Skoltech
Skolkovo Institute of Science and Technology

Skolkovo Institute of Science and Technology
Computational Data-Intensive Science and Engineering center
Deep Learning course

Depth Map Real-Time Estimation from a Stereo Pair of Images

Aleksandr Safin, Grecia Diaz, Iaroslav Koshelev, Iurii Minin

June 5, 2018

Table of contents

1 Abstract

The present paper demonstrates concept disparity expenditures for convolutional neural networks. We facilitate three synthetic stereo image datasets with sufficient realism, variation, and size to successfully train large networks. Training and evaluation of some methods. We implemented convolutional neural network for real-time disparity estimation that provides good results.

In this paper we are learning a mapping from image pixels into a dense template grid through a fully convolutional network. By using manually annotated landmarks, we solve this problem as a regression problem and train our network. The landmarks are revealing a depth correspondence field for 3D objects. They then serves as the ground-truth for training our regression system. We implemented semantic segmentation with regression networks. Our system is estimating dense image-to-template correspondences in a fully convolutional manner. We use correspondence information to feed automatic system and then to obtain landmark localization. The code it is available at https://github.com/keqpan/DispNet

2 Introduction

Several recent works in deep learning have aimed at enriching deep networks with information about shape by explicitly modelling the effect of similarity transformations [1] or non-rigid deformations [2, 3, 4]; several of these have found success in classification [1], fine-grained recognition [2], and also face detection [4]. There are works [5, 6] that model the deformation via optimization procedures.

Recent works on 3D surface correspondence [7, 8] have shown the merit of CNN-based unary terms for correspondence. There are works that address the problem of establishing dense correspondence for the human body from static RGBD images [9, 10, 11, 12].

We proceeded the depth map of all visible points in a stereo images.It comes to reconstruction and provides an important basis for numerous higher-level challenges such as advanced driver assistance and autonomous systems. The joint estimation of all components would be advantageous with the respect to efficiency and accuracy. We implemented these results suggested that disparities can be estimated via a convolutional network, ideally jointly, efficiently, and in real-time [14]. We use [14] dataset that includes stereo color

images and ground truth for bidirectional disparity. Moreover, the full camera calibration and 3D point positions are available (RGBD data). We demonstrate various usage examples in conjunction with convolutional neural network training. We train a network for disparity estimation, which yields competitive performance also on previous benchmarks, especially among those methods that run in real-time.

We are recovering informative signal variations by training convolutional network discriminatively. Implemented system combines the merits of the mentioned methods. Learning-based approaches typically pursue invariance to shape deformations. We implement the feedforward manner and a single shot to model the deformation. We tackle the more challenging task of establishing a 2D to 3D correspondence in the wild by leveraging upon recent advances in semantic segmentation [18]. We implement a discriminatively trained network to obtain, in a fully-convolutional manner, dense correspondences between an input image and a deformation-free template coordinate system by connecting [19, 20, 21].

3 Data Set

We trained our system using three different data set:

- **Driving scene from [14]** : This was our principal dataset. It is a synthetic data set that contains over 35 000 stereo image pairs with ground truth disparity. It consist of dynamic street scene from the viewpoint of a driving car. The stereo baseline is set to 1 Blender unit, which together with a typical car model width of roughly 2 units is comparable to KITTI's setting (54cm baseline, 186cm car width).

- **Virtual KITTI [26] and KITTI [25]** : Virtual KITTI contains 21,260 frames generated from five different virtual worlds in urban settings under different imaging and weather conditions. These worlds were created using the Unity game engine.

  The KITTI datasets contains stereo videos of road scenes from a calibrated pair of cameras mounted on a car, this car was driving around the city of Karlsruhe.Ground truth for optical flow and disparity is obtained from a 3D laser scanner combined with the egomotion data of the car. While the dataset contains real data, the acquisition method restricts the ground truth to static parts of the scene. Moreover, the laser only provides sparse data up to a certain distance and height.

- **Air Sim [17]** : Additional to the previous dataset we used the AirSim simulator for cars to generated images. As equal as Virtual KITTI, it is built on Unreal Engine.We used the "Computer Vision" mode and implemented and algorithm to adapt the cameras and modify the setting to get the stereo images and depth map with the characteristics that we need.

Fig. 1: Example scenes from top to bottom; Driving scene, Virtual KITTI, KITTI, Air Sim

- **Data science bowl 2018 [33]** : This dataset contains a large number of segmented nuclei images (Fig. 2). The images were acquired under a variety of conditions and vary in the cell type, magnification, and imaging modality(mixture of bright-field and fluorescence). There are 670 training images and multiple masks for each training images file. The total images of mask from training dataset is: 29461. From

the test folder, we have 65 images. The images in the dataset in the different size, the range shape come from $(256 \div 1388) \times (256 \div 1040)$ (Fig. 2). On average the red, green and blue (Fig. 2) channels have similar intensities for all images. It should be noted that the background can be dark (black) as as well as light (white).

