Usings CNNs to Estimate Depth from Stereo Imagery

Tyler S. Jordan, Skanda Shridhar

***Abstract*—This paper explores the benefit of using Convolutional Neural Networks in generating a disparity space image for rendering disparity maps from stereo imagery. An eight-layer fully-connected network is constructed with 3200 neurons and trained on 250,000 positive and negative image patch samples. The disparity space image is aggregated using context-aware cross-based method. The disparity map is generated by a "winner takes all" strategy.**

Great improvements can be visually observed in comparison to the naive subtractive plane-sweep method especially in regions with little to no texture.

I. Introduction

Depth-based stereoscopic image rendering and 3D reconstruction has been an important area of research in multimedia, broadcasting and in computer vision. The area has received a lot of attention from the broadcast research community for its applications in 3D Television (3DTV) and Free Viewpoint Television (FTV)[9], [7], [2], [1]. Similarly, depth-based 3D perception is an important prerequisite for computer vision, robotics and augmented reality applications[3], [4], [6]. With the recent advances in omnidirectional camera rigs, computational methods for postproduction, and head-mounted display (HMD) technologies, VR is on its way to become the next cinematic medium. At the core of all of these applications is the ability to produce precise and accurate depth maps for the scene under consideration. The ability to generate reliable depth maps helps in synthesizing novel views, which is a key step in supporting these varied applications.

Humans have an innate ability to perceive depth from stereo imagery; however, conventional stereo correspondence algorithms are generally incapable of producing reliable dense disparity maps. In this project, we have used convolutional neural networks to tackle the problem of matching cost computation in stereo estimation. This was followed by an image processing pipeline that included techniques such as cross based cross aggregation, semiglobal matching and occlusion interpolation for disparity refinement.

II. Algorithm

We chiefly followed Žbontar and LeCun's pipeline [8] shown in Figure 1. The algorithm performs the following four steps: First a cost function is computed which gives a correlation value between the shifted image pairs at each disparity. This cost function aides in determining the disparities of objects. The cost function is intelligently blurred and energy constraints are place on it. The the disparity map is generated by using the disparity which minimizes the cost function in each region. The disparity map is then refined by combining the information from the disparities gathered from both views.

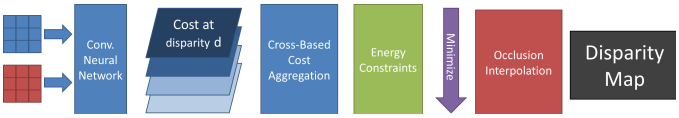

Fig. 1. Pipeline for generating the disparity map

A. Matching cost computation

Stereo correspondence algorithms generally begin by sweeping one of the images of the pair (right over left in our approach) and computing a pixel-wise cost at each disparity value. The resulting stack of costs (Figure 2) is called a disparity space image (DSI)[5]. At each disparity level in the DSI, the regions with the minimum cost are most likely to be at that disparity in the scene.

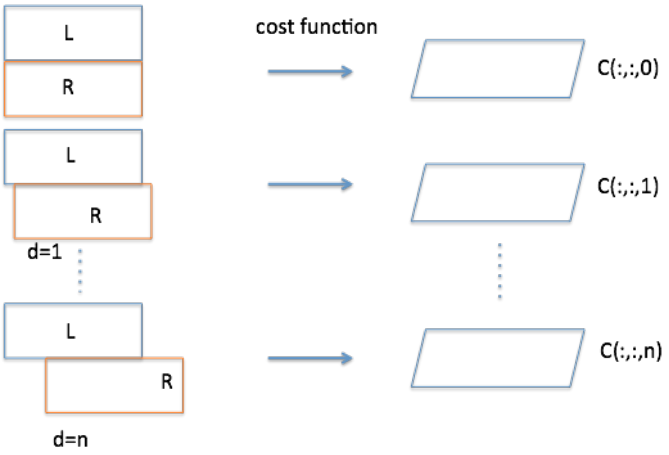

Fig. 2. A cost function is computed between the the input images at every disparity value. The disparity value that minimizes a cost in a given region is used for that region.

Common matching costs include square intensity differences (SD) and absolute intensity differences (AD) [5] such as:

$$C_{AD}(p,d) = \sum_{q \in 9\times 9 \text{ patch}} |I^L(q) - I^R(qd)| \qquad (1)$$

In our approach we have trained a Convolutional Neural Network to output a cost given a left and a right image.

1) The Convolutional Neural Network: We used the Convolutional Neural Network architecture specified in [8]. consisting of 8 layers with Rectified Linear Units inserted in between each layer. While training, the network was configured as a fully connected network (except for the first layer) and while testing it was reconfigured as a convolutional network.

