Universidad Autónoma de San Luis Potosí
Facultad de Ingeniería
Centro de Investigación y Estudios de Posgrado

ModuleNet: A Convolutional Neural Network for Stereo Vision

Para obtener el grado de:
Maestría en Ingeniería de la Computación

Presenta:
Octavio Israel Rentería Vidales

Asesor:
Dr. Juan Carlos Cuevas Tello
Co-Asesor:
Dr. Mariano José Juan Rivera Meraz

San Luis Potosí, S. L. P.	Diciembre de 2020

Abstract

Convolutional Neural Networks (CNN) has gained much attention for the solution of numerous vision problems including disparities calculation in stereo vision systems. In this paper, we present a CNN based solution for disparities estimation that builds upon a basic module (BM) with limited range of disparities that can be extended using various BM in parallel. Our BM can be understood as a segmentation by disparity and produces an output channel with the memberships for each disparity candidate, additionally the BM computes a channel with the out–of–range disparity regions. This extra channel allows us to parallelize several BM and dealing with their respective responsibilities. We train our model with the MPI Sintel dataset. The results show that ModuleNet, our modular CNN model, outperforms the baseline algorithm Efficient Large-scale Stereo Matching (ELAS) and FlowNetC achieving about a 80% of improvement.

Resumen

Las redes neuronales convolucionales (CNN) han ganado mucha atención por la solución de numerosos problemas de visión, incluido el cálculo de disparidades en sistemas de visión estéreo. En este artículo, presentamos una solución basada en CNN para la estimación de disparidades que se basa en un módulo básico (MB) con un rango limitado de disparidades que se puede ampliar utilizando varios MB en paralelo. Nuestro MB puede entenderse como una segmentación por disparidad y produce un canal de salida con las membresías para cada candidato de disparidad, además, el MB calcula un canal con las regiones de disparidad fuera de rango. Este canal extra nos permite paralelizar varios BM y hacer frente a sus respectivas responsabilidades. Entrenamos nuestro modelo con el dataset MPI Sintel. Los resultados muestran que ModuleNet, nuestro modelo modular de CNN, supera al algoritmo de referencia Efficient Large-scale Stereo Matching (ELAS) y FlowNetC logrando aproximadamente un 80 % de mejora.

UASLP
Universidad Autónoma
de San Luis Potosí

FACULTAD DE
INGENIERÍA

6 de agosto de 2020

DR. JUAN CARLOS CUEVAS TELLO
PRESENTE.

Por medio de la presente me permito informarle, que en Sesión Ordinaria del H. Consejo Técnico Consultivo celebrada el día 6 de agosto del presente, fue analizada su petición en la cual solicitó autorización para que el alumno de la **Maestría en Ingeniería de la Computación Octavio Israel Rentería Vidales**, se titule mediante la modalidad: **Publicación de Artículo en Congreso Internacional con Arbitraje o en Revista Indizada**, con el articulo denominado: **"ModuleNet: A Convolutional Neural Network for Stereo Vision".**

Al respecto, me permito informarle que su solicitud fue aprobada de conformidad, de acuerdo a las evidencias que avalan el requerimiento de titulación antes mencionada.

Sin otro particular de momento, le reitero las seguridades de mi atenta y distinguida consideración.

"MODOS ET CUNCTARUM RERUM MENSURAS AUDEBO"

ATENTAMENTE

DR. RICARDO ROMERO MENDEZ
SECRETARIO DEL CONSEJO.

UNIVERSIDAD AUTONOMA
DE SAN LUIS POTOSI
FACULTAD DE INGENIERIA
SECRETARIA

Copia. H. Consejo Técnico Consultivo.
*etn.

"1945-2020: 75 años de formación de profesionales en la Facultad de Ingeniería"

Table of contents

Chapter 1

Paper

This chapter contains the front part and details of the conference proceedings and then the paper as appeared in the proceedings.

> Renteria-Vidales O.I., Cuevas-Tello J.C., Reyes-Figueroa A., Rivera M. (2020) *ModuleNet: A Convolutional Neural Network for Stereo Vision*. In: Figueroa Mora K., Anzurez Marín J., Cerda J., Carrasco-Ochoa J., Martínez-Trinidad J., Olvera-López J. (eds) Pattern Recognition. MCPR 2020. Lecture Notes in Computer Science, vol 12088, pp 219-228. Springer, Cham, ISBN: 978-3-030-49075-1, eISBN: 978-3-030-49076-8, DOI: 10.1007/978-3-030-49076-8_21

Karina Mariela Figueroa Mora ·
Juan Anzurez Marín · Jaime Cerda ·
Jesús Ariel Carrasco-Ochoa ·
José Francisco Martínez-Trinidad ·
José Arturo Olvera-López (Eds.)

