IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 9, 2021, accepted May 19, 2021, date of publication May 21, 2021, date of current version June 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3082843

Synthetic Radar Dataset Generator for Macro-Gesture Recognition

ALEXANDROS NINOS[1,2], (Member, IEEE), JÜRGEN HASCH[1], (Senior Member, IEEE), MARIO EMILIO PIZANO ALVAREZ[1], AND THOMAS ZWICK[2], (Fellow, IEEE)

[1]Corporate Sector Research and Advance Engineering, Robert Bosch GmbH, 70465 Stuttgart, Germany
[2]Institut für Hochfrequenztechnik und Elektronik, Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany

Corresponding author: Alexandros Ninos (alexandros.ninos@de.bosch.com)

ABSTRACT Recent developments in mmWave technology allow the detection and classification of dynamic arm gestures. However, achieving a high accuracy and generalization requires a lot of samples for the training of a machine learning model. Furthermore, in order to capture variability in the gesture class, the participation of many subjects and the conduct of many gestures with different arm speed are required. In case of macro-gestures, the position of the subject must also vary inside the field of view of the device. This would require a significant amount of time and effort, which needs to be repeated in case that the sensor hardware or the modulation parameters are modified. In order to reduce the required manual effort, here we developed a synthetic data generator that is capable of simulating seven arm gestures by utilizing Blender, an open-source 3D creation suite. We used it to generate 600 artificial samples with varying speed of execution and relative position of the simulated subject, and used them to train a machine learning model. We tested the model using a real dataset recorded from ten subjects, using an experimental sensor. The test set yielded 84.2% accuracy, indicating that synthetic data generation can significantly contribute in the pre-training of a model.

INDEX TERMS Gesture sensing, data creation system, mm-wave technology, machine learning, synthetic data set.

I. INTRODUCTION

Hand gesture recognition has become available by using miniaturized, low-power Radar sensors, and was first demonstrated by the Soli project [1]. After that, many research groups focused on reproducing and improving the results, which required the collection of a significant amount of training samples. In [2], 2750 samples were recorded from 11 subjects, each performing 10 gestures 25 times. Similarly, in [3] the authors collected 7200 samples from 20 subjects which performed 12 gestures 30 times. In both cases, the authors recorded micro-gestures that were performed a few centimeters above the device. However, in our previous work [4], the subjects conducted gestures at an approximate distance of 2 m from the device, not only bore-sight but also in various positions inside the field of view (FoV) of the sensor. In total, we collected 1500 samples from 10 subjects that carried out 10 different gestures, including random movements which were regarded as noise.

The associate editor coordinating the review of this manuscript and approving it for publication was Hiu Yung Wong.

Collecting this amount of data requires manual effort and is time-consuming. Moreover, in case that the sensor hardware (e.g., antenna configuration) or the modulation is modified, the measurements have to be repeated. Therefore, the need for a simulator that is able to generate artificial samples for various experimental cases arises. It is important to point out that in many other machine learning problems several dataset generators have been proposed [5]–[8].

The contribution of such a generator for the mmWave case could be threefold: it could first and foremost significantly reduce the experimental time for the data recordings, it could enable the testing of a broader spectrum of experimental cases, and finally, it could explore various options for the modulation and hardware parameters [9]. As such, it could increase the variability of samples, especially in the case of macro-gestures during which the subject could be placed in different places.

Many approaches have been presented on simulating human motion combined with mmWave sensing. In [10] the authors used a kinematic model from [11] to generate synthetic micro-Doppler (mD) [12] spectrogram and to

train a deep learning model. A similar application-specific approach was developed in [13] for simulating the reflections of cyclists. The authors manually created a model with point targets that represented various parts of the bicycle and the person riding it. In both approaches, only a specific kind of motion could be generated, limiting the number of use-cases that could be simulated. In [14] the authors surpassed this problem by using Blender [15], and its ray tracing capability for graphic simulation. Specifically, after designing a static scene, they used the rendered images "z-pass" and "combined pass", through which they calculated the range, angle of arrival (AoA) and amplitude information of the object that was in each pixel of the images. The number of point targets was defined by the number of pixels of the rendered images, which can be set by the user. It is worth pointing out that the graphics in the scene were static and only the Radar sensor was allowed to move. In [16], the authors overcame this problem by using another rendered image called "speed vector pass", which contains information about pixels moving in two dimensions (x and y). The authors were able to generate spectrogram from a synthetically generated human figure that was waving both hands. In [9], Blender was again used to generate an arbitrary animation and created multiple variants via a Python script. Then, they extracted the position of the body joints, used them in a custom Frequency-modulated continuous wave (FMCW) Radar simulator and generated a dataset with 2000 samples per gesture. Finally yet importantly, in [17] another method was presented for generating synthetic Radar data for gesture recognition. The authors converted a Kinect dataset into Radar signatures using a simulation framework, then they extracted several features out of the spectrogram and used them to train a machine learning (ML) model with four gestures. However, they did not test the accuracy of their model with a real Radar dataset; they only compared Radar data with the output of their simulator and found meaningful similarities.

In this work, we present a novel system that can generate synthetic Radar data for seven arm gestures, which we then used to produce a dataset with 600 samples with varying speed of execution and position of the animation relative to the Radar. Using the processing chain of our previous work in [4] we generated three feature maps for each synthetic sample, extracted empirical features and trained a machine learning model. We tested the trained model using a real dataset that we collected from ten subjects who repeated the same gestures 15 times. Our model yielded 84.2% accuracy, indicating a successful combination of our proposed simulator and feature extractor. We used Python programming language in combination with Numpy [18] for matrix manipulation and Scipy [19] for scientific computing.

The remainder of this paper is structured as follows: Section II introduces the platform that generates the synthetic dataset, the generation of animations, the extraction of point targets, and the simulation of Radar output. Section III explains the signal processing chain and Section IV provides details on how the real dataset was recorded using experimental hardware. Finally, section V presents the results of the simulator and compares it against real data.

II. SYNTHETIC DATASET GENERATION

The pipeline for generating a synthetic sample consists of four main parts, as illustrated in Fig. 1. First a human model is selected and the animation is configured so that it moves its arm according to a specified gesture. Then using Blender, the animation is simulated and its point targets are extracted and used in the third step which simulates a Radar sensor.

