19, 23, 24, 29, 40, 41]. Clearly, gathering these datasets is an expensive and labour-intensive process that does not scale well. Therefore, a clearly more efficient approach is to apply synthetic data generation techniques to augment existing datasets with new, unique samples, without changing the distribution of the overall dataset [31]. Despite this, research into this approach has so far been very limited, with only an exploratory work proposing data generation through Fourier transforms [5]. This approach considers time series of orientations as signals, which are converted to power spectral densities, after which the *mean* power spectral density is modeled. Then, perturbed versions of this model are converted back to signals and finally orientational time series. This however results in synthetic time series that closely match the *mean* of the set of input time series, rather than their full *distribution*. In contrast, we propose to use a significantly more capable method of synthetic data generation, namely the Generative Adversarial Network (GAN) [18]. A GAN consists of two sub-systems trained in parallel. A *generator* generates synthetic samples, while a *discriminator* attempts to classify samples as real or synthetic. In a zero-sum game, both sub-systems interactively improve their performance: the discriminator discovers features indicative of synthetic samples, while the generator learns to avoid introducing such features. Ideally, the generator eventually outputs unique samples indistinguishable from the real ones. As each sample within a dataset of orientational data is a *time series*, credible synthetic samples must not only match the original distribution when observing individual time steps, but also when observing their evolution through time. A modification to GANs called *TimeGAN* aims to satisfy this requirement [43]. Hence, in this work, we rely on TimeGAN to generate realistic synthetic orientational data samples. During training, the TimeGAN is provided sequences of orientational data, such that it eventually learns to generate similar, but previously unseen sequences. We repeat this process with multiple datasets to show the approach works generally. In this work, we only apply TimeGAN to orientational data, and not positional data, as many applications, including beamforming, viewport-dependent encoding and redirected walking, rely mostly on orientational data. Positional data, being significantly less dynamic, has only a limited impact in these applications [35, 46].

To gauge the utility of these synthetic datasets, one needs a metric of how similar the distributions of the respectively real and synthetic datasets are. Several general-purpose metrics are commonly used for this purpose, such as Principal Component Analysis (PCA) [8], t-distributed Stochastic Neighbor Embedding (t-SNE) [37] and Train on Synthetic, Test on Real (TSTR) [16]. These metrics are however all difficult to interpret intuitively. It is unclear when these metrics indicate a "realistic" synthetic dataset in absolute terms, meaning their main use is in comparing different sources of synthetic data. Fortunately, the orientational data considered in this work is, by itself, easily interpreted intuitively. As such, we opt to forego the more generally used metrics described above, and instead define a number of metrics specific to head rotation data, which together characterise the important features of the dataset. Specifically, or set of metrics considers not only the distribution of orientations, but also how often and how smoothly users rotate. We consider this to be a more convincing indication of our approach's effectiveness.

Our work contains the following contributions:

- We present the first Deep Learning approach for generating realistic synthetic datasets of XR head rotations capable of extending any dataset with minimal expert input.
- We outline a number of metrics intuitively characterising such datasets, allowing for interpretable estimation of the practical similarity of real and synthetic datasets.
- We train the model on two different datasets, and evaluate the output using the above metrics, showing that it can generate realistic datasets, outperforming the current state of the art.

The remainder of this paper is structured as follows. Section 2 covers related work. In Section 3, we outline our approach and how to quantify its performance. This performance is then evaluated in Section 4. Finally, Section 5 concludes this work.

2 RELATED WORK

While the intersection of orientational data collection and synthetic data generation is still in its infancy, the two fields separately are well-developed. This section provides an overview of the two.

2.1 XR pose datasets

Over the years, a wide array of datasets containing orientations or poses (i.e., locations plus orientations) of XR users have been made available to the community. In this overview, we only consider works presenting novel datasets (i.e., not compiled from previous works) at a reasonably high sampling rate, which are, at the time of writing, readily available online.

Corbillon *et al.* present an orientational dataset sampled at 45 Hz gathered from 59 subjects shown 6 minutes of 360° video [12]. Lo *et al.* collected a 30 Hz dataset containing orientations along with saliency maps, identifying objects that attract attention, using 10 minutes of 360° video shown to 50 subjects [24]. The set of videos contains both recorded and pre-generated content, further subdivided into slow-paced and fast-paced content. Li *et al.* showed 221 test subjects a 1 to 2 minute sequence of videos, repeated 10 times, gathering orientation and gaze direction at 48 to 60 Hz [23]. They then used this as input to a Deep Learning model predicting perceived visual quality. Next, a dataset by Wu *et al.* contains the full pose at 100 Hz for 48 users exposed to nearly 90 minutes of video [40]. For the AVtrack orientational dataset, Fremerey *et al.* showed 10 minutes of video to 48 subjects [17]. With only 10 Hz and an angular precision of 1°, this dataset is tailored more towards investigating longer-term behaviour of subjects. In Nasrabadi *et al.*'s dataset, orientation was measured at 60 Hz from 14 minutes of video shown to 60 subjects [29]. Xu *et al.* showed 35 minutes of video to 58 subjects, recording orientation and gaze at 60 Hz [41]. Next, Hu *et al.* recorded orientation and gaze at 100 Hz with 30

subjects [19]. Each was shown 7.5 minutes of video four times, each time with a separate task, then a neural network was trained to classify the measurements according to the performed task. Zerman *et al.* showed one to a few minutes of volumetric video to 20 test subjects, logging location and orientation at 55 Hz [45]. Subramanyam *et al.* showed four point cloud videos to 26 subjects for varying lengths of time, recording location and orientation at 30 Hz [36]. Finally, contrary to the previous datasets, Chakareski *et al.* used a navigable virtual experience in which three test subjects were allowed to navigate freely for six 2 minute sessions, during which the full pose was gathered at 250 Hz [10].

Overall, while datasets vary in terms of number of subjects, sampling rate and sample length, most employ pre-recorded or pre-generated video, as opposed to an interactive environment. Datasets with truly rapid motion (e.g., from a fast-paced video game) are, to the best of our knowledge, currently non-existent. These are crucial for several applications. For example, real-time beamforming becomes inherently more challenging under rapid motion, meaning that such datasets are needed to evaluate beamforming solutions in a worst-case environment [35, 46].

