Human Motion Training Data Generation for Radar Based Deep Learning Applications

Karim Ishak, Nils Appenrodt, Jürgen Dickmann, and Christian Waldschmidt

Human Motion Training Data Generation for Radar Based Deep Learning Applications

Karim Ishak*, Nils Appenrodt†, Jürgen Dickmann†, and Christian Waldschmidt*
*Institute of Microwave Engineering, Ulm University, 89081 Ulm, Germany
†Daimler AG, Group Research and Advanced Engineering, 89081 Ulm, Germany
Email: karim.ishak@uni-ulm.de

***Abstract*—Radar sensors are utilized for detection and classification purposes in various applications. In order to use deep learning techniques, lots of training data are required. Accordingly, lots of measurements and labelling tasks are then needed. For the purpose of pre-training or examining first ideas before bringing them into reality, synthetic radar data are of great help. In this paper, a workflow for automatically generating radar data of human gestures is presented, starting with creating the desired animations until synthesizing radar data and getting the final required dataset. The dataset could then be used for training deep learning models. A classification scenario applying this workflow is also introduced.**

I. Introduction

Radar sensors are increasingly being used nowadays in diverse fields and for various applications such as autonomous driving [1], hand gesture recognition [2], [3], fall detection [4], and categorization [5]. Many of the applications require lots of radar data, for instance, for classification purposes using deep learning methods. In [2], 2750 data sequences are measured with a radar sensor in order to train and test a Convolutional Neural Network (CNN) with a Recurrent Neural Network (RNN) for the purpose of dynamic gesture recognition. Moreover in [5], 1008 data points of velocity versus time spectrograms are used for training and testing a Deep Convolutional Neural Network (DCNN) in order to classify different human activities. In [6], more than 20000 spectrogram data points are collected using a radar sensor for the classification of multi-target human gait with DCNNs.

Measuring such a huge amount of radar data needs much effort, is time consuming, and may require extra labelling procedures. Moreover, up to our knowledge, there are no publicly available radar datasets—except for the Soli project database [2], which is only for a certain application with no raw radar data available. This is unlike the case of image datasets which are widely available, such as cityscape [7], ImageNet [8], CIFAR-10 [9], and Caltech 101 [10]. For classification purposes based on radars, real data are usually collected using radar sensors. However, synthetic data are also useful for examining ideas and scenarios as a pre-step before exerting all the efforts for actual measurements. Moreover, there are the opportunities of tuning different radar parameters and exploring various options for a radar sensor such as the type of radar, the modulation technique, and the antenna configuration. In addition to that, such synthetic data can be useful for pre-training of deep learning models.

In [6], a kinematic model from [11] is used for synthetic data generation for a DCNN so as to model the human gait. However, this kind of model is only available for a specific kind of motion and cannot be generalized to any desired movement. In [12], another method is described for data synthesis with the aid of Motion Capture (MoCap) data which is originally used for computer games, animation movies, clinical analysis, and others [13]. There are many existing MoCap databases publicly available such as [14]–[16]. Owing to the fact that these databases are captured for specific scenarios in addition to the high price of the professional MoCap systems, the Kinect sensor is proposed for motion capturing [17], [18]. Nevertheless, eventually for generating huge databases of specific scenarios, one has to rely on non-flexible existing databases or use a sensor for collecting data whether it is a radar sensor or any other kind of sensor.

In this paper, an approach which is built above the techniques mentioned in [12], [17], [18] is described for generating a huge amount of data with the assistance of the open source software Blender [19]. First, the different methods for obtaining the required animation data for the simulation scenarios are introduced. Afterwards, radar data synthesis and obtaining the final database are discussed. Finally, a simulation scenario for classification purpose is presented utilizing the whole synthesis chain.

II. Model Architecture

In this section, the chain for synthesizing radar data is presented. The chain is divided into three main steps: animation generation, radar simulator, and generation of the required data.

A. Animation Generation

The human animation is handled in Blender, in which a skeleton for the human body and the required motion are the two main requirements. There are many options for obtaining a human skeleton and ready-made movements. They could be obtained from traditional MoCAP databases such as [14]–[16], [20]. Moreover, the Kinect sensor could also be used for capturing the desired animation as suggested in [17], [18]. The skeleton and animation are imported to Blender using the bvh format. In addition to this, more flexibility is also available, and one could build their own skeleton and/or perform the required movement with the assistance of Blender.