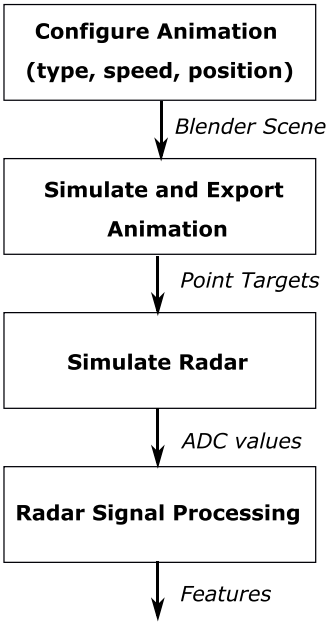

FIGURE 1. High level description of the sample generation process.

A. GENERATION OF HUMAN-BODY ANIMATIONS

The Blender platform was used for simulating human animations. It is an open source 3D creation suite, which supports the entirety of the 3D pipeline: modeling, rigging, animation, simulation, rendering, compositing, and motion tracking [15]. It offers a rich Application Programming Interface (API) for Python scripting that can be employed to customize the application and write specialized tools.

To create a Blender scene, it is necessary to first generate 3D models out of primitives (cubes, cylinders, spheres, etc.) by joining them or modifying their mesh. Meshes consist of vertices, edges and faces, and define the shape of a 3D model. Fig 2 shows an example of mesh structure, derived from the documentation of Blender. It is also possible to import 3D models from libraries; Fig. 3 shows the one that we used, which we obtained from the TurboSquid library. Models, as well as other Blender objects (e.g., camera, lights) can be arranged in the scene by modifying their location, rotation or scale.

In order to generate gestures, an armature has to be assigned to a model. Armatures include bones, which can be rotated to modify the shape of the mesh of a model

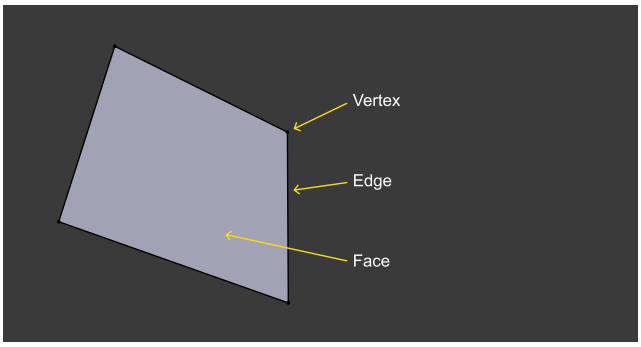

FIGURE 2. Example of mesh structure.

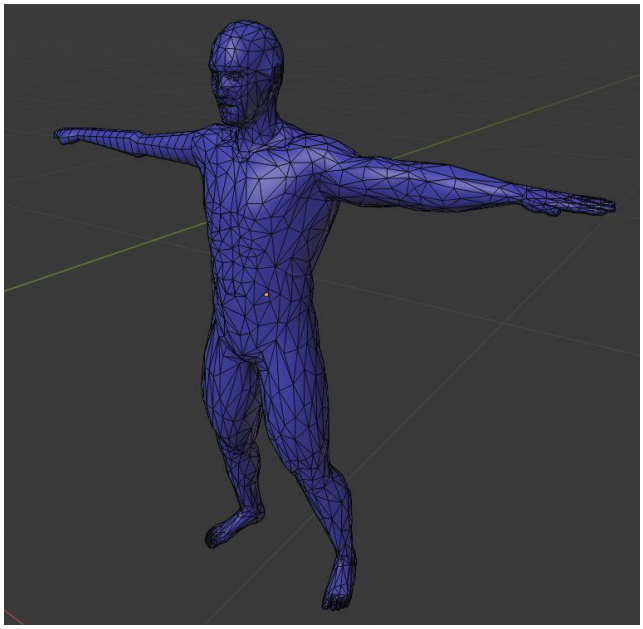

FIGURE 3. Body model from TurboSquid.

or a 3D object. This way, the pose of a model can be selected according to the use-case. For defining a gesture, the user has to set key-points in the beginning and ending part of the motion, then Blender will calculate the positions of intermediate frames by solving the inverse kinematics problem. In this work, we created synthetic data for the gesture classes "push", "pull", "swipe-left", "swipe-right", "rotate", "wave", "push-pull", similar to the ones defined in [4]. Fig. 4 shows the model together with the armature that we created, performing a "swipe-left".

B. EXTRACT POINT TARGETS FROM ANIMATION

After simulating the animation that performs a gesture, we need to extract point targets that will be given to the Radar Simulator. These targets are actually the vertices of the body model. We use the API of Blender to access their position for all simulated frames, then we can easily calculate their velocity. After that, we apply ray casting to find the points visible by the Radar at each frame. If the vertex is not visible at a particular frame, we set the Radar Cross Section (RCS) value to zero, so that it is not taken into consideration during the Radar simulation. With the above steps we managed to extract a tensor with dimensions: number of vertices × number of frames. Each element of the tensor contains position, velocity and RCS of the vertex. Fig. 5 shows an overview of the extraction process.

C. RADAR SIMULATOR

The simulator utilizes a monostatic Radar, with multiple receiving (Rx) and transmitting (Tx) antennas. It estimates the range of k-th target using the travel time $\tau_k = t_k^{Rx} - t_k^{Tx}$ of the electromagnetic wave from the transmitting element to the target and back to the receiver. In the simple case, the target is static and its range could be estimated using Eq. 1.

$$r_k = \frac{c_0 \tau_k}{2} \tag{1}$$

For a moving target, the range is a function of the initial range r_k^0, the radial velocity s_k relative to the sensor and time t as shown in Eq. 2.

$$r_k(t) = r_k^0 + s_k t \tag{2}$$

Using the well known Radar equation [20], we can estimate received power P_r on an antenna, depending on transmitted power P_t, gain of transmitting and receiving antennas G_t and G_r, Radar cross-section (RCS) σ, wavelength λ and range of a target, according to the formulation given by Eq. 3.

$$P_r = \frac{P_t G_t G_r \lambda^2 \sigma}{(4\pi)^3 r_k^4} \tag{3}$$

Thus, the total attenuation a_k from the transmitting antenna to target k and back to the receiving antenna is calculated using Eq. 4.

$$a_k = \frac{P_r}{P_t} = \frac{G_t G_r \lambda^2 \sigma}{(4\pi)^3 r_k^4} \tag{4}$$

Using the time-delay and attenuation defined above, the voltage referred relationship between the received signal u_k^{Rx} and the transmitted signal u_k^{Tx} is provided by Eq. 5

$$u_k^{Rx}(t) = \sqrt{a_k} u^{Tx}(t - \tau_k) \tag{5}$$

Various modulation schemes that provide range and speed measurements have been developed; we decided to use FMCW because it is well-studied in the literature and many commercial sensors are already available [21], [22]. A system that uses FMCW modulation transmits a series of chirps (also called ramps), which are reflected by the targets and when received they are mixed with the transmit chirp. This results in a frequency which is proportional to the distance, known as *beat frequency* f_b.