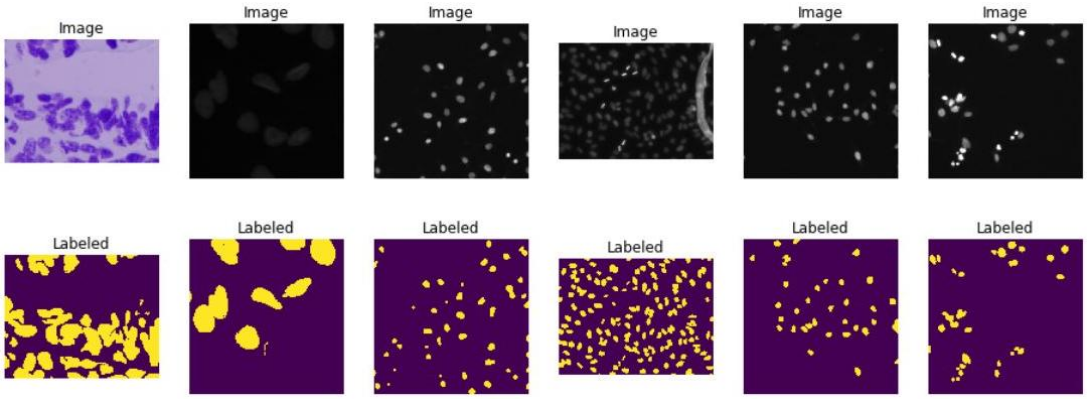


Fig. 2: Images labeling

4 Model estimators

We used ReLU activation function for DispNet while DispNet loss function was considered to be weighted sum of End-Point-Error (EPE) for all predictions. While implementing Quantized Regression, for the classification task we used Softmax and Cross Entropy loss, for the regression we used smooth L1 loss. We used in postprocessing the following estimating functions for CNN: MSE (mean square error) and IoU (Intersection over Union). IoU principle: $IoU(A,B) = \frac{A \cap B}{A \cup B}$. Furthermore, for U-Net NN, according the reasons of [28] in this work it is used the *Dice Coefficient* loss function. The coefficient compares the pixel-wise agreement between the ground truth (Y) and its corresponding predicted segmentation ($\tilde{Y}$): $dice\ coeff = \frac{2 \cdot |Y \cap \tilde{Y}|}{|Y| + |\tilde{Y}|} = -dice\ loss\ function$. There are some modification that some dropout layers had been added and ELU had been used instead of RELU.

5 Implemented methods

5.1 DispNet (Aleksandr Safin)

Since DispNet[14] was claimed as fast and accurate enough method, we have decided to apply it in our project. DispNet has been implemented in PyTorch and trained on the Driving dataset[14] from the scratch. This network resembles convolutional neural networks for segmentation task by the general structure, namely it consists of two parts: contracting and expanding. From the other hand, since the task of disparity estimation could be considered as regression task, therefore opposite to segmentation task, DispNet has regression layer and we use ReLU activations. DispNet employs "coarse-to-fine" approach (see Fig. 3): on every step of upsampling in expanding part, we predict a disparity map corresponding to the lower resolution and use it while predicting on the next step.

The tricky part is that it turned out that using BatchNorm is crucial to train such a network, however the authors have not stated this in the original paper.

Since network predicts disparity maps of lower resolutions on every level of upsampling part, one could consider all those intermediate predictions in the loss function. So, as loss function here we would consider weighted sum of End-Point-Error (EPE) for all predictions. Authors have proposed to implement weight scheduling, meaning that we start with all weight is putted on loss corresponding to the image of the lowest resolution. Then during the training, the weight corresponding to the largest image increases gradually, while the other weights will be zeroed. Therefore, learning curves look like on the Fig. 4.