Pattern Recognition

12th Mexican Conference, MCPR 2020
Morelia, Mexico, June 24–27, 2020
Proceedings

Springer

Editors

Karina Mariela Figueroa Mora
Facultad de Ciencias Físico Matemáticas
Universidad Michoacana
de San Nicolás de Hidalgo
Morelia, Mexico

Jaime Cerda
Facultad de Ingeniería Eléctrica
Universidad Michoacana
de San Nicolás de Hidalgo
Morelia, Mexico

José Francisco Martínez-Trinidad
Computer Science
Instituto Nacional de Astrofísica,
Óptica y Electrónica
Sta. Maria Tonantzintla, Mexico

Juan Anzurez Marín
Facultad de Ingeniería Eléctrica
Universidad Michoacana
de San Nicolás de Hidalgo
Morelia, Mexico

Jesús Ariel Carrasco-Ochoa
Computer Science
Instituto Nacional de Astrofísica,
Óptica y Electrónica
Sta. Maria Tonantzintla, Mexico

José Arturo Olvera-López
Faculty of Computer Science
Autonomous University of Puebla
Puebla, Mexico

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-030-49075-1 ISBN 978-3-030-49076-8 (eBook)
https://doi.org/10.1007/978-3-030-49076-8

LNCS Sublibrary: SL6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

ModuleNet: A Convolutional Neural Network for Stereo Vision

O. I. Renteria-Vidales[1,2], J. C. Cuevas-Tello[1], A. Reyes-Figueroa[2],
and M. Rivera[2]( )

[1] UASLP Universidad Autónoma de San Luis Potosí,
Álvaro Obregón 64 Col. Centro, 78000 San Luis Potosí, México
[2] CIMAT Centro de Investigación en Matemáticas A.C., Jalisco S/N Col. Valenciana,
36023 Guanajuato, México
mrivera@cimat.mx

Abstract. Convolutional Neural Networks (CNN) has gained much attention for the solution of numerous vision problems including disparities calculation in stereo vision systems. In this paper, we present a CNN based solution for disparities estimation that builds upon a basic module (BM) with limited range of disparities that can be extended using various BM in parallel. Our BM can be understood as a segmentation by disparity and produces an output channel with the memberships for each disparity candidate, additionally the BM computes a channel with the out–of–range disparity regions. This extra channel allows us to parallelize several BM and dealing with their respective responsibilities. We train our model with the MPI Sintel dataset. The results show that ModuleNet, our modular CNN model, outperforms the baseline algorithm Efficient Large-scale Stereo Matching (ELAS) and FlowNetC achieving about a 80% of improvement.

Keywords: Stereo vision · Convolutional Neural Networks · U-Net · Census transform · Deep learning

1 Introduction

The purpose of an stereo system is to estimate the scene depth by computing horizontal disparities between corresponding pixels from an image pair (left and right) and has been intensively investigated for several decades. There is a wide variety of algorithms to calculate these disparities that are complicated to include them all in one methodology or paradigm. Scharstein and Szeliski [13] propose a block taxonomy to describe this type of algorithms, following steps such as matching cost calculation, matching cost aggregation, disparity calculation and disparity refinement. One example is ELAS, an algorithm which builds a disparities map by triangulating a set of support points [8].

We present a CNN based solution for disparities estimation that builds upon a basic module (BM) with limited range of disparities that can be extended using various BM in parallel. Our BM can be understood as a segmentation

K. M. Figueroa Mora et al. (Eds.): MCPR 2020, LNCS 12088, pp. 219–228, 2020.
https://doi.org/10.1007/978-3-030-49076-8_21

by disparity and produces an output channel with the memberships for each disparity candidate, additionally the BM computes a channel with the out–of–range disparity regions. This extra channel allows us to parallelize several BM and dealing with their respective responsibilities. We list our main contributions as follows: i) We propose ModuleNet, which is a novel modular model to measure disparities on any range, which is inspired on FlowNet and U-Net. ii) We use a low computational time algorithm to measure cost maps. iii) The architecture of our model is simple, because it does not require another specialized networks for refinement as variants of FlowNet do for this problem. iv) Our model improves the baseline model ELAS and FlowNetC (the correlation version of FlowNet) with about 80% of unbiased error.