In detail, consider a linear frequency chirp $f^{Tx}(t)$ as in Eq. 6 with a slope k_r.

$$f^{Tx}(t) = f + k_r t \tag{6}$$

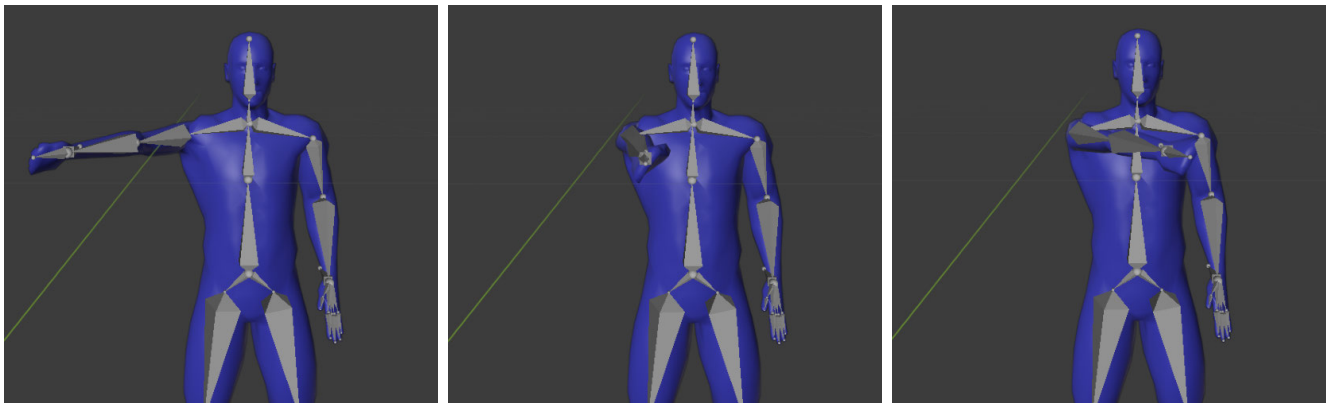

FIGURE 4. Human model with armature performing "swipe-left" gesture.

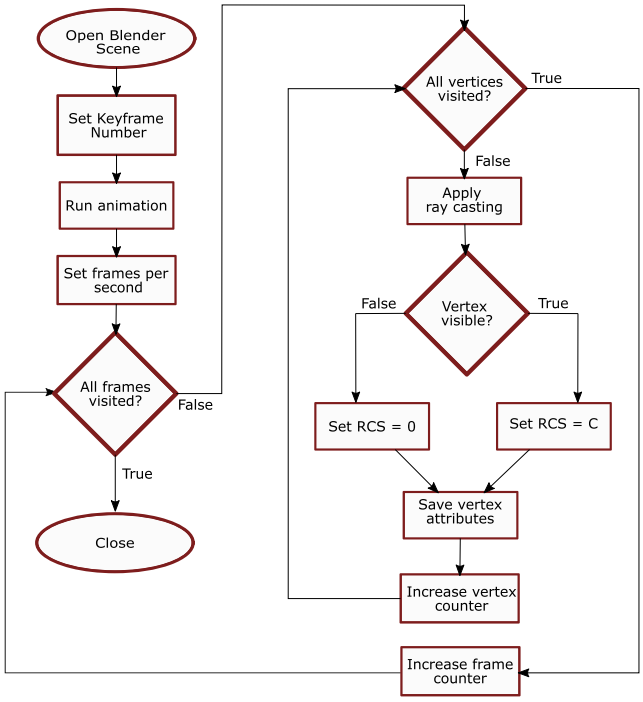

FIGURE 5. Overview of the pipeline for extracting targets from a Blender animation.

The modulated signal is calculated by integrating the frequency into the phase [20], as in Eq. 7

$$u^{Tx}(t) = e^{j2\pi \int f^{Tx}(t)dt} = e^{j(\pi k_r t^2 + 2\pi f t + \theta_0)} \tag{7}$$

θ_0 being the phase of $u^{Tx}(t)$ for $t = 0$. Using the above and Eq. 4 yields the FMCW received signal as in Eq. 8

$$u_k^{Rx}(t) = \sqrt{a_k} e^{j(\pi k_r (t-\tau_k)^2 + 2\pi f (t-\tau_k) + \theta_0)} \tag{8}$$

By mixing transmit and received signal, the baseband signal u_k^{IF} is calculated.

We developed a simulation framework that generates the output of a Radar sensor, given the location, velocity and RCS of point targets that are provided as input. The first step is to calculate the travel time of the wave from the transmitting antenna, to the target and back to the receiving antenna. Then we use Eq. 2 and 4 to calculate the attenuation of the wave. In the next step, we use Eq. 7 and 8 to calculate time domain transmitted and received signal for one chirp. By mixing the signals we can calculate the baseband signal which is what a Radar sensor would record using its analogue to digital converters (ADC). We repeat this procedure for multiple chirps in order to achieve the chirp sequence waveform like in [22].

The user can configure the simulator in order to approximately mimic the hardware. For example, the user can provide values for the noise figure (NF) and gain of the low-noise amplifiers (LNA). Moreover, an input matrix defines the antenna gain for a discrete set of AoA. This way, the return signals of the animation will depend on its aspect angle. Last but not least, the simulator can also handle Multiple Input Multiple Output (MIMO) schemes for improved AoA estimation, with Time Domain Multiplexing (TDM) [23]. Fig. 6 shows a diagram with the basic blocks of the simulator for a 2 T_x - 2 R_x configuration.