2.2 Time Series Generation

Once a reasonable amount of data is gathered using test subjects, generating synthetic, but realistic data may be necessary to obtain a sufficiently large dataset for some application. For this, some approaches have been proposed. One such approach is the classical Smith's algorithm, designed for generating instantiations of a wireless channel model [32, 44]. In essence, it considers the time series to be generated as a set of signals. Known samples are converted to frequency space using the Fourier transform. Then, random noise sequences are weighted using filter coefficients, resulting in frequency coefficients similar to those of the known samples. Converting back to the time domain using the Inverse Fourier transform results in realistic synthetic signals. Recently, Blandino *et al.* generated synthetic head rotation traces using this approach [5]. However, they only considered the *mean* power spectral density in the model, meaning the distribution *between* samples is lost. We will compare our solution to this Fourier-based solution using the authors' publicly available code [1].

Data generation has also seen significant attention from the Deep Learning community, where the GAN is generally considered to be the prime candidate [18]. Recently, Martin *et al.* applied this approach to scanpath (i.e., a sequence of gaze directions) generation, a field adjacent to head rotation generation [26]. As a regular GAN is not time series-aware, the authors added compatibility by using Dynamic Time Warping (DTW), a measure for similarity between time series, as a loss function. Other approaches for inserting general time series compatibility into a GAN model have been proposed [16, 27], with current state of the art being TimeGAN [43]. These approaches are covered extensively in the next section.

3 METHODOLOGY

This section outlines our methodology for generating realistic synthetic orientational time series using TimeGAN. We will illustrate our approach using the dataset from [10], but note that our approach is intended to be general. We first generally present the TimeGAN algorithm, and then apply it to head rotation data generation specifically.

3.1 TimeGAN

Classical approaches to data generation are often model-based, meaning that transferring the generator from one source dataset to another requires expert input, and may necessitate significant changes in design. As such, more general, model-free approaches are desirable. GANs are generally considered to be the prime candidate for this [18]. A GAN is a general design of a Deep Learning agent, where two sub-systems interact adversarially, to eventually generate samples matching the distribution of some source dataset. The *generator* receives random noise as inputs, and converts these to synthetic samples of the desired dimensions. The *discriminator* is a classifier, which, through supervised training, learns to classify samples as real (i.e., from the source dataset) or fake (i.e., from the generator). While the generator, which cannot access the source dataset, will initially output essentially random noise, it is given access to the loss function of the discriminator, meaning it can adapt its output to maximise that loss (i.e., make it more difficult for the discriminator to distinguish between fake and real). Interactively, the discriminator pushes the generator to produce more realistic samples, in turn encouraging the discriminator to discover more subtle differences between real and fake. The exact implementation of the two sub-systems depends on the type of data to generate.

In arguably the most well-known application of GANs, generating fake images, these are constructed using Convolutional Neural Networks [22]. When samples are time series however, extra care must be taken to ensure that the time-correlation between different time points *within* a time series is maintained. To this end, TimeGAN introduces significant augmentations to the GAN system [43]. Specifically, TimeGAN's discriminator and generator consist of Gated Recurrent Units (GRUs), a component capable of considering time-dependencies. Furthermore, *embedder* and *recovery* sub-systems are added, which respectively encode a time series to a latent space of lower dimension, and decode this latent space back to the original space. These two are trained first, and then the generator produces samples in this latent space, to be converted to time series through recovery. This reduces the complexity of the generator's task to a more practically feasible level. Finally, a supervised learning step is introduced. In this step, the generator is made to complete (latent representations of) incomplete time series from the source dataset. A loss function, measuring the distance between the generated time steps and the actual time steps from the source dataset, further encourages the generator to learn the time-correlation within time series. Regular adversarial learning and this supervised learning are performed alternatingly, and their loss functions are implemented as cross-entropy loss and mean squared error, respectively. The system is summarised in Figure 1. TimeGAN was shown to outperform earlier GAN-based approaches on financial and energy datasets in its initial presentation, and has since been applied successfully to medical data [13]. In this paper, we use the TimeGAN approach based on the initial authors' code[2].

[1] https://github.com/usnistgov/vrHeadRotation

[2] https://github.com/jsyoon0823/TimeGAN

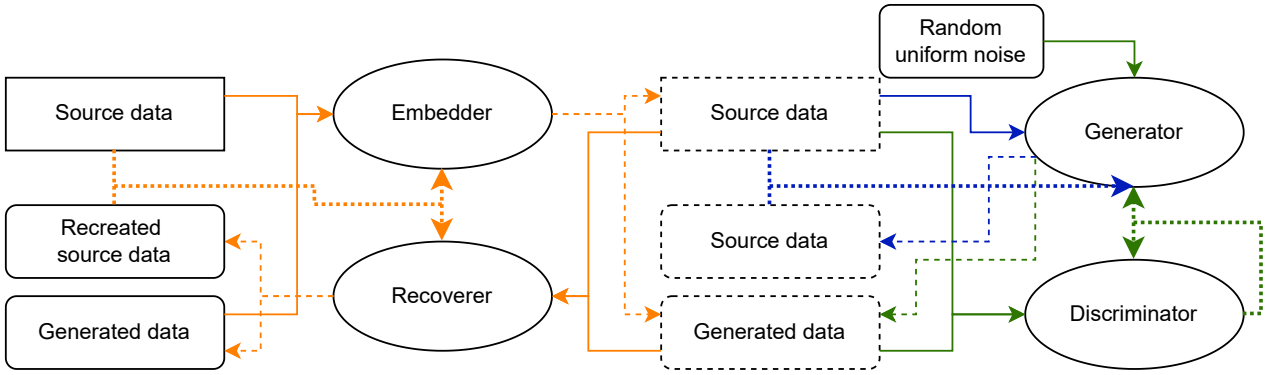


Figure 1: Illustration of the TimeGAN training process, displaying neural networks (ellipses) and data (rectangles). Rectangles with rounded corners indicate data generated by the model, while those with sharp corners represent data taken from input sequences. Each line is part of training the latent space (orange), unsupervised adversarial training (green) or supervised training (blue). Solid lines are data inputs, dashed lines are data outputs and dotted lines are loss signals. Losses originating from two data sets are a dissimilarity measure (e.g., mean squared error) while the loss originating from the discriminator is a classification loss (e.g., cross-entropy).