The input to the network is a pair of 9x9 patches, one drawn from the left image and one from the right. The network was trained to recognize well correlated patches and output a corresponding score, by constructing positive training samples using the ground truth data in the KITTI dataset. It was also trained to recognize non-matching samples.

Altogether, 250,000 samples were used to train the CNN, half of which were positive samples, and the remaining half of which were negative. The final layer of the network produces two outputs. The level outputs are passed through a softmax function to separate them and this is outputted as a cost. A cross entropy loss function is minimized using stochastic gradient descent to train the network. The batch size used for the stochastic gradient descent was 128.

While training the network is configured as a fully connected network as this is more computationally efficient. However it is tedious to evaluate an entire image with 9x9 patches alone. Therefore, after training, the fully connected network is reshaped to become convolutional.

For instance, Layer L2 is reshaped so that for each node, the 800 inputs are rearranged in the form of a 5x5x32 filter which can be used to perform a 3D convolution with the 32 5x5 outputs from Layer L1. A similar modification is applied to Layer L3, turning it into a convolutional layer with 1x1x200 filters. In similar fashion, the remaining layers are made convolutional. Selected filters can be seen in Figure 13 in the Appendix.

Once the network has been made convolutional, we can pass full images to it, and we can use it to prepare the disparity space image for the subsequent image pipeline. The intermediate outputs of the network can be seen in Figure 14 in the Appendix.

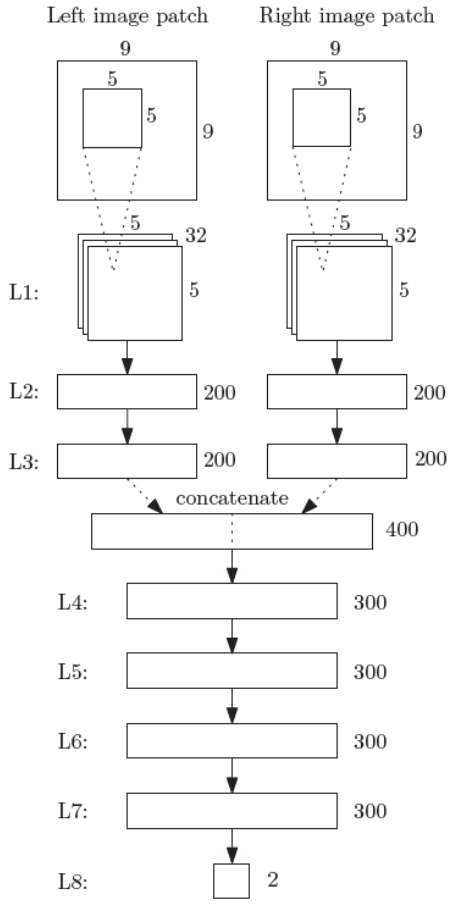

Fig. 3. Architecture of our CNN [8]

B. Cost Aggregation

To ensure that the costs outputted for a given region are smooth within that region cost aggregation is performed. To ensure regions and not just outlier pixels are minimized we want to blur the image; however, we would like to make sure that blurring does not occur across discontinuities (such as edges) in the scene. For example, it would be a bad outcome if the disparity values at the edge of a car were averaged with the wall behind it.

For this reason, we aggregate costs using a context-aware technique called "cross-based cost aggregation" [9]. The cross-based technique essentially creates size-restricted regions of similar color which heuristically correspond to objects assumed to be at the same depth.For each pixel p, in the left and right images, a support region is constructed by creating vertical and horizontal arms of a cross that are limited by a length constraint

$$||p - p_l||_2 < \eta. \tag{2}$$

and a color difference constraint.

$$\max_{c \in \{r,g,b\}} (|I_c(p) - I_c(p_l)|) < \tau \tag{3}$$

Where Žbontar and LeCun only created support regions from grayscale images, we implemented Zhang's original method [9] which compares the intensity of all three color channels.

The support region for each pixel consists of the union of horizontal arms along the vertical arm (Figure 4). The combined support region is the intersection of the support regions in each image given disparity d.

$$U_d(p) = \{q | q \in U^L(p), qd \in U^R(pd)\} \tag{4}$$

Disparity space is aggregated four times over this region for each pixel. This aggregation can be done relatively computationally efficient using integral images. First, each horizontal arm is summed using a horizontal integral image. Then a vertical integral image is created from the result and used to sum along the vertical arm of each pixel. Each integral image cost one addition per pixel and each sum costs one addition per pixel. This saves a lot of computational power when we

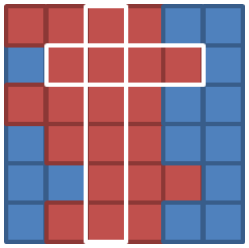

Fig. 4. Cross-based cost aggregation creates support regions around each pixel which consists of the union of all the horizontal arms of the pixels in the vertical arm. Arms for a particular pixel are outlined in white.

only use 4 additions per pixel rather than n-1 where n is the number of pixels in the support region.

