IN-VEHICLE CAMERA IMAGES PREDICTION BY GENERATIVE ADVERSARIAL NETWORK

Junta Watanabe and Tad Gonsalves

Department of Information & Communication Sciences,
Faculty of Science & Technology
Sophia University, Tokyo, Japan

ABSTRACT

Moving object detection is one of the fundamental technologies necessary to realize autonomous driving. In this study, we propose the prediction of an in-vehicle camera image by Generative Adversarial Network (GAN). From the past images input to the system, it predicts the future images at the output. By predicting the motion of a moving object, it can predict the destination of the moving object. The proposed model can predict the motion of moving objects such as cars, bicycles, and pedestrians.

KEYWORDS

Deep Learning, Image Processing, Convolutional Neural Network, GAN, DGAN

1. INTRODUCTION

Moving object detection is one of the fundamental technologies necessary to realize autonomous driving. Object detection has received much attention in recent years and many methods [1- 5] have been proposed. However, most of them are not enough to realize moving object detection, chiefly because these technologies perform object detection from one single image. Therefore, they can get the position of the desired object, but they cannot discriminate whether the object is stationary or non-stationary. For the same reason, they cannot predict the future positions of the object. This kind of technology is therefore not apt for autonomous driving. Normally, there are many pedestrians on a sidewalk. Autonomous cars driving on a roadway must discriminate whether the pedestrians will continue to be on the sidewalk or get into the roadway at a given point in time in the future.

Multiple ways to solve this problem have been considered - predicting future positions of the objects from their past positions, and predicting future camera images from past camera images, for instance. However, the former has a disadvantage. Processing time is directly proportional to the number of objects in a camera image. If there are many objects in a camera image, processing time may be too long to realize autonomous driving. Conversely, the latter is not affected by the number of objects in the image. It does not count the objects in a camera image when it predicts the future. It predicts the future positions of the objects at a time as a future image. Then, precise object detection methods detect the future positions of the object from the image of prediction. Hence, it is assumed that the latter outperforms the former.

It is not enough to predict a future image by using just a single image. This is because we cannot guess the speed of objects in an image. For example, when there are two cars in an image, we cannot guess if they are stationary or not. We guess the speed of the cars from their past and current positions. Therefore, the future image prediction needs some past images.

Dhinaharan Nagamalai et al. (Eds) : COSIT, AIAPP, DMA, SEC - 2019
pp. 45–55, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.90205

Multiple ways to predict future images from past images are considered, too. This paper suggests a model which predicts future images with Generative Adversarial Networks (GANs) [6]. GANs consist of two models: a generative model and a discriminative model. GANs learn a loss that discriminate whether the output image is real or fake, while training a generative model to minimize a loss. The image generated by Deep Convolutional GANs (DCGANs) [7] is often clear. It is because blurry images which look fake will not be tolerated by a discriminative model. To get clear prediction image, the method proposed in this paper uses GANs.

2. Related Work

Generative Adversarial Networks Generative adversarial networks are generative models. The purpose of GANs is to generate a fake data which is indistinguishable from a real one. GANs consists of two models: a generator G and a discriminator D. G generates a fake data G (z) from a noise z. D discriminates whether an input data x is real or fake. They are adversarial. Therefore, G will generate a fake data which is indistinguishable from a real one.

Pix2pix: This model can solve the image-to-image translation. A generator of pix2pix [8] uses a "U-Net" based architecture. The U-net [9] is a U-shaped convolutional neural network. The shape allows the U-net to make an output image which has features extracted in shallow layers and deep layers.

MoCoGAN: Motion and Content decomposed Generative Adversarial Network (MoCoGAN) [10] is a network which generates a video. MoCoGAN uses two discriminators: an image discriminator and a video discriminator. The image discriminator discriminates whether an input image is real or fake. The video discriminator discriminates whether an input video is real or fake.

ResNet: Residual Network (ResNet) [11] makes it easy to train deep neural networks. In ResNet, layers are reformulated as learning residual functions with reference to the layer inputs. When training deep networks, one often runs into the degradation problem which is prevented by the use of ResNet.

