# Improving Millimeter Wave Radar Perception with Deep Learning

Junfeng Guan Sohrab Madani
ECE 544 Project Report
jguan8@illinois.edu smadani2@illinois.edu

## 1. Introduction and Project Descriptions

Since the past few years, AI-Powered autonomy revolution in the automotive industry has attracted great attention worldwide. It is believed that in the not-too-distant future, fullyautonomous vehicles will be the norm rather than the exception, redefining mobility in our daily lives. With deep learning widely applied on sensor data, self-driving cars are able to localize and map objects, understand the environment, and make correct decisions. As the most fundamental task, previous works have demonstrated accurate object detection and classification, but they are limited to data obtained from LiDARs and cameras. These optical sensors have high imaging resolution, but they naturally fail in low visibility conditions such as fog, rain, and snow, because light beams are narrower than water droplets and snowflakes. [9] This fundamental limitation of optical sensors is one of the major roadblocks to achieving the 5th SAE level of full automation. [1] On the contrary, Radar wave propagates through small particles and can provide an alternate imaging solution in such inclement weather. Besides, radar can also directly measure the velocity of vehicles based on the doppler shift of the reflected signal instead of going through cluster tracking among frames like LiDAR. With the additional realtime speed information, AI can potentially have a better perception than human drivers.

Although the low resolution of traditional automotive radar overshadows its advantages, the advent of Millimeter-wave (mmWave) technology makes it possible to have a reliable 3D imaging system in inclement weather with relatively higher resolution. Along with good propagation characteristics, mmWave also provides much wider bandwidth and enables miniature-sized large-aperture antenna arrays, which improve distance and angle measurement respectively. Previous work has demonstrates sub-centimeter level imaging resolution for short range objects [10], but the fundamental difference in wavelength between radar (5 mm) and LiDAR (905 nm) cannot be easily overcome. Moreover, radar images are not as readable to non-experts. Therefore, in order to convince the automotive industry and general public that mmWave is a reliable imaging solution for autonomous driving, we should try to generate radar images targeting the resolution comparable to LiDAR, and make them more perceptually intuitive to people.

In this project, we propose to develop high-resolution and reliable imaging techniques for self-driving cars with mmWave radar. Specifically training neural networks to enhance the low-resolution and unreadable radar images to be similar to the LiDAR point cloud, which has been extensively and successfully used for self-driving perception. Eventually enable various crucial vision applications for autonomous vehicles like lane detection, image mapping, localization, and object identification with mmWave radar data only. To do this, we have to overcome challenges analyzed in the following section 2.

## 2. Challenges

### 2.1. Radar Imaging Primer

Radar generates images of objects by localizing the cluster of point reflectors that forms the object. 3D space imaging requires mapping point reflectors to voxels in a spherical coordinates with its distance, azimuth angle, and elevation angle. Distance is measured through to round-trip Time-of-Flight (ToF) of reflected radar signal, while azimuth and elevation angles can be either obtained by the beam steering angle of phased array antennas or be estimated with Direction-of-Arrival (DoA) estimation algorithms such as beam-forming or Multiple Signal Classification (MUSIC). Notice that as distance becomes further, the voxel size corresponds to the same angle increases, so that long-range objects contain much fewer voxels and appear to be more blurry. Besides, unlike the extremely narrow width of light beams, the cone-shape radar beam with sidelobes cause interference and leakage among nearby voxels and even the environment, which smears generated images. Last but not least, with a few centimeter resolution of distance measurement, a continuum of a large number of point reflector sums up and makes it an under-determined problem to localize them. Besides, radar reflection tends to be more specular than the mostly scattering reflection at optical frequencies. In other words, reflection off smooth objects might mainly towards an angle away from the radar receiver and disap-

pears in the image. Hence, "edge detection" in radar images is very different from that of pictures. Firstly, it needs to predict high frequency information with only low frequency data. Secondly, it needs to learn to fill up missing parts of the object.

### 2.2. Related Work

Previous works have attempted to adapt the optical camera-oriented Convolutional Neural Network (CNN) to its microwave counterpart to classify single-object images from high-resolution 2D synthetic aperture radar (SAR) images [3, 14, 5]. There has also been successful application of CNNs on recreating short range human body skeletons in 2D images and 3D spaces from radar images by tracking 14 key points [20, 21]. However, the scope of these application of neural networks on radar images are restricted to feature and single object classification. Besides, their radar image are either of super high resolution generated with airborne geographical sensing synthetic aperture radar or short-range where voxel resolution does not degrade too much. On the contrary, in our application of autonomous cars, we not only do not have as high resolution for long range, but required precisely traced boundary which contains the information of size, shape, and orientation of cars, bikes, and pedestrians. Therefore if we rely on fine grained feature detection that consists boundaries, we need very deep neural nets. Since CNN structures haven't be well investigated for radar signal and image, this task becomes extremely difficult. The emerge of GAN and cGAN provide a great opportunity to ...