After cost aggregation, an energy constraint is placed on the cost function as specified in Žbontar and LeCun's paper [8].

C. Disparity Computation

In our procedure we calculate the pixel-wise disparity using a "winner takes all" approach. For each pixel, the disparity level at which the cost is minimum is recorded as the disparity for that pixel.

D. Disparity Refinement

Using the previous steps we produce disparity maps referenced to both the left and the right images. Next, we can refine these maps using the following heuristic suggested by Žbontar and LeCun [8].

For each pixel in the left disparity map we check to see if a corresponding pixel in the right image shares that disparity to within +/- 1. If it does, we classify it as a match. If it doesn't, we scan the horizontal axis for a pixel that is a match. If we find one, that pixel is most likely an occlusion (Figure 5). If we do not find one it is most likely a mismatch.

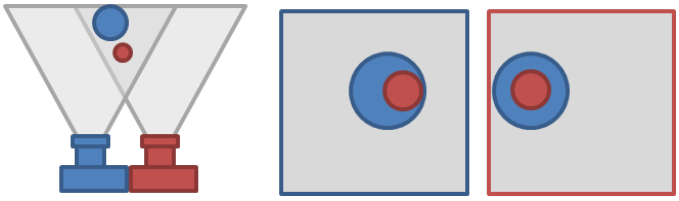

Fig. 5. The right and left cameras capture the scene from different angle (left) so some pixels in the disparity map generated from each view might be occluded in the other(right). When an occlusion region is detected in the left disparity map, pixels from the left are used to fill in the hole.

To resolve occlusions we replace the occluded pixel by the disparity value of the nearest match to the left. To resolve mismatches, we search in sixteen directions until we find a matching pixel in each direction and use the median of these to replace the mismatched value (Figure 6). The refinement can be seen in Figure 7.

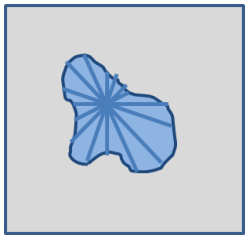

Fig. 6. Regions in the disparity map which have no match are filled in by searching in 16 directions for correct disparities, then assigning the median to the pixel value

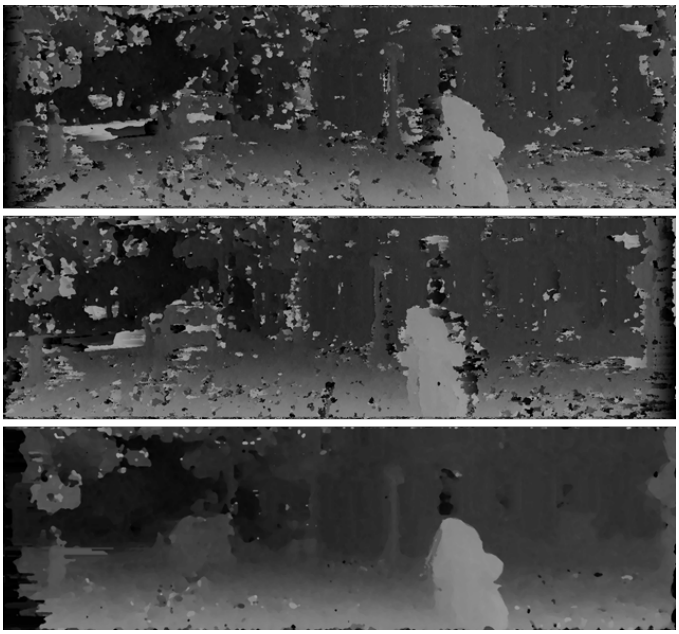

Fig. 7. The top two images are the left and right disparity maps respectively. When they are compared to one another mismatched regions are filled.

III. Results

Using the convolutional neural network approach to create a cost disparity space resulted much cleaner results than the naive plane-sweep with a 9×9 subtractive method specified in Eq. (1). Figures 8, 9, and 8 show a comparison of the two methods. The convolutional neural networks approach does a much better job at estimated disparity of near texture-less surfaces.