III. PROCESSING PIPELINE FOR GESTURE RECOGNITION

As already mentioned in the introduction, the signal processing chain relies heavily in our previous work [4]. Briefly here we explain the most important steps: For each measurement frame, after the two dimensional Fast Fourier Transform (2D-FFT) and noise thresholding using Constant False Alarm Rate (CFAR) are applied, the mD vector is calculated by selecting the maximum power bin for every fixed radial velocity value along the range dimension. Then we apply Digital Beamforming (DBF) [23] for the selected bins in azimuth and elevation dimensions. The above procedure is repeated for all measurement frames and the vectors are concatenated to create images with mD and AoA, which will be sparse due to the CFAR thresholding. After that, an event detection algorithm decides if a significant motion took place during the last frames, and only then will activate the machine learning part.

The latter, will first extract three one-dimensional signals out of the three feature maps. Then it will calculate the

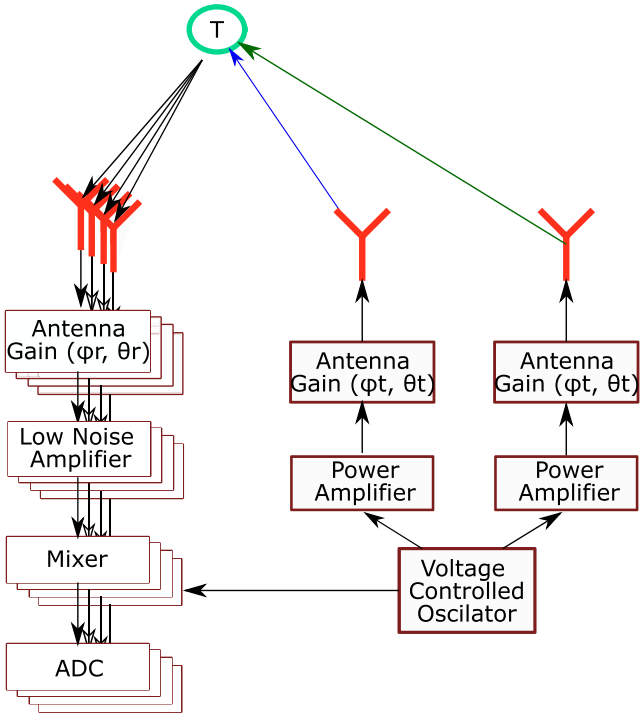

FIGURE 6. An example scenario for the Radar Simulator with two transmitting, two receiving antennas and one point-target (depicted by 'T'). The signal source will be amplified and the antenna gain will be calculated based on the target's location. Similarly, in the receiving side, the antenna gain will be calculated for each target, the returned signals will be amplified and sent to the mixer for calculating the baseband signal.

following nine features: (1) Number of zero-crossings in radial velocity, (2, 3) arguments of maximum and minimum radial velocity, (4, 5) maximum and minimum radial velocity, (6) difference between maximum and minimum angle in azimuth, (7) difference between maximum and minimum angle in elevation, (8) difference of angle in azimuth when radial velocity reached its maximum and minimum value, (9) difference of angle in elevation when radial velocity reached its maximum and minimum value. Finally, for classification we used a Multi-Layer Perceptron (MLP), with one hidden layer consisted of 32 neurons.

IV. COLLECTION OF RADAR DATA USING EXPERIMENTAL SENSOR

In our previous work [4], we collected real Radar data using an experimental sensor with two transmitting and four receiving antennas. Fig. 7 shows a simplified block diagram consisting of the high frequency and the baseband part. The former includes a two-channel transmitter and a four-channel receiver, a signal source capable of fast linear frequency ramps of up to 2 GHz bandwidth, as well as transmit and receive antennas. The latter contains analog interface electronics, ADC, digital logic, and power supply. The digital logic itself consists of a Field-programmable gate array (FPGA) part, that controls the real-time operation of the Radar sensor, and an ARM micro-controller, that communicates with a host PC, which is connected over Ethernet.

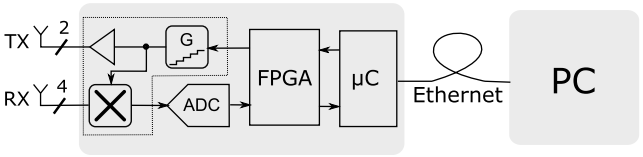

FIGURE 7. Simplified block diagram of the radar sensor setup.

TABLE 1. System parameters.

Name	Value
Antenna Gain	10 dBi
Transmit power	12 dBm
FoV (az/el)	120/60°
Antennas (receiving/transmitting)	4/2
Center Frequency	76.75 GHz
Chirp duration	66 μs
Chirps per Tx Antenna	64
Break between chirps	50 μs
Bandwidth	1.5 GHz
Measurement duration	15 ms
Break between measurements	5 ms
Total duration	20 ms

Our setup also allows for MIMO scheme, operating the transmitters in TDM. The system parameters of the Radar are given in Table 1.

The subjects were placed at a distance of approximately two meters in front of the Radar. They performed gestures not only at bore-sight but also a few degrees to left and right side in order to capture more corner cases. We collected data from ten subjects, which performed 15 repetitions of the seven gestures resulting in 1050 samples.

V. RESULTS

Through the synthetic data generation we wanted to create samples that would capture many corner-cases of a real scenario. That is why, we created 60 samples for each human animation, using different configurations. The parameters that we can modify are the position of the Radar and the speed of execution. In addition, we found that some subjects performed the "push", "pull" and "rotate" with high variability, that is why we created two different animations for each of the above three gestures. In total, we created 10 animations, through which we generated 600 samples.

Typical examples of the variability that we can create are shown in Fig. 8. We modified the duration of motion from 13 to 16 frames and we also varied by 40% the speed of the arm gesture.

A. FEATURE MAPS

In this subsection we present generated feature maps for a few typical gestures. The first row contains the results using real hardware and the second row using the simulator. Columns correspond to the mD, AoA in azimuth and elevation respectively. As it can be seen, the magnitude of the mD for the synthetically generated samples is significantly lower in comparison to the real samples. It is important to stress out that on purpose we did not want to fine-tune the simulator's parameters to match the power of the real gestures (e.g., RCS,

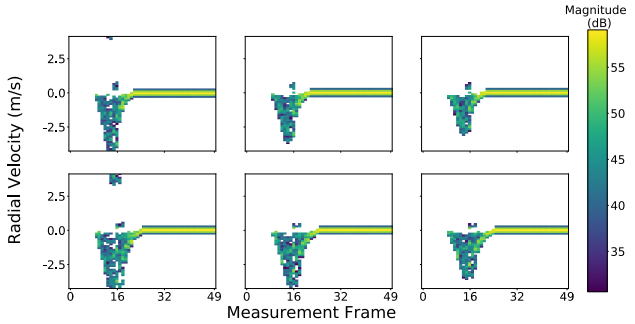

FIGURE 8. Examples of synthetically generated "push" gesture with different motion characteristics. In the first row, the motion has a duration of 13 frames, whereas in the second 16. The columns contain samples with varying maximum velocity.