### 2.3. Dataset Availability

Once we chose cGAN as the network model, we face the problem of dataset availability. Ideally, we should provide a large number of radar images of scenes that self-driving cars would typically experience, as well as precisely depicted object boundary and orientation in the images. However, we were not able to find any public dataset of 3D or even 2D radar images especially street views for the self-driving car application. Also in contrast to the standard images of cameras and LiDARs, different radar imaging system and image processing algorithms have non-negligible variation in output images. Hence we were initially forced to build our own dataset of 3D radar images with our custom-built 60GHz imaging system. However, since our system design compensates frame rate for cost and resolution, continuously collecting 500 frames will take at least 41.67 hours. We also have to wisely pick a good number of locations to mimic various scenes autonomous cars would actually face. Last but not least, for the ground truth dataset we have to obtain perfectly synced camera images or LiDAR point clouds with the radar. All of these overheads add up to a huge roadblock at the starting line of our project. Instead of building dataset from real radar images, we try to leverage well-developed EM simulation tools and the exquisitely measured waveform of our frequency modulated continuous wave (FMCW) radar to simulate radar reflections from objects and feed the simulated radar signal to synthesize radar images. We also try to compose radar reflection environment as similar to actual road environment as possible, so that we recover scenes from a well-known camera street view datasets called Cityscapes [2].

## 3. Method

Generative Adverserial Networks (or GANs) [6] have been widely used to generate images. The input to these networks can be text [12, 18], where images are generated according to some text or labels; or images [8, 15, 19, 17], where the network is trying to fill in some missing part of the input image, or translate the image onto another domain. In our case, we are looking to generate images with accurate boundaries using low resolution images with missing parts as input. Conditional GANs have already proven successful in similar settings, such as in [11, 19], where the authors have used thermal images under low light conditions where there some parts of the image are missing to retrieve the human face boundaries and estimate its orientation. Another motivation behind using conditional GANs is that loss functions such as the $L_2$ and $L_1$ (i.e. loss functions that are equal the Euclidean or $L_1$ distance between the input and the output) which are the de facto standard loss function for restoring images render blurry images, which are not suitable for our application as they do not emphasize on boundaries. On the other hand, using conditional GAN, we were be able to motivate the loss function in GAN to learn to focus on the boundaries, by designing the ground to contain information mostly about the boundary of objects, as discussed in more detail the dataset section.

### 3.1. Conditional GAN

We have adopted the CGAN architecture from [8] to train and test our model. Denoting the input, ground truth, and noise by $x$, $y$, and $z$ respectively, we can write the objective for a conditional GAN where the Generator and discriminator are $G$ and $D$ as

$$\begin{aligned}\min_G \max_D \mathcal{V}_{CGAN}(D,G) = \mathbb{E}_x(\log(D(x|y))) \\ + \ \mathbb{E}_z(\log(1 - D(x, G(x|z)))).\end{aligned} \tag{1}$$

Here, $D()$ is the score that the discriminator gives for a certain input, and $G(x, z)$ is what the generator tries to generate given the input $x$ and noise vector $z$. The point of having the noise vector, of course, it to avoid the problem of over-fitting. The difference between the standard GAN and the conditional version is that in the latter everything is conditioned to $y$. This $y$ could be anything we want, that

adds extra information about what the output $G$ should look like. In our setting, we call $y$ the ground truth, as it contains information of real boundaries of the objects. Given $y$, $D$ tries to maximize its output value when the input is real, and at the same time minimize it when the input is artificially generated by the $G$, the generator. At the same time, the generator tries to generate an image that gets a high score from $D$, motivating it to generate images that are similar to real ones.

It is suggested in [pix2pix] that we can combine generator's task of trying to get a high score from discriminator with a pre-determined loss function, such as $L_1$. That is, one could change the objective to be

$$\begin{aligned} &\min_G \max_D \mathcal{V}_{CGAN}(D,G) = \\ &\mathbb{E}_x(\log(D(x|y))) \quad + \quad \mathbb{E}_z(\log(1 - D(x, G(x|z)))) \\ &+ \lambda \mathcal{L}_{L_1}(G(x,z)). \end{aligned} \tag{2}$$

The motivation behind this is to capture the low-frequency information using the $L_1$ loss, and motivate GAN to model high-frequency information, which in our case, will translate to more precision in identifying boundaries.