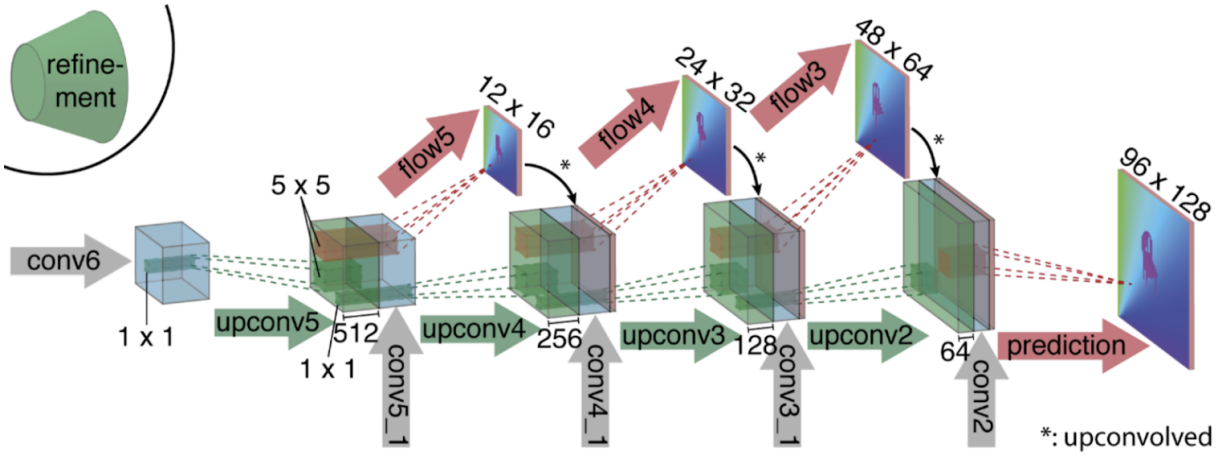


Fig. 3: Expanding part of DispNet

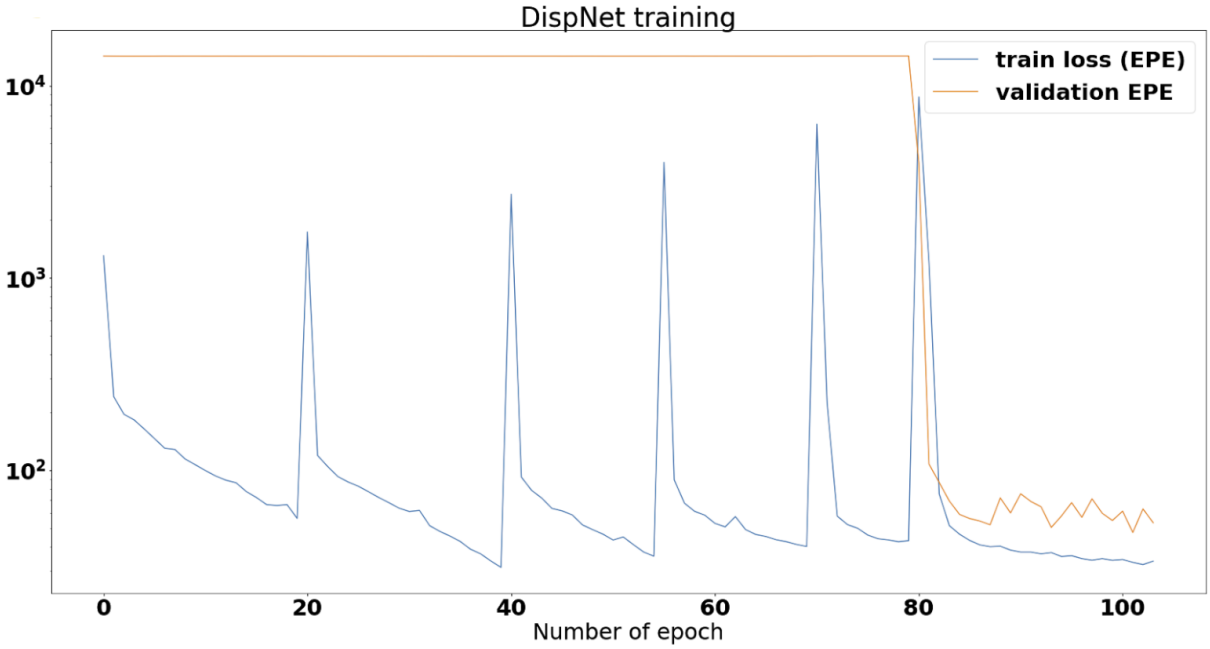


Fig. 4: DispNet training curves

5.2 Quantized Regression (Grecia Diaz)

In order to improve the results we decided to implement DenseReg [12], and adapt it to combines DispNet with regression network. The approach is to quantize and estimate the quantization error separately for each quantized value. Instead of directly regressing each pixel, the quantized regression lets us solve a set of easier sub-problems, combines a classification with a regression task.

First we estimated a grossly quantized function through a classification branch to identify a region that can contain each pixel. For each segment we use a separate residual regression unit's prediction, and multiplexing the different residual predictions. These are added to the quantized prediction, yielding a smooth and accurate correspondence field. During training the model we upscale classification and regression branches before compute the losses, we used the weights of 1000 for classification and 1 for regression in order to balance their contribution, we used 10 segments and each step of 30.

5.3 SqueezeSeg (Iaroslav Koshelev)

Besides DispNet we decided to test other network solving the problem of stereovision. Since the problem of depth or disparity estimation is well known to be pretty close to the problem of semantic segmentation [13], we approached one of the segmentation networks

Fig. 5: Results after adapt Quantized regression to DispNet

well-performing for driving surroundings. SqueezeSeg was implemented in PyTorch and trained on the Driving dataset [14] from scratch. The network itself is a fully convolutional neural network, where large convolutions were replaced with computationally lighter so-called Fire convolution and deconvolution (transposed convolution) layers (see Fig. 6).