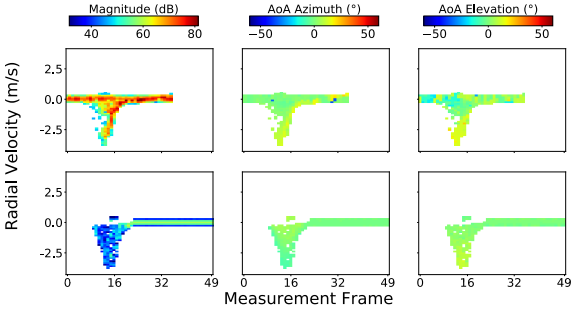

FIGURE 9. Feature maps of a "push" gesture as captured by Radar sensor in the first row and generated by simulator in the second.

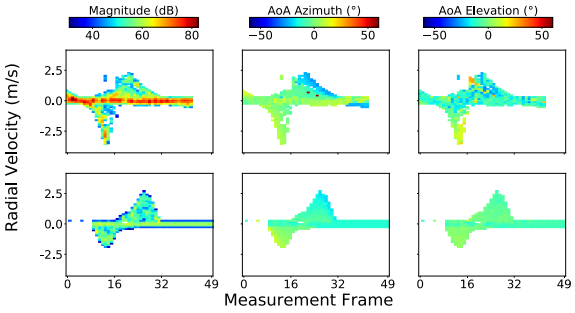

FIGURE 10. Feature maps of a "swipe-left" gesture as captured by Radar sensor in the first row and generated by simulator in the second.

transmitted power, noise figure), in order to show that the ML pipeline can extract meaningful features without taking into consideration the absolute power value. In other words, the gesture recognition pipeline that we use remains invariant to the absolute power value, and as such can generalize across different levels or Radar noise and RCS.

In Fig. 9 the feature maps correspond to the "push" gesture.

Fig. 10 depicts the mD maps of a "swipe-left", that is why in the Azimuth Feature Map, the AoA slowly decreases.

Similarly, Fig. 11 depicts a "swipe-right". Therefore, the AoA in the Azimuth Feature Map increases.

The next scenario in Fig. 12 shows the effect of the subject's position relative to the Radar. The mD map looks similar to a "push" gesture, since in this case the hand of the user always has negative radial velocity. However, the Azimuth

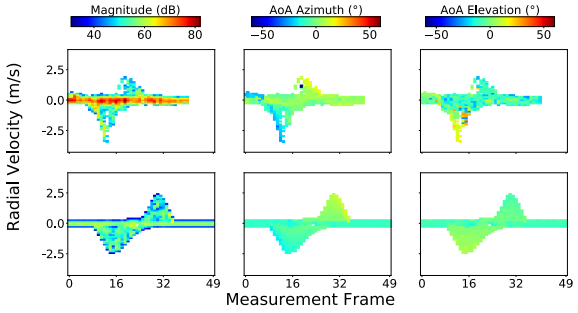

FIGURE 11. Feature maps of a "swipe-right" gesture as captured by Radar sensor in the first row and generated by simulator in the second.

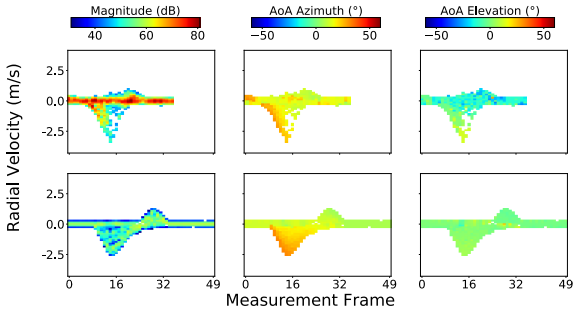

FIGURE 12. Feature maps of a "swipe-left" gesture as captured by Radar sensor in the first row and generated by simulator in the second. Subject was positioned on the right side of the Radar, which had a significant effect on the mD map.

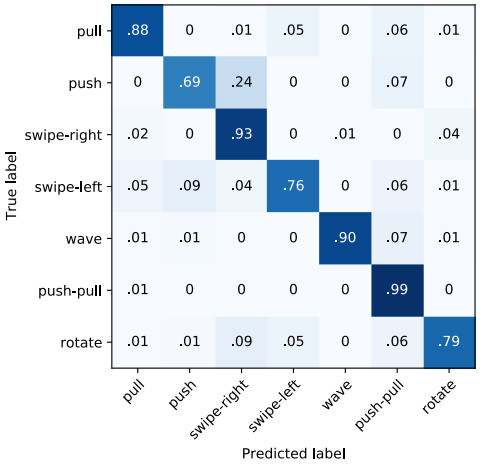

FIGURE 13. Confusion matrix of test set.

Feature Map captures the variation and that information is propagated to the ML model.

B. SUPERVISED LEARNING

We used the synthetically generated dataset and the signal processing pipeline to train a classifier and we tested it on the real dataset that we collected. The average accuracy in the test set was 84.2%, Fig 13 depicts the confusion matrix of the test set. Also Table 2 provides details of the classification results for each class. This indicates that the synthetic dataset generator can be used for pre-training a ML model and could be very helpful for capturing certain known corner cases. Then the model can be further fine-tuned with a dataset collected