The approaches of moving object detection by cameras are [12], [13], [14]. They indicate how to detect moving objects from camera images. Our approach is how to predict future camera images. Therefore, we can predict positions of future moving objects by combining their methods and our method.

Our proposed method only uses an in-vehicle camera. Methods which use multiple sensor are [15], [16], [17]. Sensors enable measurement of the distance between a vehicle and a moving object. However, it is difficult to install sensors. Our method only uses a camera and hence the implementation is easier and cost-effective.

3. Methods

Our generator predicts 4 future images from 3 past images. All these images are different from one another. For example, 1 second later, 2 seconds later, 3 seconds later and 4 seconds later. It seems the generator should generate one future image, not 4 future images. This is because if the generator repeats generating 1 future image, we will get more future images. The reason for predicting 4 future images is that training the generator that predicts 4 future images is easier than training the generator that predicts 1 future image. In the early stages, the generators generate blurry images. If we use the generator that predicts 1 future image, the generator predicts an image which is 2 seconds later from a blurry image which is 1 second later. The generator predicts an image which is 3 seconds later from a more blurred image which is 2 seconds later.

The image which is 3 seconds later is too different from the ground truth. In training the generator, we calculate a loss between the image and the ground truth. However, the image is predicted by the generator from the blurry image. Calculating the loss between the image and the ground truth cannot be said to be the right thing. Therefore, the generator calculates not 1 picture but multiple pictures. The reason for predicting 4 pictures has to do with the capacity of the Graphic Processing Units (GPUs). If we make the generator predict 4 pictures, the datasets will exceed the capacity of my GPUs. Nevertheless, if we make the generator to predict a few pictures, a discriminator of videos will not work. It is because the task of the discriminator is to discriminate whether an input video is real or fake. The shorter the videos are, the closer it is to discriminate still images from video images. Therefore, if we want the generator to generate superior videos, we must make the video long. The reason for predicting images from 3 images is the capacity of Graphic Processing Units (GPUs), too.

3.1 Network Architectures

Our network consists of 3 sub-networks: a generator G, an image discriminator Di and a video discriminator Dv (Figure 1). The generator G is a generative model that learns a mapping from an input video clip x to an output video clip V. We define a predicted video clip as PV. The function of generator G can be expressed as:

$$PV = G(x) = \{PI_{t+3}, PI_{t+4}, PI_{t+5}, PI_{t+6}\} \quad (1)$$

$$x = \{I_t, I_{t+1}, I_{t+2}\} \quad (2)$$

$$V = \{I_{t+3}, I_{t+4}, I_{t+5}, I_{t+6}\} \quad (3)$$

PI means predicted image. Di and Dv come from MoCoGAN.