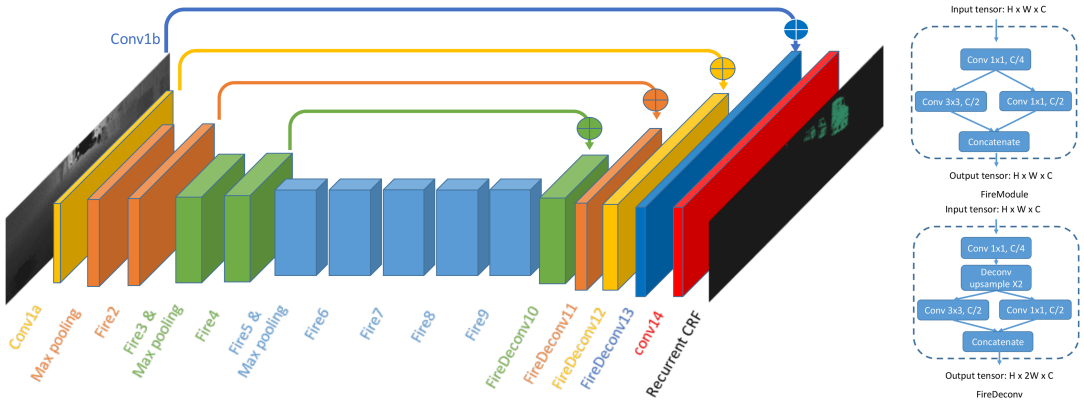


Fig. 6: Architecture of SqueezeSeg CNN

First 10 layers (from conv1 to fire9) perform the feature extraction, so the output of fire9 layer is a down-sampled feature map. Transposed convolution modules are used to obtain full-resolution disparity prediction. Skip connections are used to combine both high and low level fature maps. The loss function was changed from cross-entropy to mean square error (end point error), thus the network may perform on disparity estimation.

Overall network shows a good training and evaluation behavior. This is shown on the Fig. 7, where training and validation losses during training are depicted.

With that network we achieved slightly worse results than with DispNet. The evaluation end point error was abut 8.4 and one map prediction takes about 10ms on GTX1060.

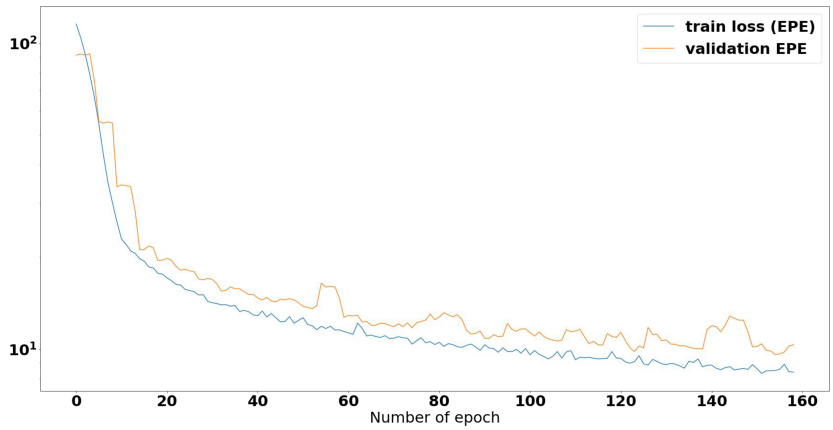


Fig. 7: Training and evaluation losses of SqueezeSeg while training

However, the network trains much faster and easier, than DispNet.

5.4 CycleGAN Domain Adaptation (Aleksandr Safin, Grecia Diaz, Iaroslav Koshelev)

Since both networks were trained on synthetic image dataset and a real KITTI dataset contains only 500 ground truths, we tried to approach neural network style adaptation techniques. We considered different modern GAN networks such as pix2pixHD [15], and CycleGAN [16]. For such networks the dataset consisting of real and synthetic images of the same environment is needed. Since we owned only images from different dataset without strict correspondence between, we might perform only unpaired learning. So CycleGAN was the only choice. The principal architecture of the network is presented on Fig. 8. In words, we ask only to output image from domain B given image from domain A, without explicitly requiring one-to-one correspondence to be given. To achieve this, it was proposed in[16] that one could consider the whole system as: generator from A to B, another generator network which transform image from domain B to image from domain A and a discriminator network. Discriminator should only classify images (fake vs real), thus it guarantee visual realism of images. Moreover, we should require cycle consistency here, meaning that if we start with some image from domain A, then transform it to domain B, and finally transform the obtained image into domain A

We used vKITTI dataset for synthetic images and KITTI dataset for real ones. Both left and right images of stereo-pair of both datasets were consequently adjusted (crop + resize) to match the horizon points.