From the results images we can see that the disparity maps are not very reliable at the edges which is to be expected because we don't have overlapping data there. Additionally, as we slide the images with respect to each other to generate the cost function, we lose information from the edges as our disparity increases. Another discrepancy occurs not because of the algorithm, but because of optical reflections which trick the cost function to pick disparities which are further away because the image in the reflection is optically further than the reflective object. Besides these small problems, the algorithm is shown to perform quite well.

Fig. 8. (Top) the left image in the stereo pair.(Middle) disparity map created with the convolutional neural network. (Bottom) disparity map created with 9×9 subtractive method. The CNN performs better especially on smooth surfaces such as the road. Edges are not so well defined because of lack of overlapping area between the input images

Fig. 9. (Top) the left image in the stereo pair.(Middle) disparity map created with the convolutional neural network. (Bottom) disparity map created with 9×9 subtractive method.

To test the subjective quality of our disparity maps, we generated red-cyan anaglyphs to judge the quality of the resulting 3D image . We tried two methods: generating only a novel right view from the left image and disparity map, and generating novel left and right views from the the left image and disparity map.

The novel right view is generated by applying the appropriate amount of negative disparity to each pixel in the left image according to the depth map (Figure 11). Similarly, the left novel view is generated by applying the half the positive disparity to each pixel in the input left image. The holes remaining in the novel view from occlusions are filled in with pixels to the right in the right image and to the left in the left image (Figure 12).

The anaglyphs give us a very intuitive way to judge the disparity map quality. We found that they produced fairly realistic 3D scenes. The novel views, however, contain small distortions in certain areas due to the minor imperfections in the disparity map. Though the method which generates only a right image requires more occlusion holes to be interpolated, it produces more appealing results. This is probably due to the minor distortion our disparity map applies to the novel images. When one image is perfect, our brain tends to interpolated it to the other eye, but when both images have distortions the resulting scene does not look as good.

Another way to assess the quality of the results is to compare them with depth maps produced using a naive subtractive approach. As can be seen from Figures 8,9 and 10, using the matching cost produced by the CNN yielded depth maps that are smoother, and without the jarring holes in the maps that result from the naive approach. Also, regions with smooth texture that are traditionally considered difficult to produce depth values for are modeled surprisingly well.

IV. Conclusion

In our project we attempted to use a Convolutional Neural Network to output a matching cost that could then be aggregated and refined to compute pixel-wise disparities. To make training computationally efficient it was necessary to use a fully connected network. To make testing computationally efficient, it was necessary to transform the fully connected network into a convolutional network.

Once matching costs were computed, we made use of a context-aware aggregation scheme called Cross-Based Cost Aggregation. We then estimated disparities using a “winner takes all” minimization approach. We also made use of occlusion interpolation to refine the computed pixel-wise disparities.

We found that the CNN based approach leads to disparity maps that are smoother than those obtained with a naive approach. Regions with low texture which are traditionally considered difficult to produce disparity values for are modelled rather well by this approach. Anaglyphs were generated from the depth maps to subjectively evaluate the results.

The next step step for us is to rigorously compare our performance against ground truth for our data-set. Owing

Fig. 10. (Top) the left image in the stereo pair.(Middle) disparity map created with the convolutional neural network. (Bottom) disparity map created with 9×9 subtractive method. Reflections through off the disparity measure.

Fig. 11. A novel right view (top) is synthesized from the disparity map and a left image. The original right image is shown for comparison (bottom). Notice how much of the scene looks very natural such as the roads and buildings, but smaller features such as signposts are distorted.

to computing constraints we have not yet been able to run exhaustive evaluations, but we are in the process of obtaining these resources, and plan to produce quantitative against the data-set results in the coming days.

V. APPENDIX: INNER-WORKINGS OF THE CNN

ACKNOWLEDGMENT

This was a joint assignment with CS 229 for which we partnered with Jayant Thatte to create the convolutional neural network.

Fig. 12. 3D anaglyphs generated from the original left image and synthesized right view(top), synthesized left and right views (middle), and original stereo imagery (bottom)

REFERENCES

[1] Christoph Fehn. Depth-image-based rendering (dibr), compression, and transmission for a new approach on 3d-tv. In *Electronic Imaging 2004*, pages 93–104. International Society for Optics and Photonics, 2004.

[2] Julien Flack, Philip V Harman, and Simon Fox. Low-bandwidth stereoscopic image encoding and transmission. In *Electronic Imaging 2003*, pages 206–214. International Society for Optics and Photonics, 2003.

[3] Hansung Kim and Adrian Hilton. 3d scene reconstruction from multiple spherical stereo pairs. *International journal of computer vision*, 104(1):94–116, 2013.