3.2 Data Inspection and Preparation

We mainly evaluate our approach using the dataset in [10]. We select this dataset to enable direct comparison with [5], based on the same dataset. As this iteration of our work focuses on orientational data, we disregard the positional data in the dataset entirely. Remember that the dataset contains 18 orientational traces of 2 minutes each, performed by 3 test subjects, sampled at 250 Hz, provided in yaw-pitch-roll format. We maintain this representation as it is easily interpretable. Figure 2 shows the Probability Density Function (PDF) of the yaw, pitch and roll, quantized into buckets 10° wide, with the middle bucket centered around 0°. Clearly, yaw motion is significantly more pronounced. This is unsurprising, as points of interest in a virtual world are usually distributed in roughly a horizontal plane around the user. Furthermore, turning around one's axis or looking to the side is more comfortable than looking up/down or tilting one's head. Because the virtual experience was an indoor environment, users' gaze was generally aimed at one of the walls, explaining the local maxima around −90°, 0° and 90°. While the normal distributions of the pitch and roll are easily generated by neural networks, the yaw's distribution may be more challenging on two fronts: its multiple peaks, along with discontinuities through time, whenever the representation rolls over between 180° and 180°. To combat the data's non-normality, we transform the data non-linearly using a quantile transformer, forcing a normal distribution [1, 3]. Experimentation showed this to be more effective than power transforms such as Box-Cox [7] and Yeo-Johnson [42] for this application. To resolve the discontinuities, we simply shift the remainder of a time series by 360° whenever a discontinuity occurs. While this means we can no longer ascertain the exact data range a priori, data does remain well within reasonable bounds in practice. We emphasise that all transformations are reversible, and that synthetic data is transformed to the original representation before comparison with the source dataset.

Next, we discuss how to arrange the time series into an appropriate format for synthetic generation. Originally, the data is divided into eighteen time series of 2 minutes at 250 Hz, resulting in 30 000 samples per time series. As the GAN's training time and difficulty scale with the input size, and Deep Learning traditionally requires many distinct samples, subdividing the samples is inevitable. In line with experiments in the original TimeGAN paper, we propose to subdivide each time series into smaller instances of 25 samples each, using a sliding window. As this would result in only 100 ms windows, we additionally downsample the data. We argue that downsampling does not significantly reduce the utility of the dataset. Intuitively, a human can only perform a few distinct head rotations per second, especially when impaired by XR hardware. Furthermore, the law of inertia implies that these motions will be relatively smooth, and therefore easily recreated accurately through simple interpolation techniques. For a more rigorous justification, we first refer to the power spectral density estimations of the data, presented in [5]. Energy is focused around the lowest frequencies, with over 90 % at 5 Hz and lower. As per the Shannon-Nyquist sampling theorem, this information will be maintained when downsampling to 10 Hz. Additionally, we investigate empirically how well the original data can be recreated after downsampling. We downsample the full dataset for different downsampling factors, then attempt to upsample back to the original frequency using a simple cubic spline interpolator. Figure 3 shows how well interpolation performs. Up to a downsampling factor of 25 (i.e., 10 Hz), the difference between the actual and interpolated values remains minimal, but increases rapidly with further downsampling. Based on the analysis above, we decide to downsample the dataset to 16.67 Hz, a downsampling factor of 15. This results in separate time series of 1.5 seconds each. Figure 4 shows one such sample, arbitrarily selected. We expect samples of this length to be sufficient for most applications, as prediction horizons for dynamic encoding and beamforming are usually in the order of 100 ms [4]. For cases where longer samples are desirable, one could fuse several samples together. We leave this for future work.

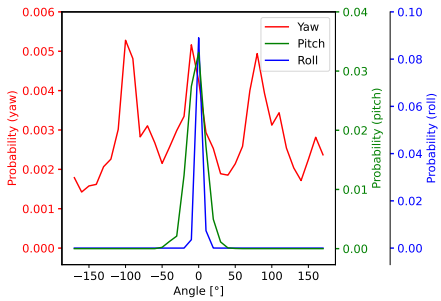


Figure 2: Distribution (quantized) of yaw, pitch and roll within the original dataset. Pitch is, by definition, restricted to $[-90, 90]$ degrees.

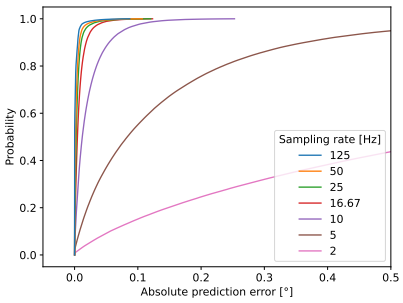


Figure 3: CDF of the absolute error between the original dataset, and the dataset first downsampled, then upsampled using a cubic spline interpolator. Errors over $0.5°$ are removed for clarity.

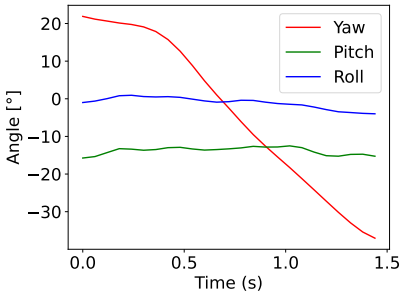


Figure 4: One sample extracted from the original dataset, after downsampling.

3.3 TimeGAN Tuning

The dataset, subdivided into 23 700 1.5 second samples of 25 time points each, is provided to the TimeGAN, where the quantile transformer is fit. For proper operation, the TimeGAN's hyperparameters have to be optimised. Analogously to TimeGAN's initial evaluation, we opt to use the same network topology for every sub-system in the TimeGAN. Table 1 summarises a well-performing set of hyperparameters. Note that we report *epochs* as number of full passes through the dataset, while the reference implementation uses *iterations*, each iteration being a single batch sampled from the dataset. Finally, we generate a large synthetic dataset, 10 times the size of the original, every 10 epochs. As GANs are notoriously challenging to train, and are known to degrade when trained for too long, this allows us to obtain a well-distributed result without having to fully optimise the epoch hyperparameter, or repeating the computationally expensive training process several times.