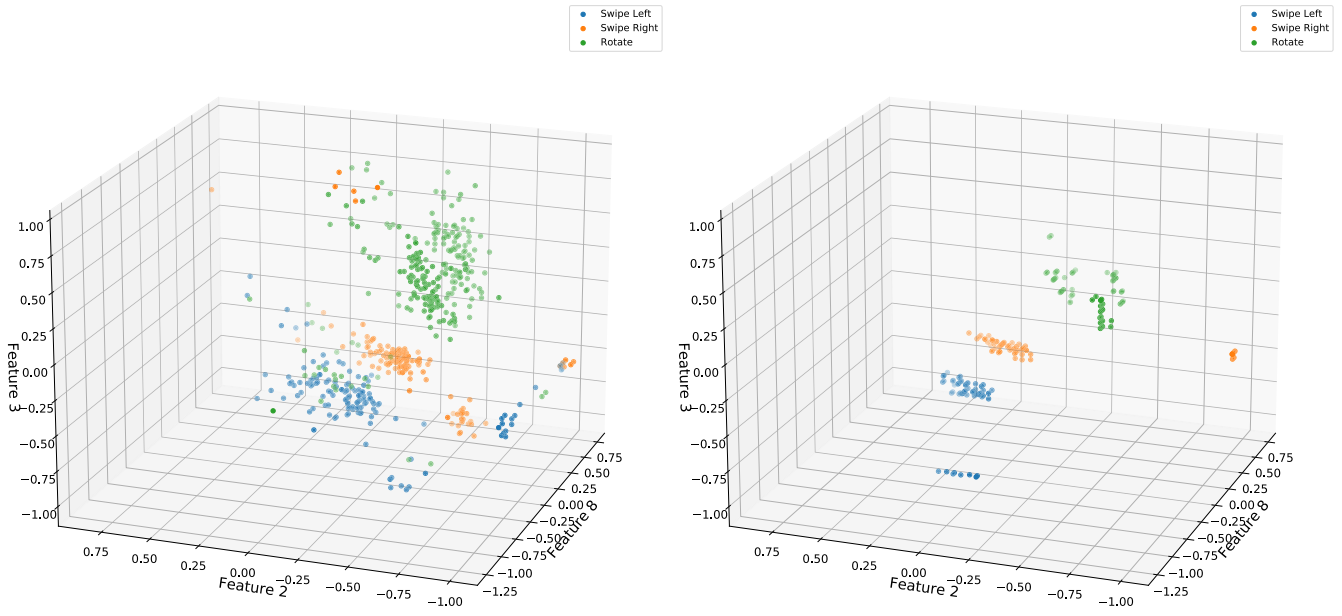

FIGURE 14. Scatter plot with three classes. The left subplot contains results from measurements and right subplot from simulation.

TABLE 2. Detailed results of each gesture from supervised learning.

Gesture	# Samples	Precision	Recall
Pull	120	0.9	0.88
Push	120	0.83	0.69
Swipe-Right	60	0.68	0.93
Swipe-Left	60	0.83	0.76
Wave	60	0.99	0.90
Push-Pull	60	0.72	0.99
Rotate	120	0.97	0.79

TABLE 3. Attributes of the datasets and result of supervised learning.

# Gestures	# Synthetically Generated Samples	# Real Samples	Test Set Average Accuracy
7	600	1050	84.2%

using the Radar sensor. Table 3 summarizes the proportion of the synthetic and real samples as well as the overall accuracy.

C. EMPIRICAL FEATURE EXTRACTION

As a final step for comparing the synthetic and real dataset, we compared the features that are generated by the processing pipeline in both cases. First, we created three dimensional scatter plots, where the left subplot shows real, and the right shows synthetic data. In Fig. 14, we show samples from gestures "swipe-left", "swipe-right" and "rotate" using the features 2, 3, and 8. Likewise, in Fig. 15, we show samples from gestures "pull", "push", "push-pull" and "wave" using features 1, 4, and 5.

In addition, we performed the Kruskal-Wallis H-test [24], to test the null hypothesis that the population median of the synthetic and real data are equal for each feature. When the p-value is below the significance level 0.05, then the test rejects the null-hypothesis. In other words, there is not enough evidence to suggest that the samples come from the same distribution, therefore it is concluded that they come from different ones. The reason why we selected a Kruskal-Wallis test instead of Analysis of Variance (ANOVA) test is that the former is a non-parametric test that does not assume normality (a Shapiro [25] test on our data indicated that our features do not come from a normal distribution). Fig. 16 shows the result of the Kruskal-Wallis test for all combinations of gestures and features. In this intra-feature level we see that the majority of features appear to come from different distributions for the real and synthetic case.

TABLE 4. Result of the Wilcoxon test for the medians of the distribution of each feature. The test accepts the hypothesis that the two feature sets come from the same distribution.

-	Pull	Push	Swipe-Right	Swipe-Left	Wave	Push-Pull	Rotate
p-value	0.30	0.35	0.83	0.35	0.67	0.87	0.36

Nevertheless, since our classifier uses non-linear combinations of the features, it is not the absolute value of each feature that defines the result, rather the offset among their distributions. To visualize that, we show Fig. 17, where it is obvious that the pairs (real - first row, synthetic- bottom row) of distributions of each feature are very close, relative to the others. To also test this beyond the apparent visual inspection, we performed a Wilcoxon non-parametric paired test [26] for the medians of the distribution of each feature, for the two cases. We did so for each of the seven gestures. The test accepted the Null hypothesis that the two (synthetic medians and real medians) come from the same distribution in all the gestures (p-value > 0.3), as it can be seen in Table 4, which supports what we also could see in the box-plot figure.

Finally, as a sanity check, we further wanted to evaluate whether this difference in the intra-feature distribution of the real and synthetic data that was pointed out by the