For the generator, was have adopted the architecture of [13], which is an auto-encoder. Similar to most such architectures, U-net consists of a contracting path and an expansive path. The first layer contracting path is made of two successive $3 \times 3$ convolutions followed by a ReLu (rectified linear unit) and a $2 \times 2$ max-pooling layer without overlapping, which shrinks the size of its input by a factor of four. Let us call such a layer a down-sampling layer. This layer is then repeated multiple time, each time doubling the feature channels. The expansive path reverses this process: at each layer, the data is first up-sampled by a factor of two, the filtered with using a $2 \times 2$ window (2 by 2 convolution) halving the number of feature channels. At this point, a cropped version feature maps from the corresponding layer in the contracting path is forwarded to this layer (i.e. for layer $k$, the feature maps from layer $n - k$ are forwarded, assuming a total number of $n$ layers) and are concatenated with the current feature maps. There are two points to be mentioned here; first, the cropping should happen because of the loss of edges that has happened due to successive convolutions. Second, and more importantly, this idea is used as a way to detour the bottleneck layer (the middle layer through which all the information should pass) by creating these skip connections between layers. One reason behind this choice of architecture is that this forwarding encourages the similar structure between the input and the output. Authors in [13] evaluated this architecture in a biomedical image segmentation problem its idea has, and in [8], it was adopted to generate a general translator of high-resolution output images to high resolution input images, which is quite similar to our setting, except that in our problem the input images are not high-resolution.

As for the discriminator, what we need is for the network to be able to identify local structures, in order to capture the properties of local objects (e.g. cars). Since the network is relying on the $L_1$ loss to guarantee the correctness of low-frequency information, it is possible to restrict the discriminator to only penalize structures that occur within a patch window of the image. In other words, the discriminator slides over the image looking at patches of size $n \times n$, and scoring each of them, and finally averaging over all patches to derive the final score. This structure has been dubbed patchGAN [8]. For our implementation, we chose $n$ to be 70 where the image size was $256 \times 256$.

### 3.2. Dataset Generation

As analyzed in section 2, when exploiting the application of deep learning in the new field of radar images, the problem of dataset availability is inevitable. Between the two options of building our own dataset: collecting real radar images through experiments and processing synthesized radar reflection, we have to make a trade-off. Although synthesizing dataset might be the only feasible way to create a big enough training dataset, the compatibility of trained model to real radar images significantly depends on the closeness between synthesized and real radar reflection. Luckily, the transmission, propagation, and reflection of radar waveform are thoroughly studied and can be precisely modeled and simulated. Furthermore, with the help of advanced electromagnetic simulation softwares such as FEKO we can even model the attenuation and specularity of various objects and surfaces. Therefore, a well-designed radar reflection synthesizing algorithm should be indistinguishable from read radar signal, so it is hopeful that the performance of a cGAN model trained with synthesized radar images should not degrade too much when we use real radar image for testing instead. We are also planning to mixing a smaller number of real radar images with the synthesized dataset for training and find the improvement of cGAN performance with respect to the portion of real and synthesized data.

The radar image synthesizing process can be further separated into four steps: scene generation, radar reflector modeling, radar signal simulation, and image processing. The last two steps are standard and straightforward, while the challenge lies in creating the 3D space distribution of point reflectors based on a realistic scene of automotives. We found two types of dataset that can be of great use for us. The first type of dataset is 2D video recording of street scenes from the view of a car, such as the Cityscapes dataset [2], while the second type of dataset is 3D CAD models of typical objects on the road. Good examples are the Deformable 3D Cuboid Model dataset and the 3D ShapeNets dataset [4, 16]. These two types of datasets contains unique information and make great complement to the other.

Video recordings from the Cityscapes includes a wide range of practical urban road environment from the point of view of a car, but the 3D shape of objects cannot be recovered at all. Besides, we could only very roughly estimate the location of most common objects like vehicles and human, so that from the 2D pictures of street view, we can at most place 2D shapes at roughly estimated angle and distance. On the other hand, the 3D CAD models provide us with precise 3D distribution of point reflectors that consists typical objects on the road such as cars, trucks, cyclists, and pedestrians. We can also find out the effective reflectors in the sight from an arbitrary angle, and even the specularity of reflection off surfaces. However, it is not trivial to place models and compose a verisimilar 3D scene. In our experiments, we first tried to create scenes with 2D and 3D datasets separately. For the 2D dataset, we use vision-based object detection neural networks Mask R-CNN [7] to detect masks of objects and label them, and then roughly estimate and place 2D shapes of objects based on their size and pixel-wise position to recover scenes in 3D space. After that, we assign a number of point reflectors to every objects according to their shape and labels, and from there radar images can be synthesized. In this way we can potentially build a dataset with the same size of the Cityscapes, although the EM simulation takes time and the synthesized 3D radar images are of 2D shapes. However, we want to argue that if we project the 3D synthesized image back to onto a screen, the effect of distance estimation of objects on the output image can be ignored. We are also planning to combine the 2D and 3D datasets to compensate the shortcomings of using 2D video recordings alone, and synthesize a realistic 3D street scene dataset. We will start with finding the 3D model of the object type classified by the Mask R-CNN, together with a point of view, and even distance that best fits the object mask in the video recording. According to this more precise location and orientation estimation, we can place the 3D models at almost the same place as in the video. Followed by the effective point reflector assignment with the specularity into consideration, and radar signal and imaging simulation. We can synthesize very authentic 3D radar images only at the cost of computation time.