Table 1: TimeGAN Hyperparameters

Epochs	1250
Batch size	128
Learning rate	0.001
Neurons/layer	18
Layers	3

3.4 Evaluation Metrics

Once a synthetic dataset is generated, some way of evaluating how well its distribution matches the original dataset's is needed. Determining this both accurately and interpretably is notoriously difficult. Some commonly used approaches are dimensionality reduction through t-distributed Stochastic Neighbor Embedding (t-SNE) or Principal Component Analysis (PCA), enabling visual comparison. Another is Train on Synthetic, Test on Real (TSTR), where some neural network taking input data of the type under consideration is trained using only synthetic data, then evaluated using real data. With these approaches it is however difficult to determine what qualifies as "good enough", making them primarily useful when comparing data generators. For this case specifically, the dataset itself is fortunately easily interpretable. Therefore, we determine a number of metrics which together interpretably characterise the relevant features of the dataset. We propose the following metrics:

- **Orientation distribution**: separate PDFs of the yaw, pitch and roll will show whether the distribution between viewing directions is maintained. We consider every time point within one sample as a separate data point, rather than taking the mean of each sample, as to avoid masking information for discontinuous samples.
- **Per-sample motion distribution**: the distribution of the range (i.e., difference between maximum and minimum value) of yaw, pitch and roll shows whether time-correlation is maintained properly. The distribution of slower-moving and more rapid samples should ideally be maintained.
- **Autocorrelation of velocity**: The autocorrelation of the velocity (i.e., first derivative) of yaw, pitch or roll reveals whether the "smoothness" of the motion is maintained.
- **Cross-correlation of velocities**: The cross-correlation of velocities quantifies time-correlation of motion between different axes. Intuitively, "diagonal" motion (i.e., not on only one axis) should cause such time-correlation. Ideally, this should be maintained in synthetic data, but can only be expected to be maintained if the three features are not generated independently.

We will evaluate our approach using these metrics. In addition, we also generate PCA and t-SNE plots, to investigate whether differences in synthetic dataset quality revealed by the above metrics are also visible in these plots.

4 EVALUATION

To obtain a synthetic dataset using TimeGAN, we trained the system for the full 1250 epochs, generating a dataset every tenth epoch. We manually inspected each result using the described metrics and selected the best-performing option. We stress that this should not be considered as overfitting. When training a network for, say, prediction, such an approach is undesirable. This would overfit the

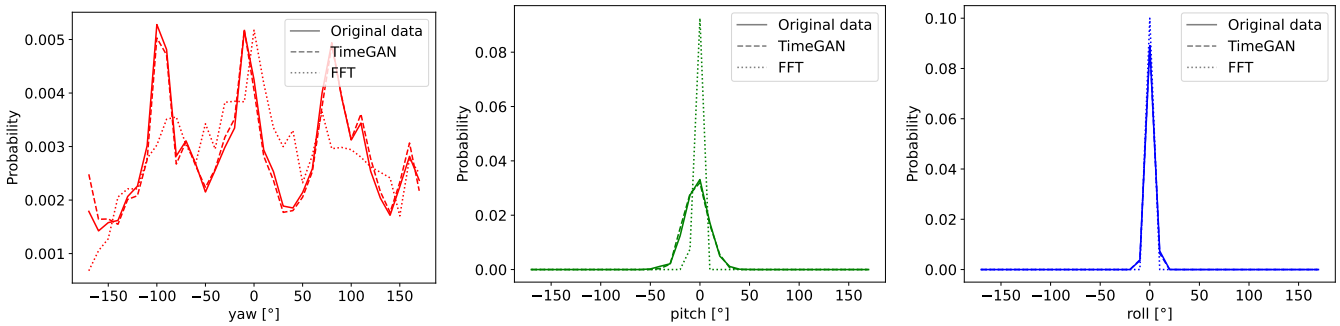


Figure 5: Distribution of yaw, pitch and roll values across all time steps of all samples

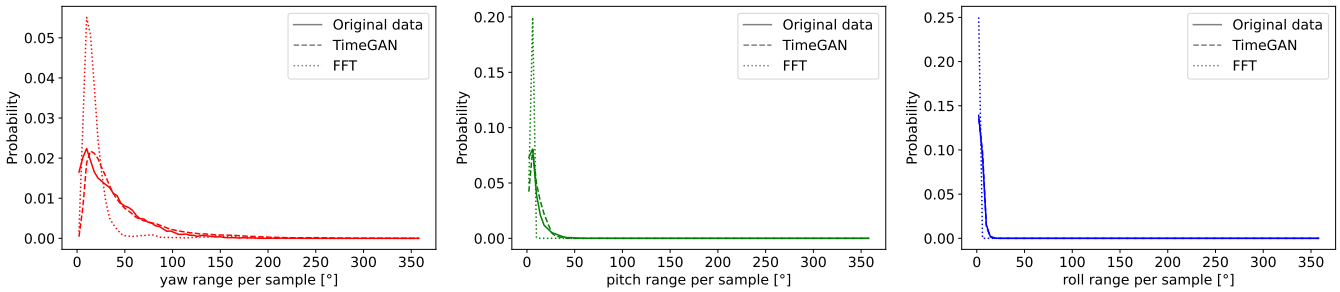


Figure 6: Distribution of the range of yaw, pitch and roll values per sample. Each range is calculated as the difference between the minimum and maximum value within the sample.

predictor to the evaluation dataset, which does not necessarily lead to a well-performing predictor for new data once the system is deployed. In this case however, the generated dataset *is* the final result of the system, and the system will not be presented with inputs of (slightly) different distributions in the wild, as the input is, by definition, uniformly distributed noise.

As a baseline, we also generate a synthetic dataset with the FFT-based approach from [5]. This generates time series of 30 000 time steps, analogous to the original data, which we then downsample and subdivide into shorter time series using a sliding window.

4.1 Head rotation metrics

We now evaluate each of the novel metrics. Note that, when plotting distributions, values are quantized into buckets 10° wide.

4.1.1 Orientation distribution. We first analyse the distribution of the raw yaw, pitch and roll values. We consider every time step of every sample as a separate data point and plot their distribution in Figure 5. The distribution of roll values is rather limited, and both synthetic datasets match it closely. Roll motion requires tilting one's head, which is rather uncomfortable. The pitch's distribution is slightly wider, and here the FFT fails to match the distribution closely, while the TimeGAN again closely matches the distribution. With the complicated yaw distribution, this occurs to an even greater extent. The three peaks are closely matched by the TimeGAN, while they are significantly less pronounced for the FFT, despite only the latter being hand-crafted to match this distribution.