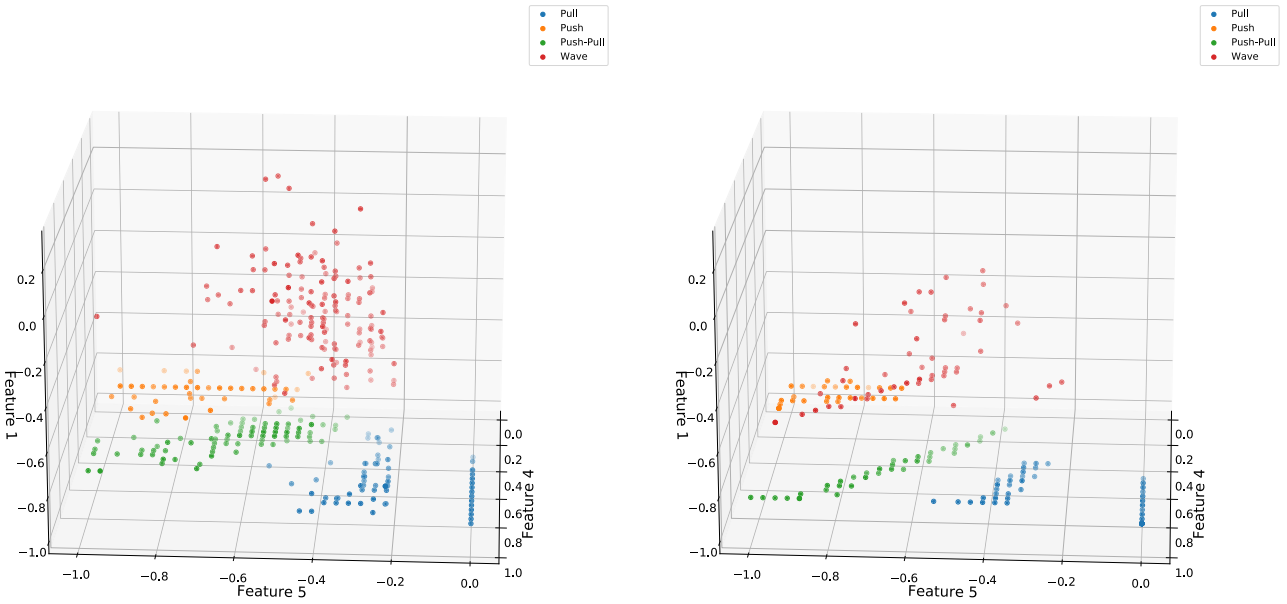

FIGURE 15. Scatter plot with four classes. The left subplot contains results from measurements and right subplot from simulation.

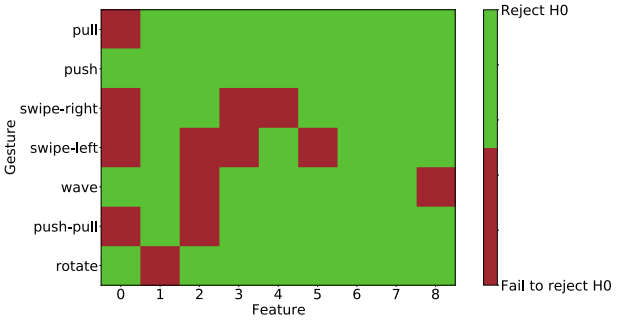

FIGURE 16. Result of Kruskal-Wallis H-test. Green color indicates p-values < 0.05 (i.e. real and synthetic data come from different distributions), and red indicates p-values > 0.05. The Y-axis shows the different gesture types and the X-axis shows the seven features used in the classifier.

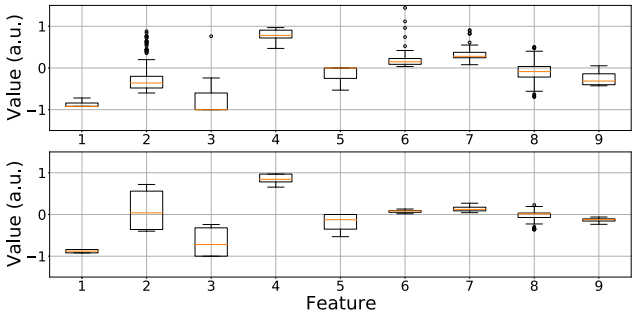

FIGURE 17. Box-plot for each feature of the "pull" gesture. The first row contains samples collected from the Radar and the second row from the simulator.

Kruskall-Wallis test plays a significant role in the supervised learning. To do so, we calculated the average accuracy of the classifier after we excluded the two features (7 and 8) that had no significant p-value (i.e. whose real and synthetic distribution was completely different) for all gesture types. The classifier yielded 74% compared to the original 84.2%. This again indicates that the classifier uses non linear combinations of the input features, and as such it remains invariant to absolute variations of the median of each single feature.

D. DISTINCTION FROM RELATED WORK

Previous work managed to simulate human motion and combine it with a Radar simulator, but did not test a classifier trained with synthetic samples on real data. As a result, a quantitative comparison with a known benchmark is not possible.

We tried to use methods suggested in literature for exporting point-targets from animations but without success. The authors of [14] published source code, which is only suitable for static scenes. Furthermore, we managed to reproduce the method explained in [16] for exporting moving point-targets from Blender using "speed vector pass". However, the point-target information was accurate enough only for 2D movements but not for complex 3D motions.

VI. CONCLUSION

In this work, we presented a method to generate synthetic datasets for arm movements, and we used it for training a ML model for gesture recognition with mmWave technology. The performance of the model was evaluated on real data, which we collected using an experimental sensor, yielding an average accuracy of 84.2%. Therefore, in this work we demonstrated how a novel data-generator like the one we presented here, can contribute in the pre-training phase of a model, as well as for capturing corner cases related to the speed of execution and the position of the subject, that are difficult to reproduce during data collection. To the best of our knowledge, this is the first time that a model trained with synthetic Radar data is tested on real Radar data and achieves such high accuracy.