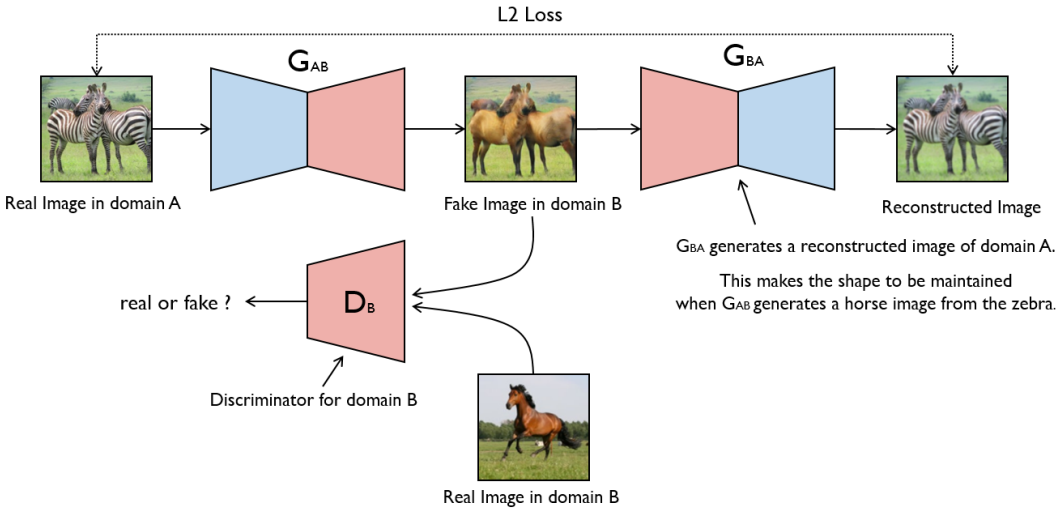


Fig. 8: General architecture of CycleGAN

The obtained style transfer results are presented on Fig. 9.

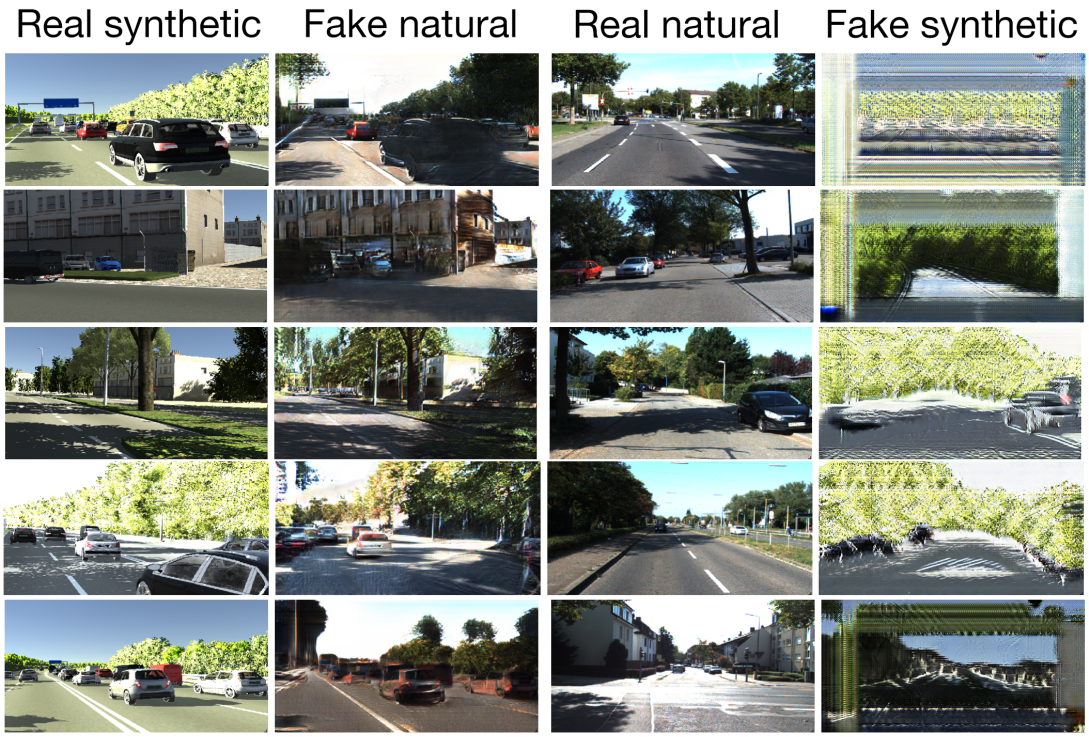


Fig. 9: Obtained results for style transfer

It is clear, that the results are inappropriate by the means of utilizing them in training and/or evaluation procedures of disparity estimators. Besides overall pure fake images quality the main reasons for that are arising artifacts (like fake trees) and disappeared objects.

We tried to solve that by increasing the quality of synthetic dataset. In order to

generate an appropriate dataset we installed AirSim driving environment, however it is unable to move the car by preprogrammed routes to generate necessary pictures.

