Frequency Modulated Continuous Wave Radar-Based Navigation Algorithm using Artificial Neural Network for Autonomous Driving

Jakob Valtl[1*#] and Vadim Issakov[2*#]

***Abstract*— Autonomous driving is a highly complex task, which involves the use of numerous sensors and various algorithms. Testing of algorithms is difficult and therefore mostly done in simulations. Radar technology will play a key part due to various advantages. In this paper we present a solution to one aspect of autonomous driving, which is the development of a detection algorithm on a moving platform, which is capable of tracking and sending the commands to follow a preceding object, by means of sensor data from a low power 60 GHz Frequency Modulated Continuous Wave (FMCW) radar. The moving platform is based on a miniaturized autonomous vehicle that is used for data gathering as well as algorithm evaluation. To the best of the author's knowledge, this is the first time that processing of radar data via Deep Convolutional Neural Networks (DCNN) for navigation purposes is performed in real time on the edge device operating in a real world environment and not simulative.**

***Index Terms*— radar applications, artificial neural networks, edge computing, real-time systems, autonomous vehicles, object recognition**

I. Introduction

Autonomous mobility and its components are an emerging research field. Numerous researchers are working currently on topics related to Artificial Intelligence (AI) as well as system strategies to enable an affordable, safe and secure fully-autonomous diving (level 5). Different sensor types offer various sensor-specific benefits as well as limitations.

Especially radar sensors are of great interest for autonomous driving thanks to their capability to operate even under bad weather conditions. Radar sensors do not have mechanically moving parts, unlike other sensors that give a depth information e.g. LiDAR, thus being physically more robust. Last but not least, the price advantage of a radar system in comparison to a LiDAR system might give radar sensors an advantage in future autonomous vehicles.

Nevertheless, there are still some challenges that need to be resolved, like the complicated and time consuming data processing. In this paper we address the aforementioned challenge by using a fast neural network based approach. The evaluation of incoming sensor data has to happen in a predefined period of time to ensure a response of an autonomous vehicle due to a changing environment. If the time constrains can not be met, the safety of the vehicle as well as surrounding pedestrians or other traffic participants can not be ensured. AI based evaluation methods are one possibility to deal with the vast amount of data in the required time constrains.

Mobile data bandwidth limits and latency issues that clash with the aforementioned time constrains only admit data processing to run on the edge devices, but do not allow a centralized data processing solution. To keep processing demand on the edge devices low, AI algorithms can be advantageous compared to traditional data processing because they require less hardware resources as they can easily be run on specialized hardware. [5]

Challenges that arise from the use of AI components are their black box behaviour and their vulnerability towards adversarial attacks. Both can lead to unforeseen reactions of the vehicle. So far many investigations have been performed using simulation tools [6], but testing, especially under real world conditions is crucial for further development.

Therefore, the proposed platform and the navigation algorithm can be used as a foundation for future research. We focused on a specific use case in the automotive domain, where we developed a navigation algorithm and tested it in the field. The use case "follower" describes a scenario in which a self driving car follows a preceding car fully autonomously. The track for data recording and algorithm evaluation of the "follower" use case is shown in Fig. 1.

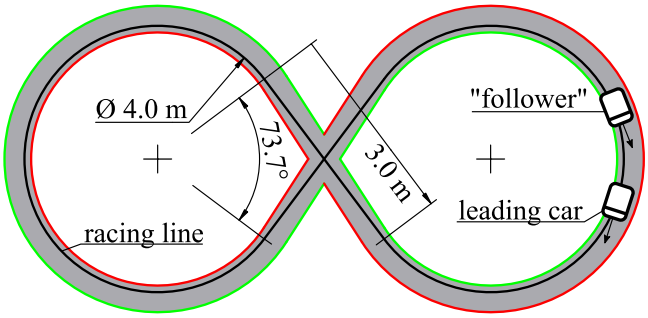

Fig. 1. Layout of the training data track.