REFERENCES

[1] J. Lien, N. Gillian, M. E. Karagozler, P. Amihood, C. Schwesig, E. Olson, H. Raja, and I. Poupyrev, "Soli: Ubiquitous gesture sensing with millimeter wave radar," *ACM Trans. Graph.*, vol. 35, no. 4, p. 142, 2016.
[2] S. Wang, J. Song, J. Lien, I. Poupyrev, and O. Hilliges, *Interacting With Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum*. New York, NY, USA: ACM, 2016, pp. 851–860.
[3] Y. Sun, T. Fei, X. Li, A. Warnecke, E. Warsitz, and N. Pohl, "Real-time radar-based gesture detection and recognition built in an edge-computing platform," *IEEE Sensors J.*, vol. 20, no. 18, pp. 10706–10716, Sep. 2020.
[4] A. Ninos, J. Hasch, and T. Zwick, "Real-time macro gesture recognition using efficient empirical feature extraction with millimeter-wave technology," *IEEE Sensors J.*, early access, Apr. 12, 2021, doi: 10.1109/JSEN.2021.3072680.
[5] G. Albuquerque, T. Lowe, and M. Magnor, "Synthetic generation of high-dimensional datasets," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 12, pp. 2317–2324, Dec. 2011.
[6] S. Popic, B. Pavkovic, I. Velikic, and N. Teslic, "Data generators: A short survey of techniques and use cases with focus on testing," in *Proc. IEEE 9th Int. Conf. Consum. Electron. (ICCE-Berlin)*, Sep. 2019, pp. 189–194.
[7] Q. Gao, X. Shen, and W. Niu, "Large-scale synthetic urban dataset for aerial scene understanding," *IEEE Access*, vol. 8, pp. 42131–42140, 2020.
[8] S. D. P. Mendonca, Y. P. D. S. Brito, C. G. R. D. Santos, R. D. A. D. Lima, T. D. O. D. Araujo, and B. S. Meiguins, "Synthetic datasets generator for testing information visualization and machine learning techniques and tools," *IEEE Access*, vol. 8, pp. 82917–82928, 2020.
[9] K. Ishak, N. Appenrodt, J. Dickmann, and C. Waldschmidt, "Human motion training data generation for radar based deep learning applications," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Apr. 2018, pp. 1–4.
[10] R. P. Trommel, "Multi-target human gait classification using deep convolutional neural networks on micro-Doppler spectrograms," in *Proc. Eur. Radar Conf. (EuRAD)*, 2016, pp. 81–84.
[11] R. Boulic, N. M. Thalmann, and D. Thalmann, "A global human walking model with real-time kinematic personification," *Vis. Comput.*, vol. 6, no. 6, pp. 344–358, Nov. 1990.
[12] V. C. Chen, F. Li, S.-S. Ho, and H. Wechsler, "Micro-Doppler effect in radar: Phenomenon, model, and simulation study," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 1, pp. 2–21, Jan. 2006.
[13] M. Stolz, E. Schubert, F. Meinl, M. Kunert, and W. Menzel, "Multi-target reflection point model of cyclists for automotive radar," in *Proc. Eur. Radar Conf. (EURAD)*, Oct. 2017, pp. 94–97.
[14] M. Ouza, M. Ulrich, and B. Yang, "A simple radar simulation tool for 3D objects based on blender," in *Proc. 18th Int. Radar Symp. (IRS)*, Jun. 2017, pp. 1–10.
[15] B. O. Community, "Blender—A 3D modelling and rendering package," Stichting Blender Found., Amsterdam, The Netherlands, Tech. Rep., 2020.
[16] K. Ishak, N. Appenrodt, J. Dickmann, and C. Waldschmidt, "Advanced radar micro-Doppler simulation environment for human motion applications," in *Proc. IEEE Radar Conf. (RadarConf)*, Apr. 2019, pp. 1–6.
[17] A. Gigie, S Rani, A Chowdhury, T Chakravarty, and A. Pal, *An Agile Approach for Human Gesture Detection Using Synthetic Radar Data*. New York, NY, USA: ACM, 2019, pp. 558–564.
[18] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, and R. Kern, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020.
[19] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, and S. J. van der Walt, "SciPy 1.0: Fundamental algorithms for scientific computing in Pyhon," *Nature Methods*, vol. 17, no. 3, pp. 261–272, Feb. 2020.
[20] M. A. Richards, *Fundamentals of Radar Signal Processing*, 2nd ed. New York, NY, USA: McGraw-Hill, 2014.
[21] J. Hasch, E. Topak, R. Schnabel, T. Zwick, R. Weigel, and C. Waldschmidt, "Millimeter-wave technology for automotive radar sensors in the 77 GHz frequency band," *IEEE Trans. Microw. Theory Techn.*, vol. 60, no. 3, pp. 845–860, Mar. 2012.
[22] M. Kronauge and H. Rohling, "New chirp sequence radar waveform," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 4, pp. 2870–2877, Oct. 2014.
[23] G. Hakobyan and B. Yang, "High-performance automotive radar: A review of signal processing algorithms and modulation schemes," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 32–44, Sep. 2019.
[24] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *J. Amer. Stat. Assoc.*, vol. 47, no. 260, pp. 583–621, Dec. 1952.
[25] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, nos. 3–4, p. 591, Dec. 1965.
[26] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, p. 80, Dec. 1945.

ALEXANDROS NINOS (Member, IEEE) received the Diploma in electrical and computer engineering from the National Technical University of Athens, in 2014. He is currently pursuing the Dr.-Ing. degree in electrical engineering with the Institut für Hochfrequenztechnik und Elektronik (IHE), KIT, Germany, and the corporate research of Robert Bosch GmbH, Germany. His research interests include radar signal processing, mm-wave based human motion recognition, and machine learning.

JÜRGEN HASCH (Senior Member, IEEE) received the Dipl.-Ing. and Dr.-Ing. degrees from the University of Stuttgart, Germany, in 1996 and 2007, respectively. He is currently a Senior Expert on RF Technology with the Corporate Research of Robert Bosch GmbH, Renningen, Germany. He has authored more than 40 scientific articles and holds more than 20 patents. His research interests include RF-based sensing technologies and integrated millimeter wave sensors in the 60–240-GHz range. He serves as a reviewer for several journals and conferences. He also serves as the Chair of the IEEE MTT-27 Technical Committee of Connected and Autonomous Systems and a member of ITG-FA Mikrowellentechnik.

MARIO EMILIO PIZANO ALVAREZ, photograph and biography not available at the time of publication.

THOMAS ZWICK (Fellow, IEEE) received the Dipl.-Ing. (M.S.) and Dr.-Ing. (Ph.D.) degrees in electrical engineering from Universität Karlsruhe (TH), Germany, in 1994 and 1999, respectively.

From 1994 to 2001, he was a Research Assistant with the Institut für Höchstfrequenztechnik und Elektronik (IHE), TH. In February 2001, he joined IBM as a Research Staff Member at the IBM T. J. Watson Research Center, Yorktown Heights, NY, USA. From October 2004 to September 2007, he was with Siemens AG, Lindau, Germany. During this period, he managed the RF development team for automotive Radars. In October 2007, he became a Full Professor at the Karlsruhe Institute of Technology (KIT), Germany. He is currently the Director of the Institut für Hochfrequenztechnik und Elektronik (IHE), KIT. He is the author or coauthor of over 400 technical articles and 20 patents. Since 2017, he has been a member of the Heidelberg Academy of Sciences and Humanities. Since 2019, he has also been a member of acatech. Together with his team, he received over 20 best paper awards on international conferences. He has served on the technical program committees (TPC) for several scientific conferences. In 2013, he was the General Chair of the International Workshop on Antenna Technology (iWAT 2013), Karlsruhe, and in 2015 of the IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM), Heidelberg. He was the TPC Chair of the European Microwave Conference (EuMC) 2013. In 2017, he was the General TPC Chair of European Microwave Week. From 2008 until 2015, he has been the President of the Institute for Microwaves and Antennas (IMA). He became selected as a Distinguished Microwave Lecturer for the 2013–2015 period with his lecturer on "QFN Based Packaging Concepts for Millimeter-Wave Transceivers."