The paper is organized as follows: Sect. 2 presents the related work. At Sect. 2 are the algorithms FlowNet, Census transform and ELAS. The proposed model is in Sect. 3. Section 4 describes the dataset used in this research. At the end are our results, conclusions and future work.

2 Related Methods

In recent years, Convolutional Neural Networks (CNN) have made advances in various computer vision tasks, including estimation of disparities in stereo vision. Fischer et al. propose a CNN architecture based on encoder-decoder called FlowNet [6]. This network uses an *ad hoc* layer for calculating the normalized cross-correlation between a patch in the left image and a set of sliding windows (defined by a proposed disparity set) of the right window and uses Full Convolutional Network (kind encoder-decoder architecture) for estimate the regularized disparity [11]. Park and Lee [9] use a siamese CNN to estimate depth for SLAM algorithms. Their proposal is to train a twin network that transforms patches of images and whose objective is to maximize the normalized cross correlation between corresponding transformed patches and minimize it between non-corresponding transformed patches. To make the inference of the disparity in a stereo pair, a left patch and a set of displaced right patches are used, then the normalized cross correlation between the twin networks transformed patches and the disparity is selected using a Winner–Takes–All (WTA) scheme. Other authors use a multi-scale CNN, where the strategy is to estimate the disparity of the central pixel of a patch by processing a pyramid of stereo pairs [4]; and the reported processing time for images in the KITTI database is more than one minute [7]. A state of the art method with really good results is reported by Chen and Jung [3], they use a CNN that is fed with patches of the left image and a set of slipped patches of the right image (3DNet). Then, for a set of proposed disparities, the network estimates the probability that each of the disparities corresponds to the central pixel of the left image patch that requires of evaluate as many patches as pixels, so it is computationally expensive.

In this section, we present FlowNet, an architecture designed for optical flow, and it can be used for stereoscopy. Also, this section introduces the Census Transform.

2.1 FlowNet

FlowNet is composed by two main blocks. The network computes the local vector that measure the dissimilarity between each pixel (x, y) in the left image I_l and its corresponding candidate pixel $(x + \delta, y)$, for a given disparity δ, in the right image I_r; where $\delta \in d$ with $d = [d_1, d_2, \ldots, d_h]$ and d_i is an integer value. This block is deterministic (not trainable) and produces a dissimilarity map (tensor) D of size equal to $(h, nrows, ncolumns)$. FlowNet is based on the U-Net [11]. The network computes the regularized disparities d^*; with dimension equal to $(1, nrows, ncolumns)$. The main disadvantage of this method is the computational cost of the normalized cross-correlation layer and it also produces blurred disparity maps [6], see in Fig. 1 the FlowNetC architecture.

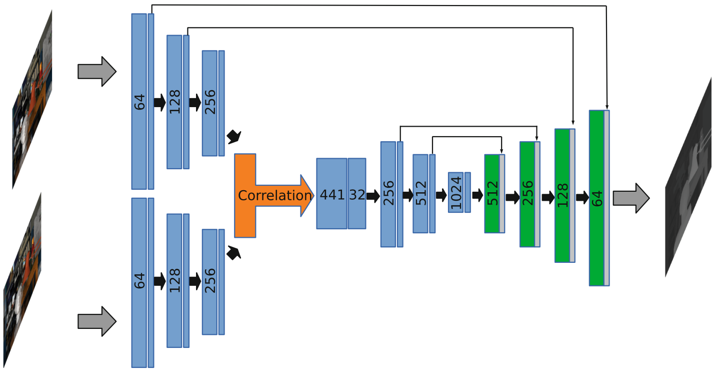


Fig. 1. FlowNet architecture.

2.2 Census Transform

Differently to FlowNet, that uses normalized cross-correlation to measure the cost maps, an alternative is Census Transform [15]. Other algorithms for this task are Sum of Absolute Differences (SAD) [14], Sum of Square Differences (SSD) [14], Normalized Cross-Correlation [5]. Because a low complexity cost function is desirable, we chose the Census Transform [15]. Figure 2 exemplify the Census algorithm, where it transforms the values of the neighbors. The values of the neighbors of a pixel are encoded within a binary chain (it is assigned “1” when they are greater than or equal to the central pixel, or “0” otherwise). This chain is called census signature, the signature retains spatial information of each neighbor given the position within the string where each bit is stored.