After the desired animation is available in Blender, the next step is to convert this single animation into a larger number of its variants and thus creating more animation data. This is done by writing a Python script integrated in Blender responsible for performing modifications of the current animation scenario. The performed modifications are application specific. However, examples for relevant modifications are varying the inclination of the back and head, varying the angles with which the arms or legs move, varying the inclination of elbows or knees. Moreover, when the focus is on the movement of the arms, extra movements from the legs can be integrated and vice versa in order to represent the irregularities in human motion. This Python script gives a lot of flexibility about the type of modifications and the number of resulting animation files. After that, the animations are exported as bvh animation files.

In order to integrate such resulting animations into a radar simulator, the x, y and z coordinates of the body joints at each motion frame should be obtained. More details about this conversion are found in [12], [21] and a Matlab implementation for such conversion is in [15].

B. Radar Simulator

At this step, the required radar type should be used to process the x, y and z coordinates resulting from the previous step. It is flexible which radar type is applied for the rest of the simulation. In this paper, a chirp sequence Frequency Modulated Continuous Wave (FMCW) radar is utilized. A suitable interpolation for the required sampling frequency is performed on the data for further processing [12]. The desired radar location is specified and the distance $R(b,i)$ to each body part b at time step i is calculated. At this point, there is another possibility for generating even more and more data by changing the position of the radar sensor. This gives the possibility of examining the target from different ranges and various aspect angles. Additive white Gaussian noise with different values of noise power also increases the variations. The generation of the Intermediate Frequency (IF) signals follows as in [22], [23]. In [22], the IF signal corresponding to a single ramp l for a target at one time step is given by

$$S_{\mathrm{IF}}(i,l) = \mathrm{e}^{\mathrm{j}2\pi\left[\frac{2f_{\mathrm{c}}(d+v_{\mathrm{r}}T_{\mathrm{RRI}}l)}{c_0}+\left(\frac{2f_{\mathrm{c}}v_{\mathrm{r}}}{c_0}+\frac{2Bd}{Tc_0}\right)t(i)\right]}\,, \quad (1)$$

where

d:	distance from the radar to the target
v_{r}:	radial velocity of the target relative to the radar
f_{c}:	carrier frequency
T_{RRI}:	ramp repetition interval
c_0:	speed of light in vacuum
T:	chirp time
B:	bandwidth.

In [22], [23], the velocity of the target is assumed to be constant. In our case, the velocity does not have to be constant, and it is already contained inside the range information, which varies from time step to the next, as follows:

$$R(b,i) = R(b,i-1) + v(b,i)\Delta t\,, \quad (2)$$

where $v(b,i)$ is the radial velocity of body part b at time step i, and Δt is the difference between consecutive time steps. The resulting IF signal is

$$S_{\mathrm{IF}}(b,i) = \mathrm{e}^{\mathrm{j}2\pi\left[\frac{2f_{\mathrm{c}}R(b,i)}{c_0}+\frac{2BR(b,i)t(i)}{Tc_0}\right]}. \quad (3)$$

This time signal is then divided into ramps according to the required radar parameters and further processed as in [22], [23]. The range peak is then found at

$$p = \frac{2BR(b,i-1)}{Tc_0} + \frac{2Bv(b,i)\Delta t}{Tc_0}\,, \quad (4)$$

such that the second term could be neglected within one ramp similarly as in [23].

The simulated IF signals of all the body parts are then added up to result in the total IF time domain signal. For a more realistic simulation, two more effects are taken into consideration which are the Radar Cross Section (RCS) of the body parts and the shadowing effects. In order to obtain an approximate RCS value, the body parts are modelled as ellipsoids [24]. The shadowing effects could then be accounted for in the case of scenarios involving parts of the body that are shadowed by other ones.