Many research efforts have been reported in the visual domain regarding test environments in the form of reduced size vehicles. Though other test environments are vision based, they focus on a variety of aspects like safety, security or efficiency. An example of reinforcement learning was implemented by Qi z. et al. [8] on a DonkeyCar [11] platform. T. Do et al. [10] used Deep Convolutional Neural Networks (DCNN) on a monocular vision-based prototype. Both report high accuracy rates on the steering and speed decision. J. Newman et al. [7] introduced a low budged self

*Infineon Technologies AG, Germany
#Technische Universität Braunschweig, Germany
[1]jakob.valtl@infineon.com
[1]V.Issakov@tu-braunschweig.de

driving model car with the focus of maneuvering through an indoor race track.

Traditional platooning approaches [12], [4] that are similar to the ”follower” use case rely on the communication between vehicles and use radar sensor input only for the distance detection, while we also extract the angular information of the preceding vehicle, needed for the steering from the radar data.

Data processing of radar data is a major topic in the research community with numerous publications for different purposes, reaching from gesture recognition [5] and air writing [1] to people counting [3] and human activity classification [9]. Even small scale movements like vital signs [2] can be detected by this technology. All these applications have in common that they are performed indoor in a shielded environment and with a stationary radar sensor. Installation in a vehicle adds additional challenges due to an outdoor environment with a non-stationary sensor. Also the tough real-time constrains implicate further challenges to the speed of evaluation.

II. Design of Experiment

In the ”follower” use case an autonomous miniaturized vehicle follows a preceding vehicle by means of an algorithm described in Section III that evaluates the information coming only from the radar sensor.

A. Track

The quality of a Neural Network (NN) is, apart from its architecture, highly dependent on the amount and the quality of the data it is trained with. In our use case (”follower”), ideally all possible curvatures to both sides as well as straight driving have to be learned just as the speed of the vehicle. On these grounds we designed a track shown in Fig. 1, which in theory has two curvatures that represent 10 meters (38.5%) each while the part of the track that goes straight aggregates to 6 meters (23%). This way both curvatures are evenly represented in the recorded data set that is used for training.

The training data is gathered by maneuvering the vehicle by means of a remote controller various laps through the track while recording the user input as well as the sensors signals. The histogram in Fig. 2 shows the distribution of the human steering input, for various laps through the track. The steering value ranges from -1 to 1, representing the deflection of the front wheels. It can be noticed that the steering is evenly distributed among both curvatures though the curvatures do not show only one deflection, as could be assumed due to the fixed curvatures of the turns, but rather smooth transitions.

The underrepresentation (14% instead of 23%) of straight driving which is indicated by a steering value of 0.0 is a result of faster driving during the straight part of the track, as well as minor corrections in heading during the straight driving part of the track.

The colors of the lines in Fig. 1 represent the colors of cones used for indication of the track in reality. Fig. 3 shows

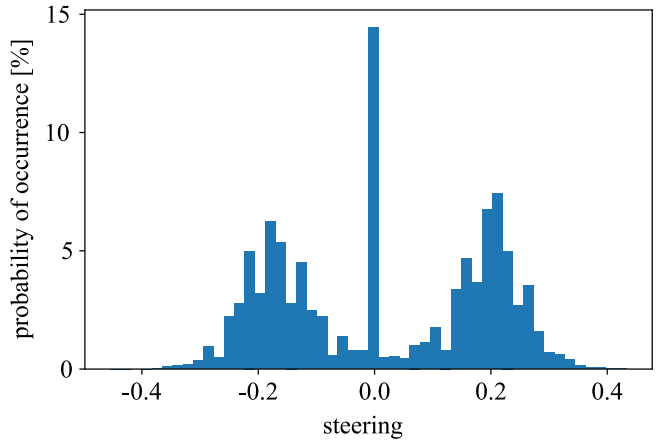

Fig. 2. Histogram of the human controlled steering input during training over a total of 138 533 frames during 46 laps. Where -1 represents a maximum deflection of the steering towards the left and 1 towards the right side.

Fig. 3. Photo of track used for data gathering for the ”follower” use case.

an image of the track with the two miniaturized vehicles. The cones are not detectable by the radar as their signal is lower than the general noise. They serve as guide lines for the human operator during the data gathering and in case of autonomous operation on a visual base, for the camera as landmarks.