For a 3 × 3 window, the census signature contains eight values and can be saved in one byte, this transformation can be computed with:

$$C_l(x, y) = Bitstring_{(i,j)\in w}(I_l(i, j) \geq I_l(x, y)) \qquad (1)$$

for the case of the left image I_l; and in a similar is computer the census transform C_r for the right image I_r. To obtain the level of correspondence, the Hamming

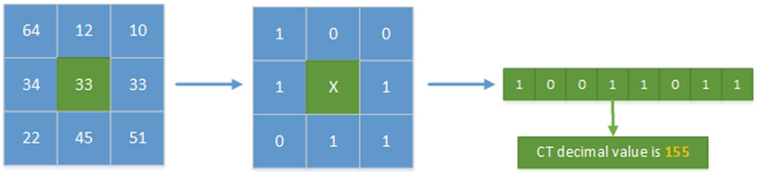


Fig. 2. Census transform

distance (H) is used to count how many bits are different between two census signatures:

$$D_m(x, y; d) = H(C_l(x, y), C_r(x + d_m, y)) \tag{2}$$

We can denote this stage by the representational function F_c that transforms the information in the images I_l and I_r into the distance tensor $D = [D_1, D_2, \ldots, D_h]$:

$$D = F_c(I_l, I_r; d) \tag{3}$$

where the parameters are the set of candidate disparities, d.

3 ModuleNet: Modular CNN Model for Stereoscopy

Our proposed model (ModuleNet) builds upon U-Net blocks and is inspired on the FlowNet. First, we describe the general block U-Net (see Fig. 3) that can find disparities in a range d. Second, we introduce the cascade U-Net for refinement, see Fig. 4. Finally, the modular CNN model (ModuleNet) for disparities out of the range d is presented, see Fig. 5.

3.1 General Block: U-Net U-Net Module

Our neural network for stereo disparity estimation is composed with blocks based on the UNet. Indeed, the most basic construction block can be seen as a simplified version of the FlowNet where the Disparity Map D is computed with the Hamming distances between the Census transformed patches (the fixed and the δ-displaced one). Another difference between our basic block and the FlowNet model is that, instead of computing directly a real valued map of disparities, we estimate the probability that a particular candidate disparity δ is the actual one at each pixel. We also compute an additional layer that estimates outliers: the probability that the actual disparity in each pixel is not included in the set of disparities d. As input to the U-Net, we have h channels of distances corresponding to the h candidate disparities and, as output, we have $h+1$ probability maps; see Fig. 3. We can represent this U-Net block by the representational function F_1 that transforms the information in the distance tensor $D = [D_1, D_2, \ldots, D_h]$ into the probabilities tensor $P = [P_1, P_2, \ldots, P_h, P_{h+1}]$:

$$P = F_1(D, \theta_1) \tag{4}$$

where θ_1 are the network weight set.

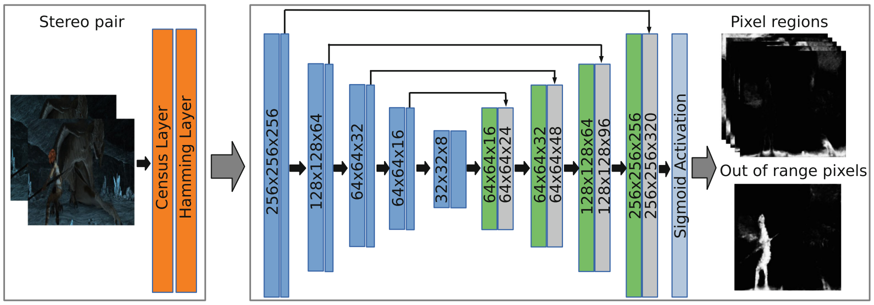


Fig. 3. General block (U-Net)

The representational U-Net F_1 (4) can be seen as a regularizer of the noisy Census-distance maps. We observed that the output of the basic (trained) block can be refined by a second U-Net. This second U-Net (in cascade) is trained using as input the census cost maps, the initial estimation of the disparity probabilities maps and the outliers' probability map and produces as output refined versions of the inputs. We represent this U-Net block by the representational function F_2 that refine probabilities tensor P using also as input the distance tensor D:

$$\hat{Y} = F_2(P, D, \theta_2) \tag{5}$$

where θ_2 are the weight set. We denote our basic module for disparity estimation by

$$D = F_c(I_l, I_r; d) \tag{6}$$

$$\hat{Y} = F(D) \overset{def}{=} F_2(F_1(D), D). \tag{7}$$

where we omitted the parameters θ_1 and θ_2 in order to simplify the notation. Once we have trained a basic module (7), it can be used for estimating disparities into the range defined by the disparity set d. The regions with disparities outside such a range are detected in the outliers' layer. Figure 4 depicts our block model based on two cascaded U-Nets (general blocks, see Fig. 3).