C. Generation of the Required Data

The simulated raw radar data are then used for obtaining the synthesized dataset. The required data for a classification task could be of various types. The raw radar data could be utilized in the time domain without any further processing [25]. The most commonly adopted data type is the micro-Doppler spectrogram showing the velocity or the Doppler frequency versus time [3], [5], [6], [26]. Moreover, the range versus velocity diagrams at consecutive time steps are used in combination with RNNs [2]. Furthermore, the range profile, the velocity profile, or the range versus time diagram could also be utilized [27]. In addition to the data based on range and velocity, angular-based data are also beneficial.

III. Simulation and Results

Four scenarios including a person moving one arm are considered, such that the radar is facing the person from the side corresponding to the moving arm. The four scenarios are shown in Fig. 1. These scenarios are typical examples for autonomous driving regarding situations that occur on the streets when a policeman is regulating the traffic. 2000 instances are generated per scenario such that there are variations in the Signal-to-Noise Ratio (SNR) by adding white Gaussian noise, the position of the radar sensor, the inclination of the upper body of the person, the size of the person, and the exact starting and ending positions and time instances of the arm. Moreover, random movements of the legs and elbows are taken into account. The combinations between all these variations are randomized. RCS and basic shadowing effects are also considered for a more reasonable simulation.

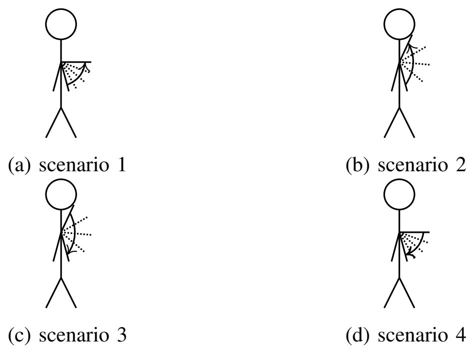

Fig. 1. The four simulated scenarios. In the first scenario, the person is raising the arm from beside his legs to an almost perpendicular position to the body. In the second scenario the arm is lifted up towards the head. The third and fourth scenarios involve the same movements as the second and first scenarios, respectively, but in a reversed direction.

The block diagram in Fig. 2 shows the workflow performed to obtain the dataset. First, the animation of each scenario is performed in Blender using a human skeleton from [20]. Afterwards, a Python script is written in Blender in order to introduce the variations for obtaining more data points. The output from Blender is in bvh file format which is then converted so that the output is the position data for each body part at consecutive time points. For each body part, the simulated radar signal is obtained with the required radar parameters and for various SNRs and radar positions. The final radar signal is calculated by summing up the radar signals of the contributing body parts. Subsequently, the radar signal is processed using Fourier transform and Short Time Fourier Transform (STFT) to obtain the micro-Doppler spectrograms which form the needed dataset.

The velocity versus time spectrograms are converted to grey-scale images of size (70×100) pixels, which are then used for classification. Some images of the micro-Doppler spectrograms for the first scenario are shown in Fig. 3. The 8000 generated data points are divided such that 70% of them are used for the purpose of training and the rest are for testing. The network architecture is shown in Fig. 4. The neural network has only two hidden layers, however, this is enough to show that the workflow is realistic. Each image is first flattened into a vector and then fed into the network. The activation function used in all the layers except for the output layer is the Rectified Linear Unit (ReLU) and a softmax activation is used at the output layer. No regularization is done for this network. Adam optimizer [28] is applied with a learning rate of $5 \cdot 10^{-5}$ and the number of epochs is set to 6000. The resulting accuracy on the test dataset is 72.3% and the confusion matrix is shown in Tab. I. The confusion matrix shows a good classification of the first and fourth scenarios. However, there is difficulty in differentiating between the second and third scenarios, which is expected because they have a similar velocity profile. These results show that such a simulation framework is beneficial for getting first results on the classification ability of different architectures. It also