4.1.2 Motion distribution. Figure 6 illustrates how well the motion of the synthetic dataset matches that of the original dataset. The FFT shows a major peak at very low motion, while the TimeGAN again closely matches the original dataset. We do note that, when considering yaw, *very-low-motion* samples are underrepresented in the TimeGAN dataset. We hypothesise that, when the motion distribution is broad, generating these very-low-motion time series is inherently challenging with the TimeGAN. The time series is generated per-step, and to form a very-low-motion time series, every next step's value must be very close to the previous. A noteworthy deviation for even a single time step increases the overall motion in the entire time series. Even if the probability for generating values close to the previous ones is reasonably high, the probability of this happening for *every* time step in the 25-step sequence will be low. Further evidence for this hypothesis is that when motion is consistently low, the TimeGAN easily generates this motion distribution accurately. Overall, we argue that this phenomenon is inevitable with the current approach, and leave potential solutions for future work. We do note that very-low-motion samples are less valuable. Applications such as viewport-dependent encoding or beamforming are mainly challenging under higher motion. We argue that failure to generate higher-motion samples, as occurs with FFT, is significantly more damaging to the data's utility. This failure occurs at least partially due to the FFT being designed to consistently match the distribution of the *mean* time series, rather than the distribution of all time series.

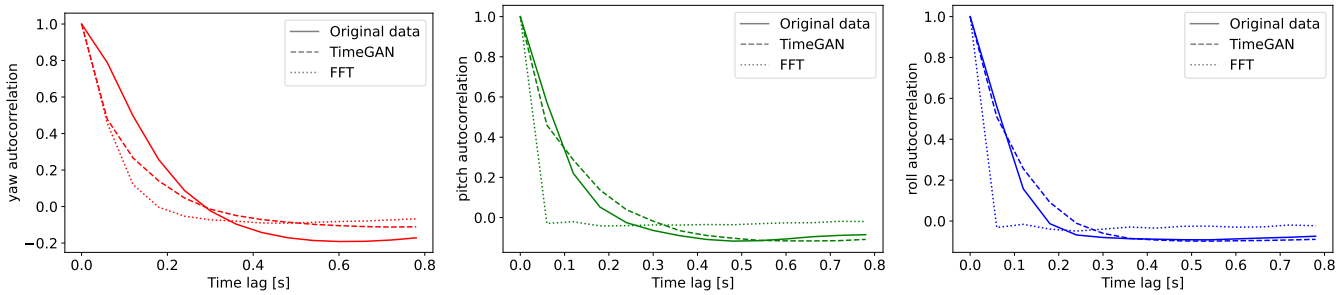


Figure 7: Mean autocorrelation of yaw, pitch and roll across all samples.

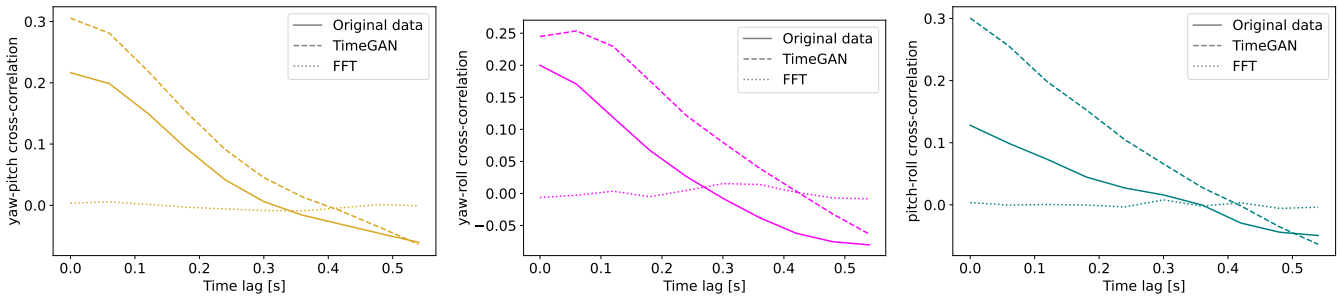


Figure 8: Mean cross-correlation of each combination of two features across all samples.

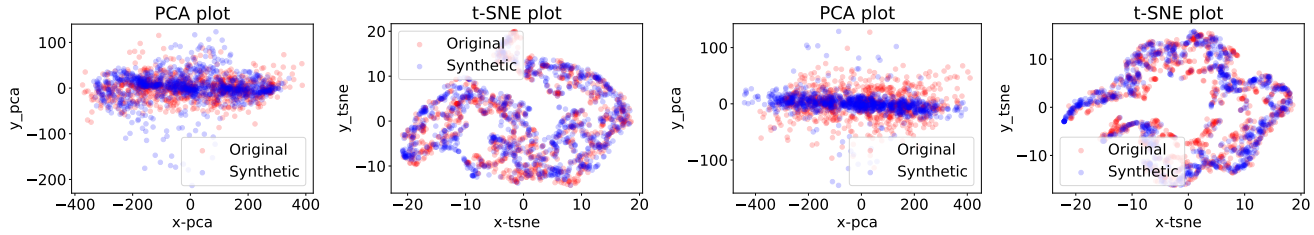
4.1.3 Autocorrelation of velocity. In a realistic time series, velocity changes gradually, largely due to the law of inertia. Intuitively, the autocorrelation, a measure of similarity between a time series and a time-lagged copy of itself, should decrease gradually as the time lag increases. This has been observed in previous work [4], and Figure 7 also clearly shows this behaviour for the original dataset. With the TimeGAN, the autocorrelation for pitch and roll is matched closely. With yaw, it decreases somewhat more rapidly, which is likely a side-effect of the lack of very-low-motion samples discussed above. For the FFT, the autocorrelation decreases significantly more rapidly, becoming near-zero earlier. One may expect autocorrelation to be high with the FFT, as the generated signals are of low frequency, making them inherently relatively smooth. We hypothesise that this result is at least in part due to the lower overall motion in that dataset, meaning the effect of minor perturbations is more noticeable in the (normalised) autocorrelation.

4.1.4 Cross-correlation of velocity. Intuitively, one would expect the yaw, pitch and roll to display at least some correlation, as humans do not naturally only rotate their heads along one axis at a time. Any "diagonal" motion results in motion on multiple axes, while holding one's head still results in no motion on each axis. Indeed, Figure 8 confirms this intuition: the high correlation at 0 time lags shows that high motion along one axis implies a high likelihood of high motion along another. As the number of time lags increases, the correlation subsides: motion along one axis has little effect on the motion along another axis half a second later. This holds for both the original dataset and the TimeGAN, where the cross-correlation is even slightly higher. With the FFT however, this behaviour does not occur at all. The cross-correlation is near-zero for every time lag, meaning that there is no correlation between the different axes. This is unsurprising, as the FFT generates data for the three axes entirely independently from each other. Clearly, this results in a significantly less realistic dataset.