B. Data Gathering and Evaluation Platform

The autonomous platform used for the data gathering as well as the algorithm evaluation of the vehicle is described in detail in [13]. Major changes are the upgrade of the processing unit to a Raspberry Pi 4 and changes in the software. The radar used in the platform is a Frequency Modulated Continuous Wave (FMCW) radar with one transmit and three receive channels with antenna in package, as displayed in Fig. 4.

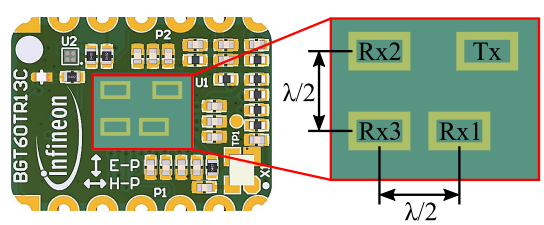

Fig. 4. Infineon’s BGT60TR13C shield with antenna configuration.

Only the two horizontal antennas (Rx3 and Rx1), as shown in Fig. 4 are used. The center frequency is 60 GHz and

the bandwidth is set to 4 GHz. Each frame consists of 64 chirps, while for each chirp 128 samples are recorded at a sample rate of 2 MHz. The chosen frame rate during the data recording is 30 Hz for both the radar data recording and the vehicle's actuator actualization. During the evaluation of the DCNN the frame rate is set to 20 Hz due to the extra processing time required by the neural network for the data evaluation, computing the action for the actuators.

C. Dataset

With the earlier described platform a dataset is recorded on the track specified above. A total of 153 648 frames are recorded during 50 laps. The dataset is split in a validation (15 115 frames) and a train/test (138 533 frames) subset. During the recording the radar data as well as the user input is stored. During the recording the vehicle moves with an average speed of 253 mm/sec.

III. Navigation Algorithm

In order to maneuver the miniaturized vehicle autonomously in the "follower" use case, we have to extract the following information from the radar data: distance, speed, angle and heading of the preceding vehicle. The autonomous control based on a DCNN navigates the platform by the output coming from the radar pipeline.

A. Radar Pipeline

The raw radar data gathered with a FMCW-radar is structured as a three dimensional data cube for each frame, where the dimensions represent i) the amount of antennas ii) the amount of chirps per frame and iii) the amount of samples per chirp.

Traditional approaches for radar processing make use of the Fast Fourier Transformation (FFT) over the fast time dimension to get a map of the intensity over the range. Followed by a second FFT in the slow time dimension, which results in a Range Doppler Map (RDM) for each antenna, representing the complex-valued velocity in the different range bins for each antenna. From the RDMs of antennas that are oriented in a $\lambda/2$ distance horizontally or vertically the azimuth as well as the elevation of localized targets can be calculated and thus the Range Angle Map (RAM) constructed. In our use case we limit the amount of available targets in the radar's field of vision to one, being the preceding vehicle.

The construction of the RAM from the RDMs is described by the following formulas

$$\mathrm{res}_{a,k} = e^{(-1)^k \cdot j \cdot \frac{\Delta d}{2} \cdot 2\pi \cdot \frac{c}{f} \cdot \sin\left(\frac{a}{a_{\mathrm{res}}}(a_{\max} - a_{\min}) + a_{\min}\right)}, \qquad (1)$$

where $res_{a,k}$ describes a complex-valued array needed for the construction of RAM as described in Equation 3. The angular information is represented by a which iterates over possible angles of the detected targets reaching from a minimum angle of $a_{\min}$ to a maximum angle of $a_{\max}$ with a resolution of a_{res}. The speed of light is c and the center carrier frequency is f, while Δd represents the distance of the antennas k. The imaginary unit is chosen to j and e describes the Euler's formula representation.

$$\mathrm{cov}_{r,k,l} = \sum_d \mathrm{RDM}_{k,d,r} \cdot \overline{\mathrm{RDM}_{l,d,r}} \ , \qquad (2)$$

is a covariance matrix holding the information of the angle of arrival of the target detected in each range bin. The indices k and l represent the antennas in question, while d iterates over the Doppler- and r the range-dimension. The complex-conjugate of a complex number is described by a line above the complex number.