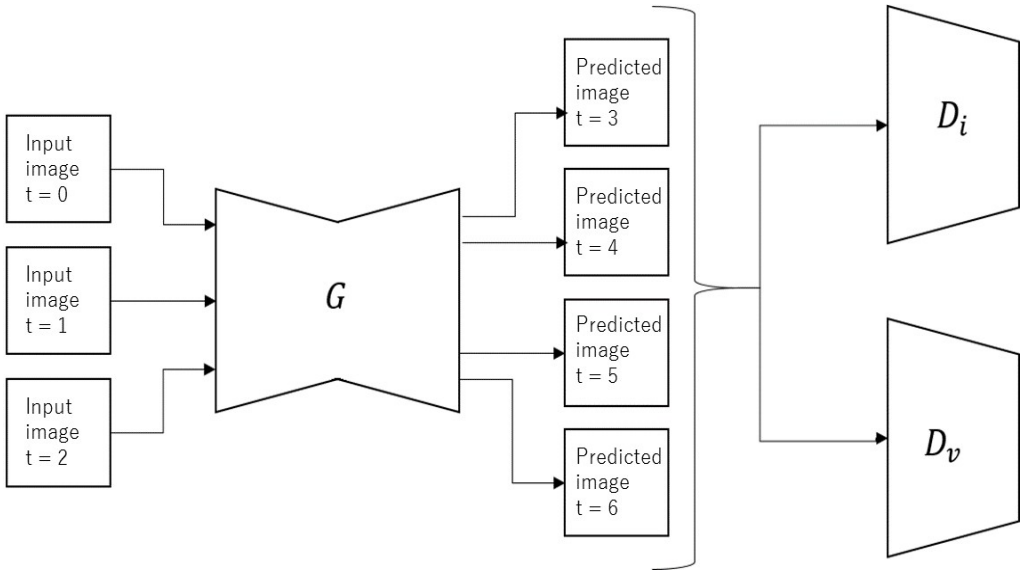

Figure 1. Our proposed method

3.1.1 Generator

The generator consists of 4 parts: pre-processor, encoder, concatenator and decoder.
Preprocesser: Preprocessor has a convolutional layer (3 × 3). The size of the input image is 128 x 128 x 3. The layer resizes it to 128 x 128 x 16. After the convolution, we do Batch Normalization (BN) [18] to speed up the training process.

Encoder: The input is fed to the ResBlock of the Encoder and Average Pooling is repeated. ResBlock consists of 4 residual units (ResUnit). Each of the ResUnit consists of 6 layers and shortcut connections. This structure is found in [19]. The Encoder successively outputs the images after passing through the ResBlock. In all, here are 3 Encoders, corresponding to the t = 0, 2, 3 output.

Concatenator: Concatenator connects the same-sized outputs from the Encoder and passes them through the Res-Block. The number of the channels in the ResBlock output is 4/3 times that of the input. This is because there are 4 Decoders corresponding to 3 Encoders.

Decoder: There are 4 Decoders, handling the t = 3, 4, 5, 6 outputs. The output array from the Concatenator is split into 4 parts, each of which is then fed to the Decoder. The 8 × 8 × 256 convolutions performed at the Encoder are then subjected to deconvolution [20] inside the Decoder.