4.1.5 t-SNE and PCA. In addition to the head rotation metrics, we also report the commonly used t-SNE and PCA plots, for visual inspection of synthetic data quality. Figure 9 shows these for the TimeGAN and the FFT. Overlap in the plots is said to indicate similarity of the corresponding datasets. From the plots, it is difficult to tell if either synthetic dataset is superior. Despite this, our metrics clearly indicate the superiority of the TimeGAN dataset, and highlight several shortcomings of the FFT dataset. As such, we argue that this analysis does not sufficiently illustrate how realistic time series data is, proving the value of the metrics defined in this work.

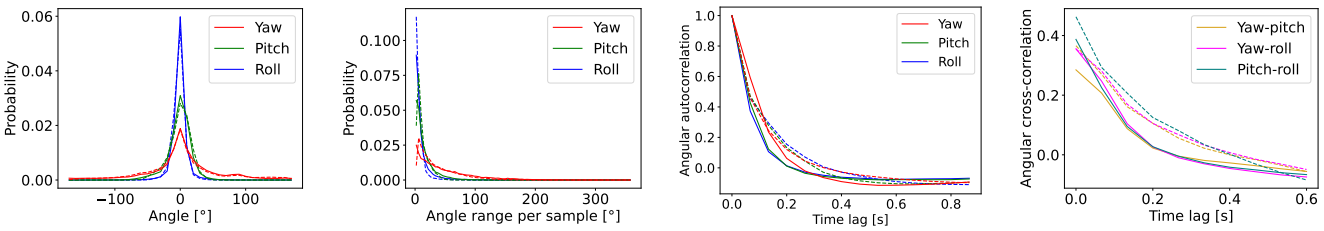
4.2 Generality

While we have provided some intuition for the generality (i.e., dataset-independence) of this approach, we provide further evidence by repeating the evaluation with another dataset. We selected the dataset by Lo *et al.*, discussed in Section 2.1, for its broad range of content. Remember that this contains 50 30 Hz orientational traces of 10 minutes each, each gathered by another user. Without further hyperparameter tuning, we applied the same process to generate a synthetic dataset as above. Figure 10 summarises all metrics. These results again indicate a close match between the source and synthetic datasets. The under-representation of very-low-motion samples again occurs, but no other issues appear. We consider this as a strong indication of the generality of this approach. Intuitively, this is unsurprising, as the TimeGAN had no knowledge of any of



(a) TimeGAN-based synthetic dataset **(b) FFT-based synthetic dataset**

Figure 9: PCA and t-SNE plots, comparing the distributions of the original dataset and a synthetic dataset.



(a) Orientation distribution **(b) Motion distribution** **(c) Autocorrelation** **(d) Cross-correlation**

Figure 10: All metrics evaluated on a second dataset.

the metrics used in this evaluation. The only feedback available to the generator is how well the discriminator is able to recognise samples as real or fake. Without any additional steering, the generator learned from only this information source to match the original dataset very well.

4.3 Discussion

Based on the analysis above, we discuss the utility of the TimeGAN, compared to the FFT on several fronts.

4.3.1 Dataset-specific manual design. : with the FFT, an expert needs to analyse the dataset, determining models for the distributions within the time series. In [5], it is claimed that the model is general, as it can be adapted to another dataset by simply changing the parameters of the distributions. We note that this may not necessarily be the case. As the dataset was recorded in an indoor virtual experience, the yaw is approximated with a multi-modal Gaussian distribution, with their means representing the directions towards the room's walls. We do not expect this distribution to apply well to an outdoor virtual environment, where the user would likely look in any direction with similar probability. In applying TimeGAN however, no such dataset-specific design was applied. The model converts the source dataset to have a normal distribution, meaning it is insensitive to the distributions within the source dataset. The only type of tuning needed was hyperparameter tuning. During this process, we observed that the quality of the synthetic dataset was not particularly sensitive to the hyperparameters. Furthermore, no additional tuning was needed for the second dataset in Section 4.2. Determining the optimal dataset from the different snapshots does require manual intervention, however this essentially comes down to comparing the plots of the metrics presented above, which can be performed rapidly by an individual with no expert knowledge, and could even be automated to an extent.

4.3.2 Runtime. : the FFT is relatively fast, requiring only minutes of computation on a regular workstation computer. GANs however are notoriously computationally inefficient. Training our model on a workstation computer may take over 10 hours. We do note that this high runtime is not prohibitive for this application. Training needs to occur only once, after which the model can output new sequences in a matter of seconds.

Overall, we are convinced that this approach is capable of producing a large array of highly usable head rotation samples regardless of the specific head rotation distribution, for applications such as proactive viewport-dependent streaming and XR beamforming.

5 CONCLUSIONS

In this paper, we presented a novel approach for generating synthetic head rotation data for Extended Reality applications. We showed that, unlike the only other approach currently described in the literature, our TimeGAN-based approach is able to generate realistic data according to a range of metrics which together characterise the head rotation data. Our metrics incorporate where the users look and how they turn their heads. We expect this approach to be valuable to researchers in several XR-related fields, including dynamic multimedia encoding and millimetre-wave beamforming. As such, we commit to releasing an open-source implementation of the system by this paper's publication date. In future work, we intend to reduce TimeGAN's tendency to generate datasets where very-low-motion samples are underrepresented in case of a wide motion distribution.

REFERENCES

[1] Md Manjurul Ahsan, M. A. Parvez Mahmud, Pritom Kumar Saha, Kishor Datta Gupta, and Zahed Siddique. 2021. Effect of Data Scaling Methods on Machine Learning Algorithms and Model Performance. *Technologies* 9, 3 (2021). https://doi.org/10.3390/technologies9030052

[2] Sepehr Alizadehsalehi, Ahmad Hadavi, and Joseph Chuenhuei Huang. 2020. From BIM to extended reality in AEC industry. *Automation in Construction* 116 (2020), 103254. https://doi.org/10.1016/j.autcon.2020.103254

[3] Ksenia Balabaeva and Sergey Kovalchuk. 2019. Comparison of Temporal and Non-Temporal Features Effect on Machine Learning Models Quality and Interpretability for Chronic Heart Failure Patients. *Procedia Computer Science* 156 (2019), 87–96. https://doi.org/10.1016/j.procs.2019.08.183 8th International Young Scientists Conference on Computational Science, YSC2019, 24-28 June 2019, Heraklion, Greece.