$$\mathrm{RAM}_{a,r} = \sum_k \sum_l \overline{\mathrm{res}_{a,k}} \cdot \mathrm{cov}^{+}_{r,k,l} \cdot \mathrm{res}_{a,l} \ , \qquad (3)$$

finally is the Range Angle Map, where the indices follow the nomenclature of the equations before mentioned. The only additional operator is $^+$, which represents the Moore-Penrose pseudo-inverse of a matrix using its singular-value decomposition.

To reduce the amount of data forwarded to the neural network and increase the processing speed, we do not calculate the RAM which would be the traditional approach but we introduce a new method of only considering $Im(cov_{r,0,1})$.

B. Machine Learning Approach

A goal in our work, was to employ a small network topology to reduce computational needs and power so that system can respond with a maximum latency of 50 milliseconds. Thereby we realized that all the information needed to detect the preceding vehicle and thus the adequate orders for the actuators, are already present in one single sub-vector of cov calculated in Equation 2. In exquisite detail $Im(cov_{r,0,1})$ is a vector of length r, the range bins (in our case 64), where its values represent the accumulated angle of arrival of recognized targets in the range bin. A snippet of the gathered data including $Im(cov_{r,0,1})$ is shown in Fig. 6 in the results section. This vector is normalized and then used as an input for the neural network and will from now on be referred to as cov_r. The output of the DCNN are the commands for the steering and the throttle of the platform.

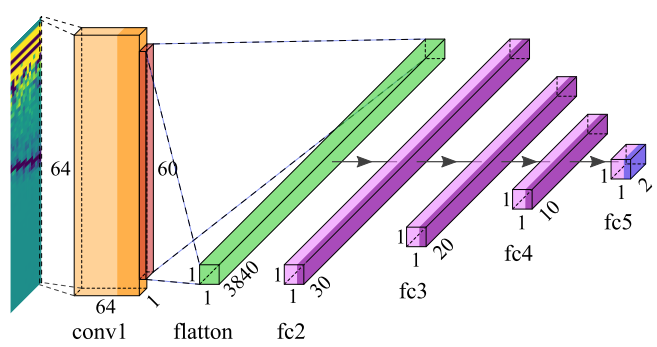

Fig. 5. Architecture of the deep convolutional neural network.

The selected DCNN design is shown in Fig. 5. The first layer is a 1D-convolution (conv) layer to cope with the effects originating from the quantization of the samples for each