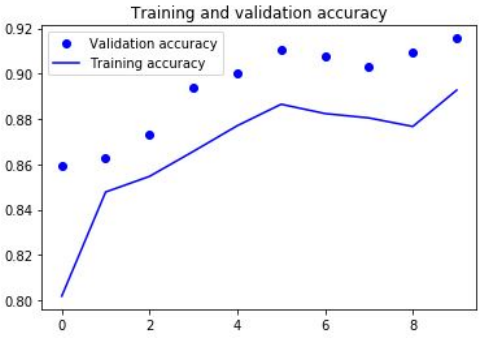
6 Postprocessing (Iurii Minin)

The main idea of this post processing is to define the edges of object and the domain of object that is the nearest to the flying device. Thus, we may define the the nearest object edges in 3D space that will help device to control its movement better. Thus, the main contribution of this work is an active learning methodology application in imaging segmentation system. Here was used only biomedical dataset because problem to find edges of object is so close to define the edges of nuclei. But the results have not been applied to the stereo images. By training a Convolutional Neural Network architecture for semantic segmentation, applying the methods presented in the Section 6.2, this chapter discuses the practical application in medical imaging segmentation, achieving the best possible performance. In this work our CNN is trained end-to-end on MRI volumes depicting prostate, and learns to predict segmentation for the whole volume at once. We optimize objective function during training, based on Dice coefficient.

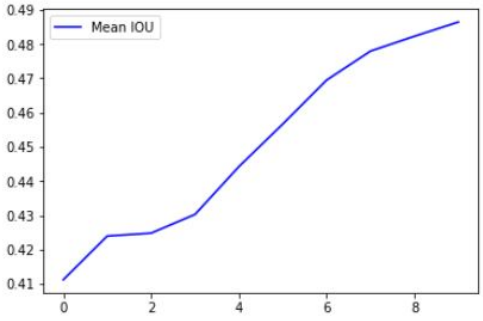
6.1 Convolutional Neural Networks for Image Segmentation

The desired output should include localization, requiring an assignation of a class label to each pixel. This is the main idea of a semantic segmentation using ConvNets. Recent semantic segmentation algorithms [24], convert an existing CNN architecture con-structed for classification to a fully convolutional network (FCN). They obtain a coarse label map from the network by classifying every local region in image, and perform a simple deconvolution, which is implemented as bilinear interpolation, for pixel-level labeling. In addition, novel proposals [27] introduce the idea of deconvolution network to generate a dense pixel-wise class probability map by consecutive operations of unpooling, deconvolution and rectification.

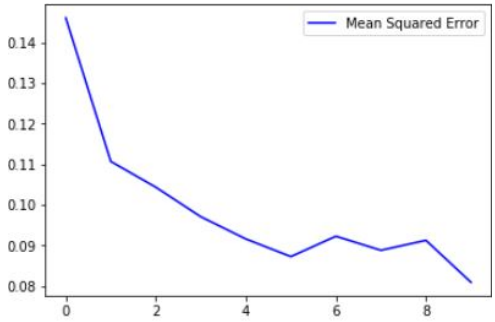
Results of CNN. We have used 6 convolutional layers, 4 Leaky ReLUs and 4 Dropouts. Figures 10 represents losses and accuracies of model after 10 epochs.



Training and validation accuracies

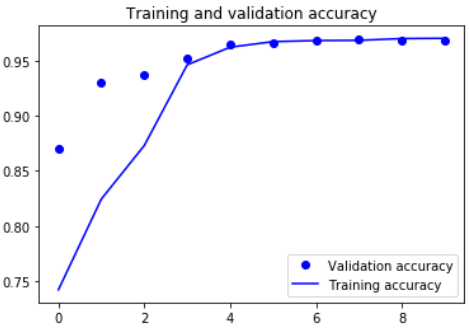


Mean IoU

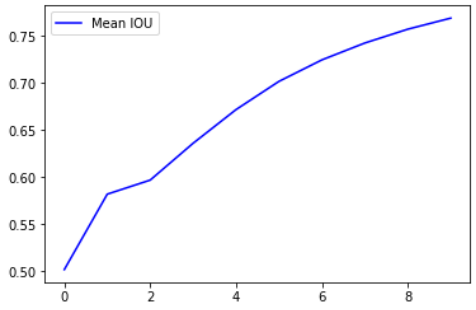


Mean Squared Error

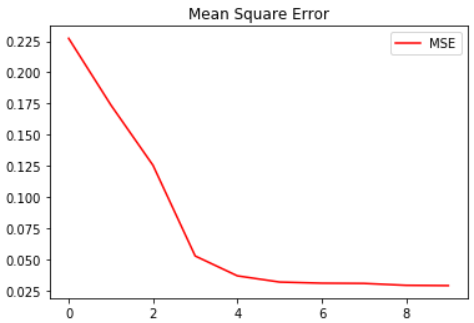
Fig. 10: For Convolutional Neural Networks applied for Image Segmentation