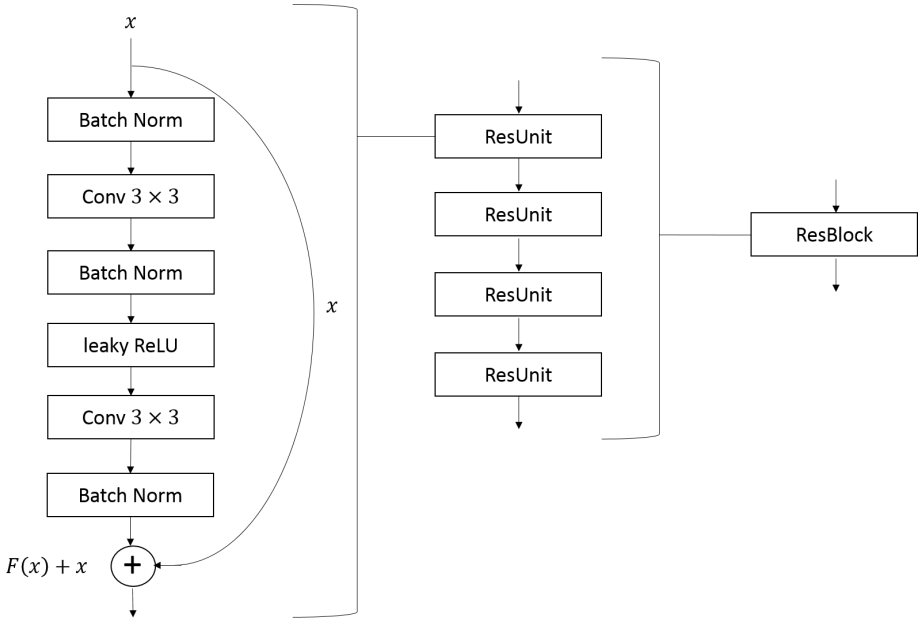

Figure 2. ResBlock of PyramidNet

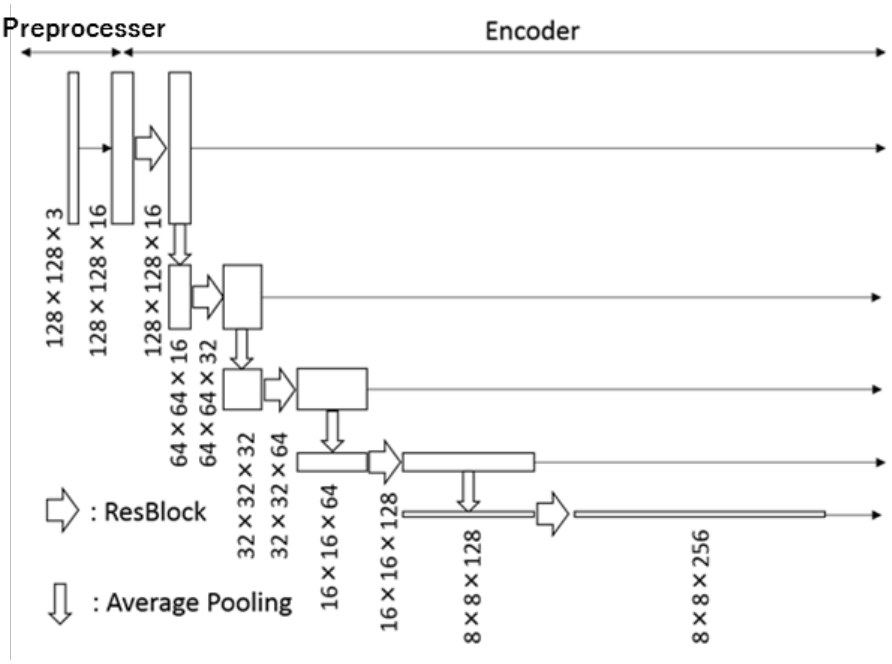

Figure 3. Architecture of our pre-processor and encoder

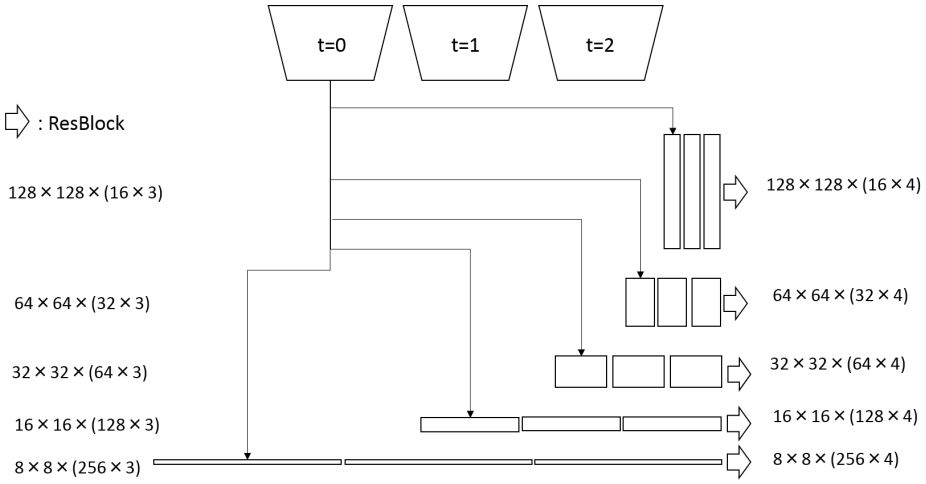

Figure 4. Architecture of our concatenator

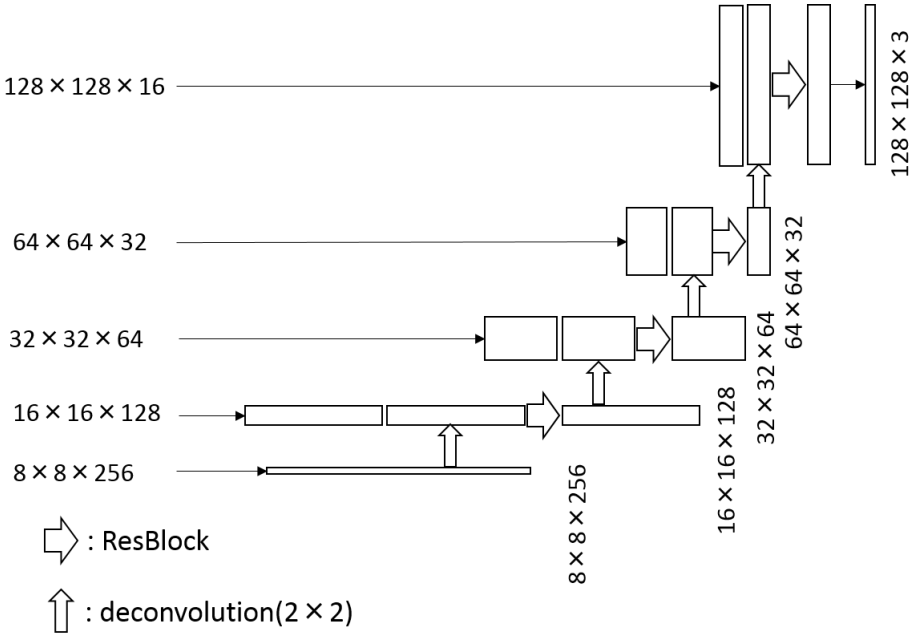

Figure 5. Architecture of our decoder

3.1.2 Discriminators

Image Discriminator: Discriminator plays the role of identifying the "ground truth" of the input images. The images produced by the generator at time steps t=3,4,5,6 and the real images are fed into the system.

Video Discriminator: This discriminator plays the role of identifying the "ground truth" of the input videos. The expected video at time steps t=3,4,5,6 and the real videos are combined and fed into the system. The Video Discriminator performs 3D convolution computations. Since four 128 × 128 × 3 images are input, the input size is 128 × 128 × 4 × 3. However, for the convenience of 3D convolution, it is set to 128 × 128 × 128 × 3 by 0 padding.