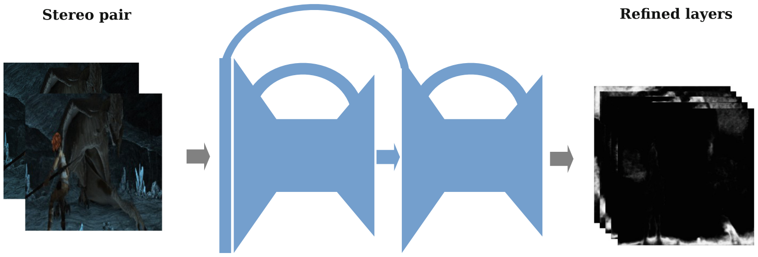


Fig. 4. Our Basic Block composed with two U-Net in cascade.

3.2 ModuleNet: Modular CNN Model

Assume, we have a trained basic module for the disparities into the interval $[d_1, d_h]$ and the actual range of disparities, in the stereo pair, lays into the interval $[d_1, 2d_h]$. We can reuse our basic model for processing of such a stereo pair if we split the calculations for the disparities sets $d^{(0)} = [d_1, d_2, \ldots, d_h]$ and $d^{(1)} = [d_{h+1}, d_{h+2}, \ldots, d_{2h}]$. Then, we can compute two census distance tensors $D^{(0)} = F_c(I_r, I_l; d^{(0)})$ and $D^{(1)} = F_c(I_r, S\{I_l, h\}; d^{(1)} - h)$; where we define the shift operator

$$S\{I, d_h\} \overset{def}{=} I(x + d_h, y). \tag{8}$$

Thus, we can estimate the probability that the disparity is in the set $d^{(0)}$ with $\hat{Y}^{(0)} = F(D^{(0)})$ and in the set $d^{(1)}$ with $\hat{Y}^{(1)} = F(D^{(1)})$; where F is our basic module 7.

This idea can be extended for processing stereo pair with a wide range of disparities. First we define the k-th set of disparities as

$$D^{(k)} = F_c(I_r, S\{I_l, kh\}; d^{(k)} - kh) \tag{9}$$

for $k = 1, 2, \ldots, K$. Second, we estimate, in parallel, the K tensor of probability:

$$\hat{Y}^{(k)} = F(D^{(k)}) \tag{10}$$

Note that the network F is reused for processing the K modules. The CNN transforms the representation $D^{(k)}$ into $\hat{Y}^{(k)}$: the probability that disparities $\delta^{(k)}$ of the module k at the pixel (x, y) are the correct displacement. To estimate the tensor $\hat{Y}$ that integrates the individual probability tensors $\hat{Y}^{(k)}$'s, we use the additional layer with the probability that the correct displacement of each pixel is not the k-th interval:

$$\hat{Y}_{(kh+i)} = \hat{Y}_i^{(k)} \odot \left(1 - \hat{Y}_{h+1}^{(k)}\right) \tag{11}$$

for $i = 1, 2, \ldots, h$, $k = 0, 1, \ldots, K - 1$ and $\odot$ denotes the element-wise product. Finally, the disparity estimation, d^* is computed by applying a WTA procedure in the disparities map $\hat{Y}$:

$$d^*(x, y) = \arg\max_l \hat{Y}_l(x, y) \tag{12}$$

for $l = 1, 2, \ldots, Kh$. Figure 5 depicts ModuleNet – our modular model.