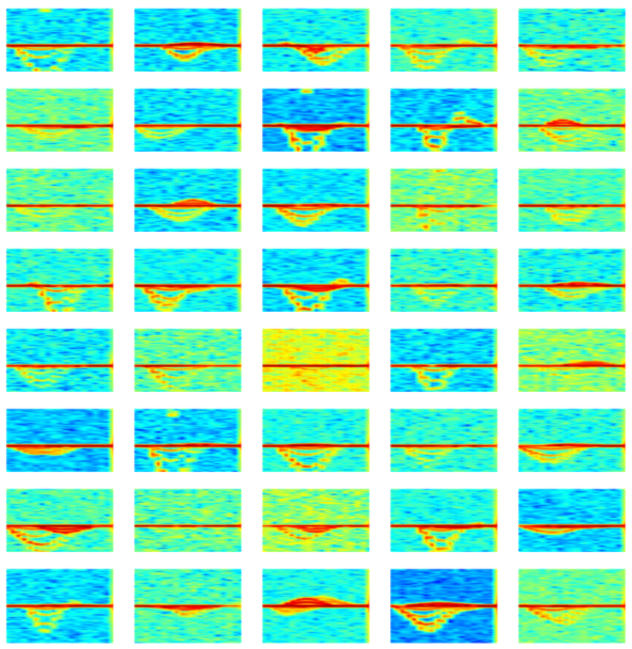

Fig. 3. Some micro-Doppler spectrograms from the first scenario.

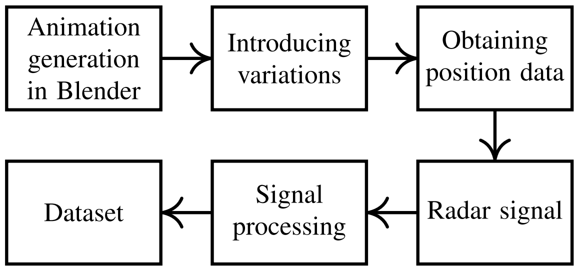

Fig. 2. A block diagram for the workflow.

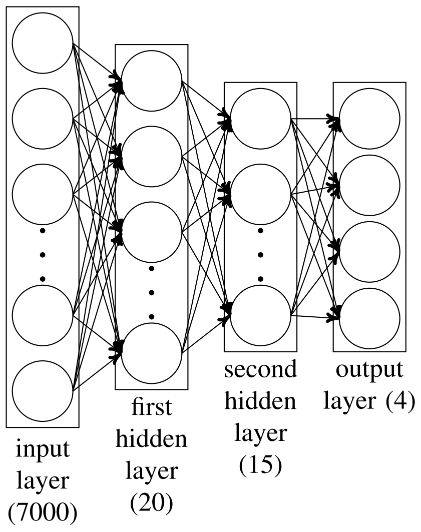

Fig. 4. The neural network architecture.

TABLE I
THE CONFUSION MATRIX FOR THE CLASSIFICATION TASK. THE NETWORK IS ABLE TO DIFFERENTIATE BETWEEN THE FIRST AND FOURTH SCENARIOS. HOWEVER, AS EXPECTED, THE SECOND AND THIRD SCENARIOS ARE DIFFICULT TO BE CORRECTLY CLASSIFIED DUE TO THE SIMILARITY IN THE VELOCITY PROFILE.

	scenario 1	scenario 2	scenario 3	scenario 4
scenario 1	547	26	9	18
scenario 2	42	336	197	25
scenario 3	31	214	323	32
scenario 4	20	27	23	530

gives a better look about the chances of differentiating between the scenarios in a certain domain—micro-Doppler domain in our case. Actually, the huge amount of obtained data opens the opportunity to perform realistic tests on them, utilizing a simulated radar with tuned parameters, and at the same time saves the measurement efforts each time a change is needed.

IV. Conclusion

A workflow for an automatic generation of a huge amount of radar simulated data for human motion is presented. Different methods for increasing the number of synthesized data after generating the required animations are suggested. The radar data are then converted to the required form of data representation for further processing. Deep learning techniques then make use of the generated datasets. In addition to that, the presented workflow is flexible and can be modified at different points according to the application.

References

[1] C. Waldschmidt and H. Meinel, "Future trends and directions in radar concerning the application for autonomous driving," in *European Radar Conference (EuRAD), 11th*. IEEE, 2014, pp. 416–419.

[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," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 2016, pp. 851–860.

[3] Y. Kim and B. Toomajian, "Hand Gesture Recognition Using Micro-Doppler Signatures With Convolutional Neural Network," *IEEE Access*, vol. 4, pp. 7125–7130, 2016.