# 4. Experimental Results

For our experiments, we ran our data one NVIDIA Tesla K80 GPU available on Google Cloud using CUDA 9.0 and PyTorch 0.4, along with a local Pascal GTX 1080 GPU using CUDA 10.0 and PyTorch 1.0. We used 1056 images of size $256 \times 256$ as for training, and 452 images for testing. The training phase takes about 11 hours to finish if the complete training dataset is used.

The first experiment we ran was using blurred images. We took the images of cars, added some Gaussian noise and convolved the image with a two-dimensional $sinc$ function. We fed these images as input, and the original images to the network as ground truth. the results showed that the network is able to restore the boundaries of the cars with a high accuracy. The problem with this implementation was that the image was not gray-scaled, and we did not go through a precise simulation according the to underlying physics of mmWave radars to generate the images. The purpose of this step was to evaluate the feasibility of our idea of restoring the boundaries using GANs on a high level. The results from this experiments are shown in [figure]. It can be seen that the boundary of the car has been fairly accurately reconstructed. It is also noteworthy that the GAN was able to restore the side mirror of the car successfully, although there is no visible point in GAN.

In the second experiment, as mentioned in the dataset section, we synthesized a more realistic dataset that are similar to radar images. In most cases, the GAN was able to restore the boundaries

## 4.1.

## 4.2. Dataset

Training dataset 118 Testing dataset 509

## 4.3. Results

# 5. Discussion and Conclusion

Analyze the results, summarize the findings and point out possible future directions

# References


[1] Automated vehicles for safety. 1

[2] Cityscapes dataset. 2, 3

[3] S. Chen and H. Wang. Sar target recognition based on deep learning. *International Conference on Data Science and Advanced Analytics, DSAA*, 2014. 2

[4] S. Fidler, S. Dickinson, and R. Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. *Conference on Neural Information Processing Systems, NIPS*, 2012. 3

[5] M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao. Change detection in synthetic aperture radar images based on deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems , DSAA*, 2016. 2

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. *arXiv:1703.06870 [cs.CV]*, 2017. 4

[8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017. 2, 3

[9] W. Jones. Keeping cars from crashing. *IEEE Spectrum*, 2001. 1

[10] B. Mamandipoor, G. Malysa, A. Arbabian, U. Madhow, and K. Noujeim. 60 ghz synthetic aperture radar for short-range imaging: Theory and experiments. *Asilomar Conference on Signals, Systems and Computers*, 2014. 1

[11] A. Nambi, S. Bannur, I. Mehta, H. Kalra, A. Virmani, V. Padmandabhan, R. Bhandari, and B. Raman. Demo: Hams: Driver and driving monitoring using a smartphone. *ACM MobiCom*, 2018. 2

[12] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016. 2

[13] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 3

[14] C. Schwegmann, W. Kleynhans, B. Salmon, L. Mdakane, and R. Meyer. Very deep learning for ship discrimination in synthetic aperture radar imagery. *IEEE International Geoscience and Remote Sensing Symposium, IGARSS*, 2016. 2

[15] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. *arXiv preprint arXiv:1711.11585*, 2017. 2

[16] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shape modeling. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2015. 3

[17] H. Zhang, V. Sindagi, and V. M. Patel. Image de-raining using a conditional generative adversarial network. *arXiv preprint arXiv:1701.05957*, 2017. 2

[18] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint*, 2017. 2

[19] T. Zhang, A. Wiliem, S. Yang, and B. Lovell. Tv-gan: Generative adversarial network based thermal to visible face recognition. In *2018 International Conference on Biometrics (ICB)*, pages 174–181. IEEE, 2018. 2

[20] M. Zhao, T. Li, M. Alsheikh, Y. Tian, H. Zhao, A. Torralba, and D. Katabi. Trhough-wall human pose estimation using radio signals. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2018. 2

[21] M. Zhao, Y. Tian, H. Zhao, M. Alsheikh, T. Li, R. Hristov, Z. Kabelac, D. Katabi, and A. Torralba. Rf-based 3d skeletons. *SIGCOMM*, 2018. 2