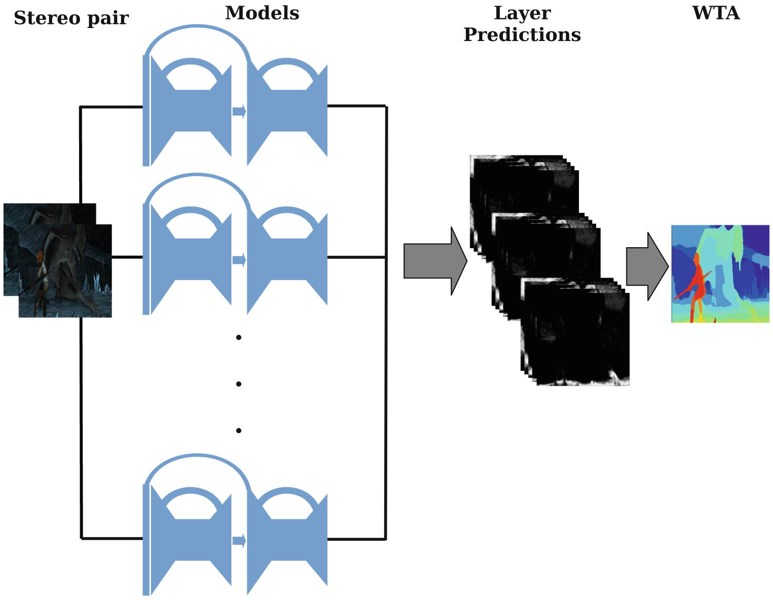


Fig. 5. ModuleNet: Modular CNN Model

4 Dataset and Training Parameters

We used the MPI Sintel dataset for train our model. The MPI Sintel-stereo dataset is a benchmark for stereo, produced from the open animated short film Sintel produced by Ton Roosendaal and the Blender Foundation [1]. This dataset contains disparity maps for the left and right image, occlusion masks for both images. The dataset consist of 2128 stereo pairs divided in clean and final pass images. The left frame is always the reference frame. For our experiments, we use the clean subset pairs that consist of 1064 pairs; 958 for training and 106 for testing. See example in Fig. 6, the disparity map is the ground truth. Our training set consisted on patches (256×256 pixels) randomly sampled from of 958 stereo pairs (1024×460 pixels) and 106 stereo pairs were leave-out for testing.

We change the number of filters distributions across the layers according to Reyes-Figueroa et al. [10]. It has been shown that in order to have more accurate features and to recover fine details, more filters are required in the upper levels of U-Net and less filters in the more encoded levels. Our model's architecture is summarized in Fig. 3. We trained our model during 2000 epochs with mini-batches of size eight.

We used data augmentation by randomly resizing the frames (random scaling factor into the range $[.6, 1]$), adding Gaussian noise (mean zero with standard deviation equal 1% the images' dynamic range). The ADAM optimization algorithm was used with fixed $lr = 1 \times 10^{-4}$ and $\beta = [0.9, 0.999]$. For processing the data set, we used a basic block with sixteen disparities ($h = 16$) and $K = 24$ parallels blocks.

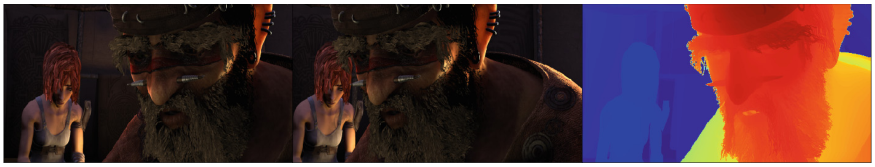


Fig. 6. Example of MPI Sintel data: left and right images and disparity map.

5 Results

In Fig. 7 are shown the results from seven scenes by using the MPI Sintel dataset. We show a single image per scene for illustrating the algorithm's performance. We compare the results from our model versus ELAS and FlowNetC. Visually, one can see that the proposed model is closer to the ground truth than ELAS and FlowNetC.