[4] L. Liu, M. Popescu, M. Skubic, M. Rantz, T. Yardibi, and P. Cuddihy, "Automatic Fall Detection Based on Doppler Radar Motion Signature," in *5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*. IEEE, 2011, pp. 222–225.

[5] Y. Kim and T. Moon, "Human Detection and Activity Classification Based on Micro-Doppler Signatures Using Deep Convolutional Neural Networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 1, pp. 8–12, 2016.

[6] R. Trommel, R. Harmanny, L. Cifola, and J. Driessen, "Multi-target Human Gait Classification Using Deep Convolutional Neural Networks on Micro-Doppler Spectrograms," in *European Radar Conference (EuRAD)*. IEEE, 2016, pp. 81–84.

[7] Cityscape website. [Online]. Available: https://www.cityscapes-dataset.com [Accessed: 21.02.2018]

[8] ImageNet website. [Online]. Available: http://image-net.org [Accessed: 21.02.2018]

[9] CIFAR-10 website. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html [Accessed: 28.11.2017]

[10] Caltech 101 website. [Online]. Available: http://www.vision.caltech.edu/Image_Datasets/Caltech101 [Accessed: 21.02.2018]

[11] R. Boulic, N. M. Thalmann, and D. Thalmann, "A global human walking model with real-time kinematic personification," *The visual computer*, vol. 6, no. 6, pp. 344–358, 1990.

[12] S. S. Ram and H. Ling, "Simulation of Human MicroDopplers Using Computer Animation Data," in *Radar Conference, RADAR'08*. IEEE, 2008, pp. 1–6.

[13] T. B. Moeslund and E. Granum, "A Survey of Computer Vision-Based Human Motion Capture," *Computer vision and image understanding*, vol. 81, no. 3, pp. 231–268, 2001.

[14] The CMU MoCap database website. [Online]. Available: http://mocap.cs.cmu.edu/ [Accessed: 21.02.2018]

[15] The HDM05 MoCap database website. [Online]. Available: http://resources.mpi-inf.mpg.de/HDM05/ [Accessed: 21.02.2018]

[16] The Ohio MoCap database website. [Online]. Available: https://accad.osu.edu/research/motion-lab/system-data [Accessed: 21.02.2018]

[17] B. Erol, C. Karabacak, S. Z. Gurbuz, and A. C. Gurbuz, "Simulation of Human Micro-Doppler Signatures with Kinect Sensor," in *Radar Conference*. IEEE, 2014, pp. 863–868.

[18] B. Erol and S. Z. Gurbuz, "A Kinect-Based Human Micro-Doppler Simulator," *IEEE Aerospace and Electronic Systems Magazine*, vol. 30, no. 5, pp. 6–17, 2015.

[19] Blender website. [Online]. Available: https://www.blender.org/ [Accessed: 21.02.2018]

[20] Mixamo website. [Online]. Available: https://www.mixamo.com/ [Accessed: 21.02.2018]

[21] M. Meredith, S. Maddock *et al.*, "Motion Capture File Formats Explained," *Department of Computer Science, University of Sheffield*, 2001.

[22] V. Winkler, "Range Doppler Detection for automotive FMCW Radars," in *Microwave Conference, European*. IEEE, 2007, pp. 1445–1448.

[23] C. Schroeder and H. Rohling, "X-Band FMCW Radar System with Variable Chirp Duration," in *Radar Conference*. IEEE, 2010, pp. 1255–1259.

[24] V. C. Chen, *The Micro-Doppler Effect in Radar*. Artech House, 2011.

[25] H.-S. Yeo, G. Flamich, P. Schrempf, D. Harris-Birtill, and A. Quigley, "RadarCat: Radar Categorization for Input & Interaction," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 2016, pp. 833–841.

[26] J. Park, R. J. Javier, T. Moon, and Y. Kim, "Micro-Doppler Based Classification of Human Aquatic Activities via Transfer Learning of Convolutional Neural Networks," *Sensors*, vol. 16, no. 12, p. 1990, 2016.

[27] 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 Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 142, 2016.

[28] D. Kingma and J. Ba, "ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION," *arXiv preprint arXiv:1412.6980*, 2014.