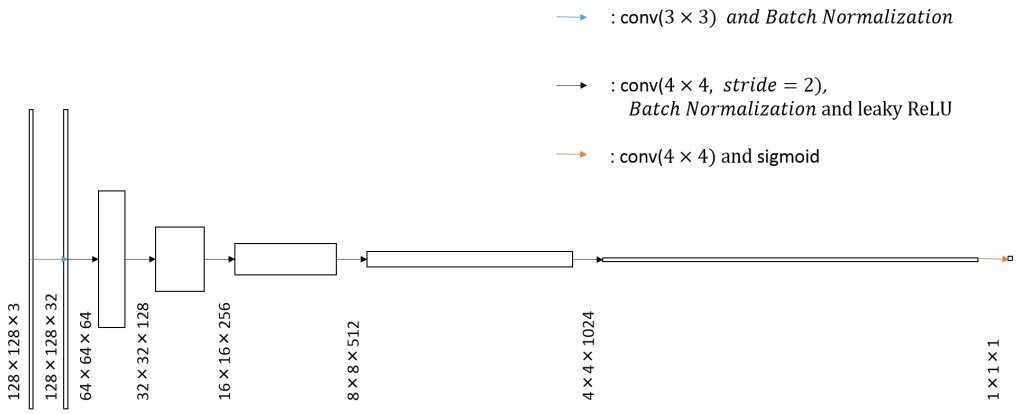

Figure 6. Architecture of Image Discriminator

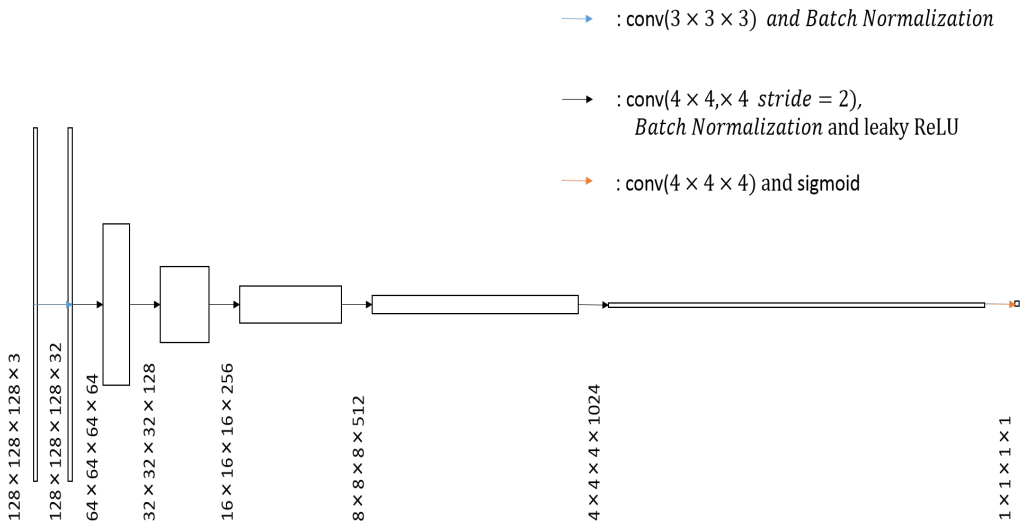

Figure 7. Architecture of Video Discriminator

3.2 Parameters Updating

The loss function of Generator is evaluated as follows:

$$L_G(G, D_i, D_v) = \lambda_1 L_{L1}(G) - \lambda_2 L_{adv}(G, D_i, D_v) \tag{4}$$

$$L_{L1}(G) = \mathbb{E}_{I,PI\sim\mathbf{I},\mathbf{PI}}[\|I - PI\|_1] \tag{5}$$

$$L_{adv}(G, D_i, D_v) = \mathbb{E}_{(PI\sim\mathbf{PI})}[\log D_i(PI)] + \mathbb{E}_{(PV\sim\mathbf{PV})}[\log D_v(PV)] \tag{6}$$

With the L1 distance, we train Generator to bring the distance between *I* and *PI* closer. The second term of L_G encourages D_i and D_v to output 0 for a video frame from a generated image *PI* and a generated video *PV*.

We define the following loss function of Image Discriminator:

$$L_{D_i}(G, D_i) = \mathbb{E}_{(PI\sim\mathbf{PI})}[\log D_i(PI)] + \mathbb{E}_{(I\sim\mathbf{I})}[\log(1 - D_i(I))] \tag{7}$$

The first term encourages D_i to output 1 for a real image *I* and 0 for a generated image *PI*.
The loss function of Video Discriminator can be written as:

$$L_{D_v}(G, D_v) = \mathbb{E}_{(PV\sim\mathbf{PV})}[\log D_v(PV)] + \mathbb{E}_{(V\sim\mathbf{V})}[\log(1 - D_v(V))] \tag{8}$$

Similarly, the first term encourages D_v to output 1 for a real video *V* and 0 for a generated video *PV*.

We set $(\lambda_1, \lambda_2) = (100, 1)$. We use the He parameter initializer and the Adam optimizer for training. We also use the weight decay strategy. The rate of Generator and Discriminator updating is 3:1. This is to prevent the premature development of the Discriminator.

4. EXPERIMENTAL RESULTS

4.1 Dataset

In our experiment, we use LISA Traffic Light Database. The database is made for the detection of traffic light from an in-vehicle camera image. It consists of in-vehicle camera images, the coordinates of traffic lights, and information of the signal color. In our experiment, we only use in-vehicle camera images.

4.2 Result

In this experiment, the network was trained for 10 epochs (140,000 iterations). The learning result is as shown in Figure 8 and Figure 9. From the top, real images at t = 0, 1, 2 and the predicted images at t = 3, 4, 5, 6 are arranged. From Figure 8, it can be seen that the motion of the moving object can be predicted. We can also predict the movement of cars and road signs that are about to go out of the screen. Thus, we may conclude that the system can predict the motion of cars, bicycles, pedestrians, etc.

Figure 8. Prediction of moving objects (1)

Figure (8). Prediction of moving objects (2)

However, the shape of the objects such as automobile and bicycle parts appear blurred (Figure 9). Three possibilities are conceivable as the cause of the blurred output images. Firstly, there are few images of left or right turn in the training data. Of the 14,021 training data, only about 550 images deal with turning. When the training data are small, the generalization performance of the model is lowered, and the quality of prediction for the test data is lowered. This can be solved by increasing the amount of training data.

The second is the possibility that the function of the Discriminator was insufficient. Increasing the updating frequency and the ratio of output of the Discriminator in error function, and changing the structure of the Discriminator, etc. can be considered.