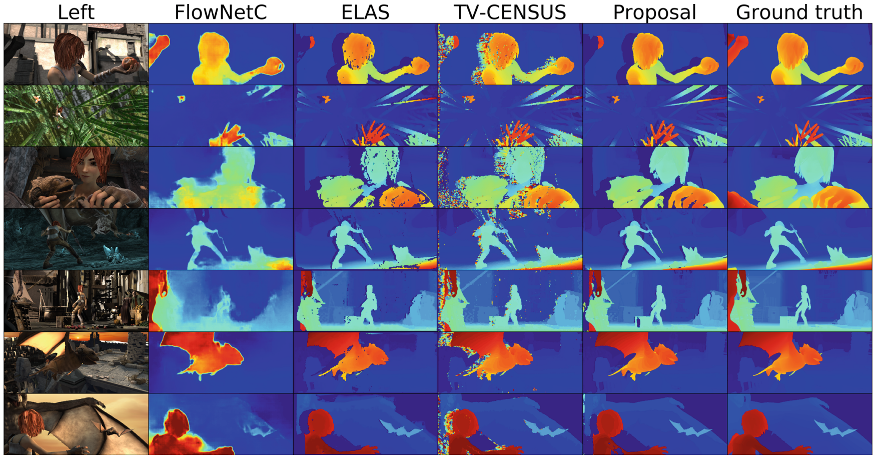


Fig. 7. Results from MPI Sintel dataset on selected scenes

In Table 1 is the comparison of results from applying a Total–Variation potential for edge–preserving filtering to the Distance Tensor D (here named TV–Census) [2], ELAS, FlowNetC and our proposal (ModuleNet); in bold font the best results. We use the metric Mean Absolute Error (MAE) in non-occluded areas to measure the results quantitatively. Our proposed model outperforms the compared methods. The advantage of the MPI Sintel dataset is that the ground truth is provided, so the accuracy (MAE) is unbiased. Show particular results from seven representative stereo pairs and the average over the total of frames. Additionally we tested our method with the Middlebury Stereo Datasets 2014 [12] which consist of 33 image pairs, divided in 10 evaluation test sets with hidden ground truth, 10 evaluation training sets with ground truth and 13 additional sets, the first 20 sets are used in the new Middlebury Stereo Evaluation. Figure 8 shows a visual comparison of the computed results.

Table 1. MAE results from MPI Sintel dataset on selected scenes

Scene	FlowNetC	ELAS	TV–Census	Proposed
alley_1	2.98	2.98	0.92	**0.44**
bamboo_1	2.91	2.39	0.63	**0.51**
bandage_2	14.09	12.77	2.60	**2.14**
cave_2	3.95	3.10	1.85	**0.65**
market_2	1.94	2.07	0.54	**0.43**
temple_2	2.26	2.44	0.60	**0.38**
temple_3	6.09	2.85	0.74	**0.43**
All test images	24.3	14.1	1.7	**1.5**

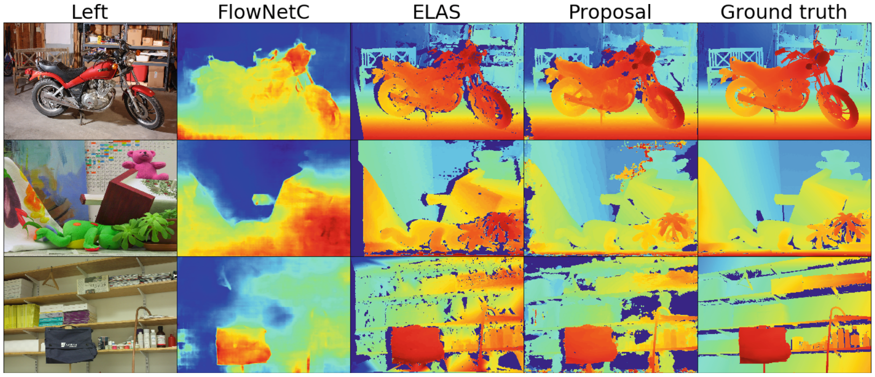


Fig. 8. Results from Middlebury dataset on selected stereo pairs

6 Conclusions and Future Work

We proposed a new model called ModuleNet for disparities estimation that can be applied in stereoscopy vision. Our model is build upon FlowNet, U-Net and Census transform. The modularity of our method allows generating disparity maps of any size simply by adding more blocks. The extra layer, for detecting pixels with disparities out of range, helps us to classify pixels that usually adds noise since these pixels are outside the range of work or are pixels of occluded regions. Our results show that qualitatively and quantitatively our model outperforms Census–Hamming approach (robustly filtered), ELAS and FlowNetC; which are baseline methods for disparities estimation. The unbiased error was improved by about 80%.

Our future work will focus on extend the training set with real stereo pairs, conduct more exhaustive evaluations and implement our model on an embedded system (e.g. NVIDIA® Jetson Nano^{TM} CPU-GPU and Intel®Movidius^{TM} USB stick). We plan to compare the performance of our model with other state-of-the-art methods, regardless the complexity and computational time with GPU hardware. As most of the methods, the texture-less regions are difficult to identify. So an algorithm to detect such textures is desired.