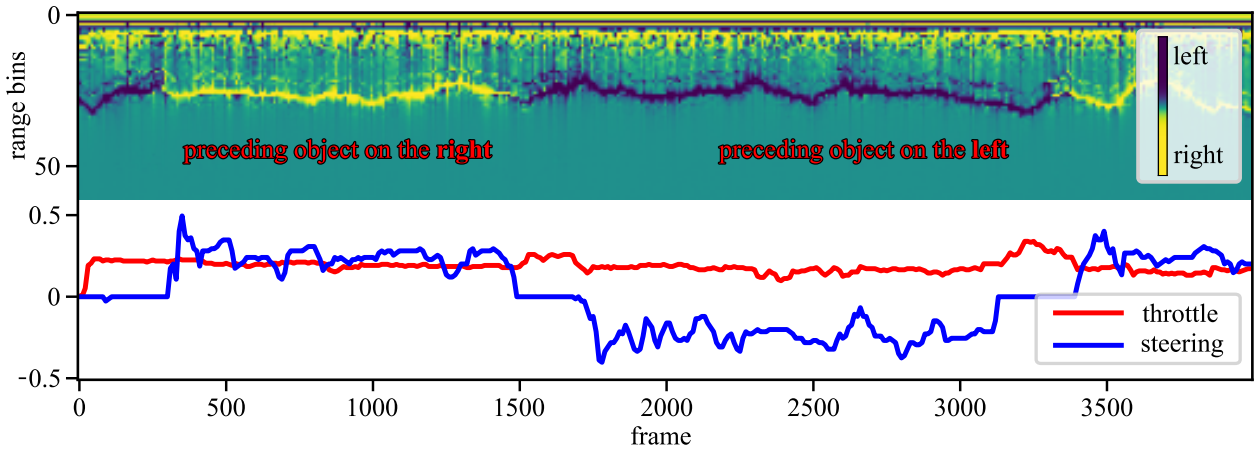

Fig. 6. Above: cov_r from one lap through the track, where the range bins reflect the distance to the preceding vehicle. The color is the visual representation of the value in each bin, depicting the relative angle of the received radar signal in each bin;
Below: the corresponding input from the human operator during the recording. The correlation between steering and the angle information in the radar signal, as well as the distance of the main reflection and the throttle is shown.

chirp that result in the range bins. We chose a kernel of size 5 and a stride of 1. No padding is applied and the amount of filters is chosen to 64. After the initial convolution layer its output is flattened. The following fully connected (fc) layers learn the linkage between the extracted features from cov_r and the outputs needed for the autonomous control of the vehicle. Each layer, be it conv or fc, is followed by a rectified linear unit activation function. Finally the dense output layer is performed with a linear activation function. The hyperparameters chosen for the DCNN are given in the following table.

TABLE I
HYPERPARAMETERS

hyper parameter	value
optimizer	adam
loss function	mean absolute error
learning rate	$5 \cdot 10^{-4}$
decay	$5 \cdot 10^{-7}$
early stopping	true
minimum delta	10^{-10}
patience	5
batch size	209
epochs	150

IV. Experimental Results

The evaluation of the dataset described in Section II-C shows that the values of cov_r are in the range of [-2 410; 25 792] with a mean of 388 and a median of 0.033. cov_r is normalized by applying $\tanh(\frac{cov_r}{10})$ in order to enhance the Signal to Noise Ratio (SNR), limit outliers and for the better interpretability of the DCNN. The division factor of 10 is experimentally determined as it gives the best results during the DCNN training process.

Fig. 6 shows a series of 4 000 consecutive frames from the validation subset that represent one entire lap through the track. In the upper part cov_r is plotted while in the lower part the corresponding value of steering and throttle, originating from the human operator during the recording, is depicted.

The final size of the model (h5 version) with 116 466 trainable parameters is 1.4 MB which is small enough to be running on the Raspberry Pi under the given timing constrains. Training of the model with the dataset described in Section II-C takes about 50 seconds on an office laptop for each epoch.

The trained model is evaluated on the validation subset where the networks prediction deviates by 0.038 on average for the steering and by 0.008 for the throttle. Evaluation of the model with the demonstrator and not prerecorded data gives satisfying results.

The trained model reaches its limits if the leading vehicle operates outside the trained conditions, especially smaller turns than the 4 meters in diameter lead to unexpected behaviour of the "follower". The utilization of the processors is on average about 85% at a processing frame rate of 20 Hz.

To benchmark the presented approach we evaluated the recorded data with a more classical approach where the entire RAM is calculated. Filtered by a moving target indicator with a small update rate the constant ground noise is suppressed. The throttle and steering is directly determined by spotting the maximum value in the filtered RAM and aplying a linear mapping.

A third approach is to train a NN with the entire RAM similar to our proposed approach. This results in a larger network size and more computation recuses with no noticeable improvement in deviation reduction. The results of the benchmarks are given in TABLE II.

V. Conclusion

To our knowledge it is the first time that a low power 60 GHz FMCW radar sensor is successfully employed in the evaluation of the angle of arrival in a moving environment of a preceding obstacle in real time on an edge device and enables the car to follow it autonomously. We make use of

TABLE II
BENCHMARK RESULTS

	cov_r NN	**RAM NN**	**RAM max value**
frame execution time [seconds]	0.037	0.051	0.040
deviation of steering	0.038	0.037	0.151
deviation of throttle	0.008	0.009	0.013
NN size [MB]	1.4	16.6	-

cov_r, formulated by us, for angular estimation, with the benefit of being a light weight representation of the radar data. We demonstrate that a AI empowered platooning vehicle can be build with inexpensive off the shelf components. This brings us one step closer to a real sized autonomous car with the "follower" use case being independent of weather sensible sensors.