The third is the possibility that we expected too much work from the Generator model. In MoCoGAN, noise was generated for each moving object in the motion picture and its movement, and it was convoluted with it. MCnet handled moving objects and movements separately. In our model, we did not separate such images, and we had images generated at the same time. However, our system can predict the motion of the object, although a certain amount of blur remains regarding the shape of the object. Improvement measures can be considered by changing the configuration of the model to make it possible to recognize both movement and form.

5. CONCLUSION

As a method of predicting future camera images, we proposed a method using DCGAN. Images with t = 0, 1, 2 were input, and images with t = 3, 4, 5, 6 were predicted. With this model we were able to predict the image of the future, but although the movement of the moving object could be predicted, the shape was blurred, and it resulted in lack of accuracy. In the future it is necessary to try to build a model that can solve this problem and increase the amount of training data.

REFERENCES

[1] S.Ren, K.He, R.Girshick, & J.Sun, (2015) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, NIPS.

[2] J.Redmon, S.Divvala, R.Girshick, & A.Farhadi, (2016) You Only Look Once: Unified, Real-Time Object Detection, CVPR.

[3] J.Redmon, & Ali Farhadi, (2018) YOLOv3: An Incremental Improvement, arXiv preprint arXiv: 1804.02767v1.

[4] W.Liu, D.Anguelov, D.Erhan, C.Szegedy, & S.Reed, (2016) SSD: Single Shot MultiBox Detector, ECCV.

[5] T.Lin, P.Dollar, R.Girshick, K.He, B.Hariharan, & S.Belongie, (2017) Feature Pyramid Networks for Object Detection, CVPR.

[6] I. J.Goodfellow, J.Pouget-Abadie, M.Mirza, B.Xu, D.Warde-Farley, S.Ozair, A.Courville, & Y.Bengio, (2014) Generative Adversarial Networks, NIPS.

[7] A. Radford, L. Metz & S. Soumith, (2015) Un-supervised Representation Learning With Deep Convolutional Generating Adversarial Networks, CoRR.

[8] P.Isola, J.Zhu, T,Zhou, & A.A.Efros, (2017) Image-to-Image Translation with Conditional Adversarial Networks, CVPR.

[9] O. Ronneberger, P. Fischer, & T.Brox. (2016) U-Net: Convolutional Networks for Biomedical Image Segmentation, ICLR.

[10] S.Tulyakov, M.Liu, X.Yang, & J.Kautz, (2017) MoCoGAN: Decomposing Motion and Content for Video Generation, arXiv preprint arXiv: 1707.04993.

[11] K. He, X. Zhang, S. Ren, & J. Sun, (2016) Deep Residual Learning for Image Recognition, CVPR.

[12] WC. Hu, CH. Chen, TY. Chen, DY. Huang, & ZC. Wu, (2015) Moving object detection and tracking from video captured by moving camera, J. Vis. Commun. Image Represent, vol. 30, pp. 164-180.

[13] A. Rozantsev, V. Lepetit, & P. Fua, (2014) Flying Objects Detection from a Single Moving Camera, CVPR.

[14] S. Chen, T. Xu, D. Li, J. Zhang, & S. Jaing, (2016) Moving Object Detection Using Scanning Camera on a High-Precision Intelligent Holder, Sensors, vol. 16, no. 10, p. 1758.

[15] R. O. Chavez-Garcia & O. Aycard, (2016) Multiple sensor fusion and classification for moving object detection and tracking, IEEE Trans. Intell. Transp. Syst., vol. 17, no. 2, pp. 525-534.

[16] F. Fayad & V. Cherfaoui, (2008) Detection and Recognition confidences update in a multi-sensor pedestrian tracking system, Information Processing and Management of Uncertainly in Knowledge Based Systems, pp. 409-416.

[17] R. Labayrade, D. Gruyer, C. Royere, M. Perrollaz, & D. Aubert, (2007) Obstacle Detection Based on Fusion Between Stereovision and 2D Laser Scanner, Mobile Robots: Perception & Navigation

[18] S.Ioffe, & C.Szegedy, (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, ICML.

[19] D.Han, J.Kim, & J.Kim, (2017) Deep Pyramidal Residual Networks, CVPR.

[20] V.Dumoulin, & F.Visin, (2016) A guide to convolution arithmetic for deep learning, arXiv preprint arXiv:1603.07285v2.