Training and validation accuracies



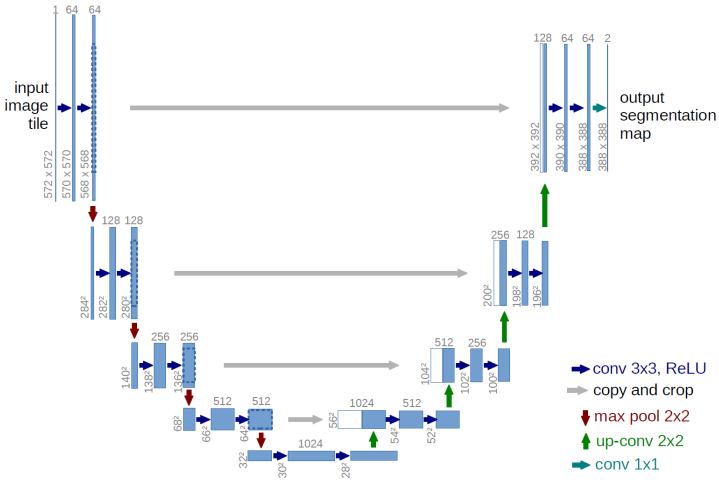
Mean IoU



Mean Squared Error

Fig. 11: For U-Net architecture applied for Image Segmentation

Fig. 12: U-Net architecture (example)



6.2 U-Net architecture

The U-Net [22] is a CNN to solve Biomedical Image Segmentation problems. It has won ISBI cell tracking challenge[23]. The network architecture is illustrated in Figure 6.2. As explained in Section 6.1, the network merges a convolutional network architecture with a deconvolutional architecture to obtain the semantic segmentation. The convolutional network is composed of a repetitive pattern of two 3×3 convolutions operations, followed by a ReLU layer and a downsampling process through a 2×2 maxpooling operation with stride 2.

Results of U-Net. Figures 11 represents that after 10 epochs, the MSE (mean square error) stopped at 0.0291. Mean IoU values = 0.769. Loss = $Dice\ Coefficient$ loss function.

7 Team member contributions

- **Aleksandr Safin** has implemented DispNet (along with weights scheduling) in PyTorch, worked on dataloading pipeline and has carried out experiments regarding DispNet training. Also has made pretty gif animation regarding which was demonstrated on the presentation.

- **Grecia Diaz** generated training data from AirSim by accessing the APIs and modifying it in C++ and also implemented Quantized Regression in Pytorch.

- **Iaroslav Koshelev** has implementerd, trained and tested SqueezeSeg in PyTorch,

worked on style transfering (CycleGAN), datasets and their pre-processing.

- **Iurii Minin** was responsible for programming of postprocessing part of project, the project presentation representation and the project report writing. He has implemented 2 neural networks (U-Net and Convolutional Neural Network with Adam optimizer and dropout techniques) as postporocessing augmentation to the main proceeding neural networks but he did not combine these mentioned neural networks into one automatized neural system. He has represented the project presentation but didn't answer questions during the presentation representing.

8 Things we learned

It seems strange, but it turns out that there is a lack of open synthetic dataset for autonomous car driving and drone navigation. And it is not that easy to generate images by ourselves, although finally we managed it.

Regarding DispNet and SqueezeSeg training:

- BatchNorm is vital for training
- Weight scheduling is reasonable technique and is really required
- Quantized regression (only on last layer) does not help
- Quantized regression on every upsampling layers also does not help
- Really fast inference, less than 5ms per image
- Fine-tuning for quantized regression yields similar results as from the scratch

Approach	EPE
DispNet (no BN)	109
DispNet	6.7
DispNet-QR	9.4
DispNet-QR (fine-tuning)	9.05
SqueezeSeg	8.4

Moreover, it is sad but true, Movidius (Vision Processing Unit) does not support some operations (slicing and transposing). Actually, there were more work done regarding drone, but it is related to the 'Sensors for IoT' course.

Regarding postprocessing:

Deep learning in Computer Vision; Research to find and applied different method to solve problem; Keras frameworks; Found some algorithms for semantic segmentation : R-CNN, DeepLab, PSPnet.

9 Conclusion

Behavior of **DispNet** training loss function (EPE) was signal package in the initial point. A lot of initial high peaks were caused by training images which sizes are getting bigger with stepping. Altitudes of these peaks are bigger than altitudes of peaks of other functions. It is connected with the different type of chosen model estimators and with differences of model approaches. **DispNet** prediction algorithm (has taken 5 ms and 100 iterations to achieve some validation function value to achieve the same value of validation function) is approximately two times faster than **SqueezeSeg - alternative network** prediction algorithm (has taken 10 ms to achieve the same value of validation function) and than **SqueezeSeg** prediction algorithm (has taken 8 ms), but approximately 1.5 times faster than **SqueezeSeg-Adapted** (has taken 160 iterations to achieve the same value of validation function). In addition, we have tried to have success in implementing **Domain Adaptation - CycleGAN**. However, we have succeeded in implementing first pair decoder-encoder. However, not without artifacts such as appearing new tree in the left corner of picture or disappearing street lamps in the artificial picture that should be the same as well as initial one, but the artificial is not exactly the same as initial one after first pair encoder-decoder. However, after the second pair encoder-decoder the picture is getting blurring. That is not acceptable for **DispNet**. Let's consider postprocessing NNs. The **U-net** model had been provided better performance than the original **CNN**. We proceeded the dataset by using deep learning research for challenging stereo vision task. We implemented a fully-convolutional regression approach to establish dense correspondence fields between objects in natural images and three-dimensional object templates. Throughout the paper, we focus on shapes, where applications are abundant and benchmarks allow a fair comparison. We implemented dense regression method out-of-the-box that outperforms a state-of-the-art semantic segmentation approach.