VI. FUTURE WORK

The very low deviation of the throttle compared to the steering can be explained by the fact that during the recording as well as during testing the speed was always more or less constant. Starting and stopping as well as accelerations were not evaluated. Another simplification was a speed invariant distance control as well as the assumption of exactly one target being present in the range of the radar. A future recording will include various different scenarios.

Another task is to further optimize the pre-processing of the radar data in order to reduce processor utilization. Finally, the security of the radar data and its evaluation algorithms will be tested regarding possible threats like adversarial attacks or jamming, as the radar sensor data is safety critical in this scenario, especially if applied in a future, real sized vehicle.

ACKNOWLEDGMENT

This work is a result of a collaboration between the projects "KI-Flex" (project number 16ES1027), funded by the German Federal Ministry of Education and Research (BMBF) within the founding program Microelectronic from Germany innovation driver, and the project "TEACHING" (project number 871385) founded by the Horizon 2020 program.

REFERENCES

[1] M. Arsalan, A. Santra, and V. Issakov. "Radar Trajectory-based Air-Writing Recognition using Temporal Convolutional Network". In: *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 2020, pp. 1454–1459. DOI: 10.1109/ICMLA51294.2020.00225.

[2] M. Arsalan, A. Santra, and C. Will. "Improved Contactless Heartbeat Estimation in FMCW Radar via Kalman Filter Tracking". In: *IEEE Sensors Letters* 4.5 (2020), pp. 1–4. DOI: 10.1109/LSENS.2020.2983706.

[3] Cem Yusuf Aydogdu et al. "Multi-modal cross learning for improved people counting using short-range FMCW radar". In: *2020 IEEE International Radar Conference (RADAR)*. IEEE. 2020, pp. 250–255.

[4] Eric Chan. "SARTRE Automated Platooning Vehicles". In: June 2016, pp. 137–150. ISBN: 9781786300270. DOI: 10.1002/9781119307785.ch10.

[5] Mateusz Chmurski and Mariusz Zubert. "Novel Radar-based Gesture Recognition System using Optimized CNN-LSTM Deep Neural Network for Low-power Microcomputer Platform". In: (2021).

[6] Francesco Concas et al. "Validation Frameworks for Self-Driving Vehicles: A Survey". In: *Smart Cities: A Data Analytics Perspective*. Springer, 2021, pp. 197–212.

[7] J. Newman, Z. Sun, and D. -J. Lee. "Self-Driving Cars: A Platform for Learning and Research". In: *2020 Intermountain Engineering, Technology and Computing (IETC)*. 2020, pp. 1–5. DOI: 10.1109/IETC47856.2020.9249142.

[8] Zhang Qi and Du Tao. "Self-driving scale car trained by Deep reinforcement Learning." In: *ArXiv* abs/1909.03467 (2019).

[9] Michael Stephan et al. "Radar Image Reconstruction from Raw ADC Data using Parametric Variational Autoencoder with Domain Adaptation". In: *25th International Conference on Pattern Recognition*. IEEE. 2021.

[10] Q. Dang T. Do M. Duong and M. Le. "Real-Time Self-Driving Car Navigation Using Deep Neural Network". In: *2018 4th International Conference on Green Technology and Sustainable Development (GTSD), Ho Chi Minh City, 2018* abs/1909.03467 (2018), pp. 7–12. DOI: 10.1109/GTSD.2018.8595590.

[11] *The DonkeyCar Website*. (Accessed: 10.03.2021). URL: https://www.donkeycar.com/.

[12] Sadayuki Tsugawa, Shin Kato, and Keiji Aoki. "An automated truck platoon for energy saving". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2011, pp. 4109–4114. DOI: 10.1109/IROS.2011.6094549.

[13] J. Valtl et al. "Autonomous Platform based on Small-Scale Car for Versatile Data Collection and Algorithm Verification". In: *2021 25th IEEE International Conference on Pattern Recognition(ICPR)*. 2020.