[4] Yanan Bao, Huasen Wu, Tianxiao Zhang, Albara Ah Ramli, and Xin Liu. 2016. Shooting a moving target: Motion-prediction-based transmission for 360-degree videos. In *2016 IEEE International Conference on Big Data (Big Data)*. 1161–1170. https://doi.org/10.1109/BigData.2016.7840720

[5] Steve Blandino, Tanguy Ropitault, Raied Caromi, Jacob Chakareski, Mahmudur Khan, and Nada Golmie. 2021. Head Rotation Model for Virtual Reality System Level Simulations. In *2021 IEEE International Symposium on Multimedia (ISM)*. 43–49. https://doi.org/10.1109/ISM52913.2021.00016

[6] Matthew Botvinick, Sam Ritter, Jane X. Wang, Zeb Kurth-Nelson, Charles Blundell, and Demis Hassabis. 2019. Reinforcement Learning, Fast and Slow. *Trends in Cognitive Sciences* 23, 5 (2019), 408–422. https://doi.org/10.1016/j.tics.2019.02.006

[7] G. E. P. Box and D. R. Cox. 1964. An Analysis of Transformations. *Journal of the Royal Statistical Society: Series B (Methodological)* 26, 2 (1964), 211–243. https://doi.org/10.1111/j.2517-6161.1964.tb00553.x

[8] Fred B Bryant and Paul R Yarnold. 1995. Principal-components analysis and exploratory and confirmatory factor analysis. (1995).

[9] Jacob Chakareski, Ridvan Aksu, Xavier Corbillon, Gwendal Simon, and Viswanathan Swaminathan. 2018. Viewport-Driven Rate-Distortion Optimized 360° Video Streaming. In *2018 IEEE International Conference on Communications (ICC)*. 1–7. https://doi.org/10.1109/ICC.2018.8422859

[10] Jacob Chakareski, Mahmudur Khan, Tanguy Ropitault, and Steve Blandino. 2020. 6DOF Virtual Reality Dataset and Performance Evaluation of Millimeter Wave vs. Free-Space-Optical Indoor Communications Systems for Lifelike Mobile VR Streaming. In *2020 54th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 1051–1058. https://doi.org/10.1109/IEEECONF51394.2020.9443328

[11] Jack C. P. Cheng, Keyu Chen, and Weiwei Chen. 2020. State-of-the-Art Review on Mixed Reality Applications in the AECO Industry. *Journal of Construction Engineering and Management* 146, 2 (2020), 03119009. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001749

[12] Xavier Corbillon, Francesca De Simone, and Gwendal Simon. 2017. 360-Degree Video Head Movement Dataset. In *Proceedings of the 8th ACM on Multimedia Systems Conference* (Taipei, Taiwan) *(MMSys'17)*. Association for Computing Machinery, New York, NY, USA, 199–204. https://doi.org/10.1145/3083187.3083215

[13] Saloni Dash, Andrew Yale, Isabelle Guyon, and Kristin P. Bennett. 2020. Medical Time-Series Data Generation Using Generative Adversarial Networks. In *Artificial Intelligence in Medicine*, Martin Michalowski and Robert Moskovitch (Eds.). Springer International Publishing, Cham, 382–391.

[14] Jeffrey De Fauw, Joseph R Ledsam, Bernardino Romera-Paredes, Stanislav Nikolov, Nenad Tomasev, Sam Blackwell, Harry Askham, Xavier Glorot, Brendan O'Donoghue, Daniel Visentin, et al. 2018. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature medicine* 24, 9 (2018), 1342–1350.

[15] Mohammed S. Elbamby, Cristina Perfecto, Mehdi Bennis, and Klaus Doppler. 2018. Toward Low-Latency and Ultra-Reliable Virtual Reality. *IEEE Network* 32, 2 (2018), 78–84. https://doi.org/10.1109/MNET.2018.1700268

[16] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633* (2017).

[17] Stephan Fremerey, Ashutosh Singla, Kay Meseberg, and Alexander Raake. 2018. AVtrack360: An Open Dataset and Software Recording People's Head Rotations Watching 360° Videos on an HMD. In *Proceedings of the 9th ACM Multimedia Systems Conference* (Amsterdam, Netherlands) *(MMSys '18)*. Association for Computing Machinery, New York, NY, USA, 403–408. https://doi.org/10.1145/3204949.3208134

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc.

[19] Zhiming Hu, Andreas Bulling, Sheng Li, and Guoping Wang. 2021. EHTask: Recognizing User Tasks from Eye and Head Movements in Immersive Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics* (2021), 1–1. https://doi.org/10.1109/TVCG.2021.3138902

[20] Jong-Beom Jeong, Soonbin Lee, Il-Woong Ryu, Tuan Thanh Le, and Eun-Seok Ryu. 2020. *Towards Viewport-Dependent 6DoF 360 Video Tiled Streaming for Virtual Reality Systems*. Association for Computing Machinery, New York, NY, USA, 3687–3695.

[21] Alexandra D. Kaplan, Jessica Cruit, Mica Endsley, Suzanne M. Beers, Ben D. Sawyer, and P. A. Hancock. 2021. The Effects of Virtual Reality, Augmented Reality, and Mixed Reality as Training Enhancement Methods: A Meta-Analysis. *Human Factors* 63, 4 (2021), 706–726. https://doi.org/10.1177/0018720820904229 PMID: 32091937.