References

[1] G. Papandreou, I. Kokkinos, and P. Savalle. Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In IEEE Conference on Computer Vision and Pattern Recognition, 2015, Boston, MA, USA, June 7-12, 2015,2015.

[2] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. CoRR, abs/1506.02025, 2015.

[3] A. Handa, M. Blösch, V. Patraucean, S. Stent, J. McCormac, and A. J. Davison. gvnn: Neural network library for geometric computer vision. CoRR, abs/1607.07405, 2016.

[4] D. Chen, G. Hua, F. Wen, and J. Sun. Supervised transformer network for efficient face detection. In European Conference on Computer Vision, 2016.

[5] M. Lades, J. C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R. P. Wurtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. IEEE Transactions on computers, 42(3), 1993.

[6] M. Pedersoli, R. Timofte, T. Tuytelaars, and L. Van Gool. An elastic deformation field model for object detection and tracking. International Journal of Computer Vision, 111(2), 2015.

[7] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In Proceedings of the IEEE International Conference on Computer Vision Workshops, 2015.

[8] D. Boscaini, J. Masci, E. Rodolà, and M. M. Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. arXiv preprint arXiv:1605.06437, 2016.

[9] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for oneshot human pose estimation. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 103–110. IEEE, 2012.

[10] G. Pons-Moll, J. Taylor, J. Shotton, A. Hertzmann, and A. Fitzgibbon. Metric regression forests for correspondence estimation. International Journal of Computer Vision, 113(3):163–175, 2015.

[11] L. Wei, Q. Huang, D. Ceylan, E. Vouga, and H. Li. Dense human body correspondences using convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1544–1553, 2016.

[12] Alp Güler, Rıza & Trigeorgis, George & Antonakos, Epameinondas & Snape, Patrick & Zafeiriou, Stefanos & Kokkinos, Iasonas. (2016). DenseReg: Fully Convolutional Dense Shape Regression In-the-Wild.

[13] Arsalan Mousavian, Hamed Pirsiavash, Jana Košecká, Joint Semantic Segmentation and Depth Estimation with Deep Convolutional Networks. Fourth International Conference on 3D Vision (3DV), 2016

[14] Nikolaus Mayer, Eddy Ilg, P. Häusser, Philipp Fischer, D. Cremers, Alexey Dosovitskiy, Thomas Brox. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[15] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. "High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs", arXiv preprint arXiv:1711.11585.

[16] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", in IEEE International Conference on Computer Vision (ICCV), 2017.

[17] Shital Shah and Debadeepta Dey and Chris Lovett and Ashish Kapoor. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. Field and Service Robotics, 2017.

[18] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In International Conference on Learning Representations, 2015.

[19] C. Sagonas, E. Antonakos, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. 300 faces in-the-wild challenge: Database and results. Image and Vision Computing, 47, 2016.

[20] A. Jourabloo and X. Liu. Large-pose face alignment via cnn-based dense 3d model fitting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[21] G. Trigeorgis, P. Snape, M. A. Nicolaou, E. Antonakos, and S. Zafeiriou. Mnemonic descent method: A recurrent process applied for end-to-end face alignment. In Proceedings of IEEE International Conference on Computer Vision & Pattern Recognition, 2016.

[22] P. Fischer O. Ronneberger 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, 2015.

[23] Ieee international symposium on biomedical imaging. cell tracking challenge. [online]

[24] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *CoRR*, abs/1210.5644, 2012.

[25] Andreas Geiger and Philip Lenz and Christoph Stiller and Raquel Urtasun. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*,2013.

[26] Gaidon, A and Wang, Q and Cabon, Y and Vig, E.Virtual Worlds as Proxy for Multi-Object Tracking Analysis.CVPR,2016.

[27] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. *CoRR*, abs/1505.04366, 2015.

[28] Active Deep Learning for Medical Imaging Segmentation, Universitat Politècnica de Catalunya, 2016 - 2017

[29] Generic U-Net Tensorflow implementation for image segmentation

[30] A concise code for training and evaluating Unet using tensorflow+keras

[31] Implementation of Segnet, FCN, UNet and other models in Keras

[32] U-Net: Convolutional Networks for Biomedical Image Segmentation

[33] 2018 Data Science Bowl. Find the nuclei in divergent images to advance medical discovery(online)