[4] Hansung Kim, Muhammad Sarim, Takeshi Takai, Jean-Yves Guillemaut, and Adrian Hilton. Dynamic 3d scene reconstruction in outdoor environments. In *In Proc. IEEE Symp. on 3D Data Processing and Visualization*, 2010.

[5] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.

[6] Miriam Schonbein and Andreas Geiger. Omnidirectional 3d reconstruction in augmented manhattan worlds. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 716–723. IEEE, 2014.

[7] Graham Thomas and Oliver Grau. 3d image sequence acquisition for tv & film production. In *null*, page 320. IEEE, 2002.

[8] Jure Žbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. *arXiv preprint arXiv:1409.4326*, 2014.

[9] Ke Zhang, Jiangbo Lu, and Gauthier Lafruit. Cross-based local stereo matching using orthogonal integral images. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(7):1073–1079, 2009.

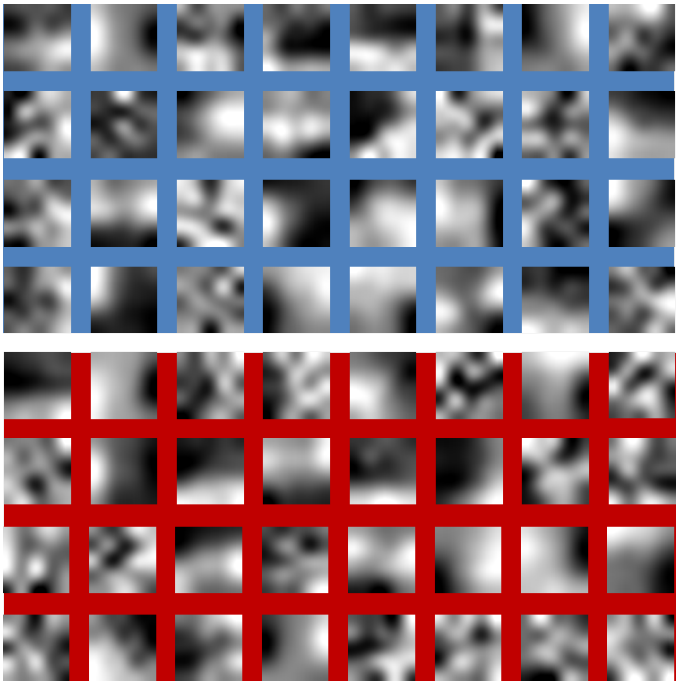

Fig. 13. The 64 filters used in the first layer of the network can be seen to correspond to common-sense filters such as edge-detectors.

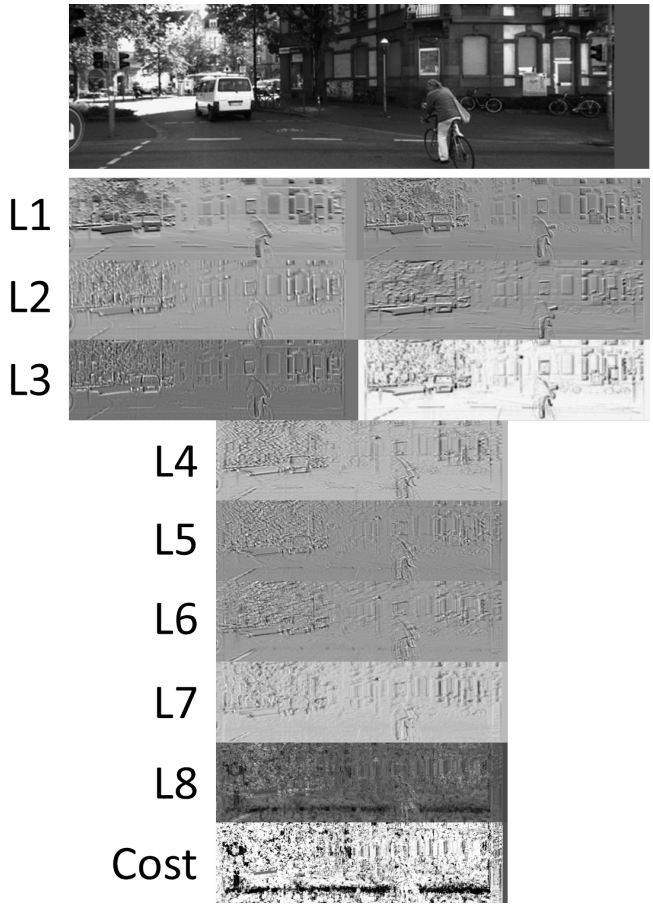

Fig. 14. Selected images from different levels in the network