[22] Tero Karras, Samuli Laine, and Timo Aila. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[23] Chen Li, Mai Xu, Xinzhe Du, and Zulin Wang. 2018. Bridge the Gap Between VQA and Human Behavior on Omnidirectional Video: A Large-Scale Dataset and a Deep Learning Model. In *Proceedings of the 26th ACM International Conference on Multimedia* (Seoul, Republic of Korea) *(MM '18)*. Association for Computing Machinery, New York, NY, USA, 932–940. https://doi.org/10.1145/3240508.3240581

[24] Wen-Chih Lo, Ching-Ling Fan, Jean Lee, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. 2017. 360° Video Viewing Dataset in Head-Mounted Virtual Reality. In *Proceedings of the 8th ACM on Multimedia Systems Conference* (Taipei, Taiwan) *(MMSys'17)*. Association for Computing Machinery, New York, NY, USA, 211–216. https://doi.org/10.1145/3083187.3083219

[25] Pietro Lungaro, Rickard Sjöberg, Alfredo José Fanghella Valero, Ashutosh Mittal, and Konrad Tollmar. 2018. Gaze-Aware Streaming Solutions for the Next Generation of Mobile VR Experiences. *IEEE Transactions on Visualization and Computer Graphics* 24, 4 (2018), 1535–1544. https://doi.org/10.1109/TVCG.2018.2794119

[26] Daniel Martin, Ana Serrano, Alexander W. Bergman, Gordon Wetzstein, and Belen Masia. 2022. ScanGAN360: A Generative Model of Realistic Scanpaths for 360° Images. *IEEE Transactions on Visualization and Computer Graphics* 28, 5 (2022), 2003–2013. https://doi.org/10.1109/TVCG.2022.3150502

[27] Olof Mogren. 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904* (2016).

[28] Dimitris Mourtzis, Vasileios Siatras, and John Angelopoulos. 2020. Real-Time Remote Maintenance Support Based on Augmented Reality (AR). *Applied Sciences* 10, 5 (2020). https://doi.org/10.3390/app10051855

[29] Afshin Taghavi Nasrabadi, Aliehsan Samiei, Anahita Mahzari, Ryan P. McMahan, Ravi Prakash, Mylène C. Q. Farias, and Marcelo M. Carvalho. 2019. A Taxonomy and Dataset for 360° Videos. In *Proceedings of the 10th ACM Multimedia Systems Conference* (Amherst, Massachusetts) *(MMSys '19)*. Association for Computing Machinery, New York, NY, USA, 273–278. https://doi.org/10.1145/3304109.3325812

[30] William J. Shelstad, Dustin C. Smith, and Barbara S. Chaparro. 2017. Gaming on the Rift: How Virtual Reality Affects Game User Satisfaction. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 61, 1 (2017), 2072–2076. https://doi.org/10.1177/1541931213602001

[31] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of big data* 6, 1 (2019), 1–48.

[32] J.I. Smith. 1975. A computer generated multipath fading simulation for mobile radio. *IEEE Transactions on Vehicular Technology* 24, 3 (1975), 39–40. https://doi.org/10.1109/T-VT.1975.23600

[33] Ryan R. Strauss, Raghuram Ramanujan, Andrew Becker, and Tabitha C. Peck. 2020. A Steering Algorithm for Redirected Walking Using Reinforcement Learning. *IEEE Transactions on Visualization and Computer Graphics* 26, 5 (2020), 1955–1963. https://doi.org/10.1109/TVCG.2020.2973060

[34] Jakob Struye, Filip Lemic, and Jeroen Famaey. 2020. Towards Ultra-Low-Latency mmWave Wi-Fi for Multi-User Interactive Virtual Reality. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9322284

[35] Jakob Struye, Filip Lemic, and Jeroen Famaey. 2021. Millimeter-wave beamforming with continuous coverage for mobile interactive virtual reality. *arXiv preprint arXiv:2105.11793* (2021).

[36] Shishir Subramanyam, Irene Viola, Alan Hanjalic, and Pablo Cesar. 2020. User Centered Adaptive Streaming of Dynamic Point Clouds with Low Complexity Tiling. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*. Association for Computing Machinery, New York, NY, USA, 3669–3677. https://doi.org/10.1145/3394171.3413535

[37] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[38] J. M. P. van Waveren. 2016. The Asynchronous Time Warp for Virtual Reality on Consumer Hardware. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology* (Munich, Germany) *(VRST '16)*. Association for Computing Machinery, New York, NY, USA, 37–46. https://doi.org/10.1145/2993369.2993375

[39] Yu Wang, Ticao Zhang, Shiwen Mao, and Theodore (Ted) S. Rappaport. 2021. Directional neighbor discovery in mmWave wireless networks. *Digital Communications and Networks* 7, 1 (2021), 1–15. https://doi.org/10.1016/j.dcan.2020.09.005

[40] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. 2017. A Dataset for Exploring User Behaviors in VR Spherical Video Streaming. In *Proceedings of*

the 8th ACM on Multimedia Systems Conference (Taipei, Taiwan) *(MMSys'17)*. Association for Computing Machinery, New York, NY, USA, 193–198. https://doi.org/10.1145/3083187.3083210

[41] Mai Xu, Yuhang Song, Jianyi Wang, Minglang Qiao, Liangyu Huo, and Zulin Wang. 2019. Predicting Head Movement in Panoramic Video: A Deep Reinforcement Learning Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 11 (2019), 2693–2708. https://doi.org/10.1109/TPAMI.2018.2858783

[42] In-Kwon Yeo and Richard A. Johnson. 2000. A new family of power transformations to improve normality or symmetry. *Biometrika* 87, 4 (12 2000), 954–959. https://doi.org/10.1093/biomet/87.4.954

[43] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc.

[44] D.J. Young and N.C. Beaulieu. 2000. The generation of correlated Rayleigh random variates by inverse discrete Fourier transform. *IEEE Transactions on Communications* 48, 7 (2000), 1114–1127. https://doi.org/10.1109/26.855519

[45] Emin Zerman, Radhika Kulkarni, and Aljosa Smolic. 2021. User Behaviour Analysis of Volumetric Video in Augmented Reality. In *2021 13th International Conference on Quality of Multimedia Experience (QoMEX)*. 129–132. https://doi.org/10.1109/QoMEX51781.2021.9465456

[46] Anfu Zhou, Leilei Wu, Shaoqing Xu, Huadong Ma, Teng Wei, and Xinyu Zhang. 2018. Following the Shadow: Agile 3-D Beam-Steering for 60 GHz Wireless Networks. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*. 2375–2383. https://doi.org/10.1109/INFOCOM.2018.8486399

[47] Arzu Çöltekin, Ian Lochhead, Marguerite Madden, Sidonie Christophe, Alexandre Devaux, Christopher Pettit, Oliver Lock, Shashwat Shukla, Lukáš Herman, Zdeněk Stachoň, Petr Kubíček, Dajana Snopková, Sergio Bernardes, and Nicholas Hedley. 2020. Extended Reality in Spatial Sciences: A Review of Research Challenges and Future Directions. *ISPRS International Journal of Geo-Information* 9, 7 (2020). https://doi.org/10.3390/ijgi9070439