Acknowledges. Part of this work was conducted while O. Renteria was at IPICYT AC, SLP-Mexico. This work was supported in part by CONACYT, Mexico (Grant A1-S-43858).

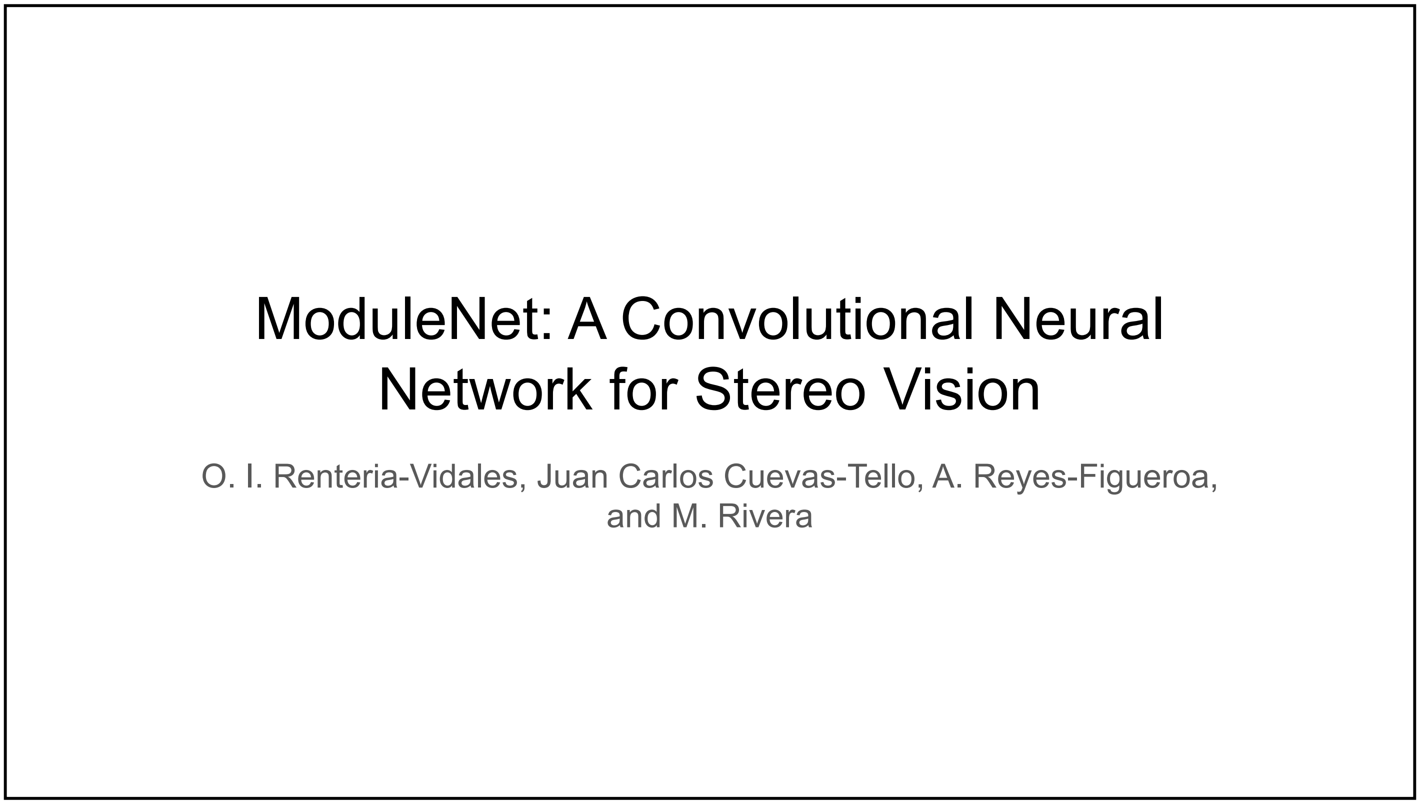
References

1. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7577, pp. 611–625. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33783-3_44
2. Charbonnier, P., Blanc-Féraud, L., Aubert, G., Barlaud, M.: Deterministic edge-preserving regularization in computed imaging. IEEE Trans. Image Process. **6**(2), 298–311 (1997)
3. Chen, B., Jung, C.: Patch-based stereo matching using 3D convolutional neural networks. In: 25th ICIP, pp. 3633–3637 (2018)
4. Chen, J., Yuan, C.: Convolutional neural network using multi-scale information for stereo matching cost computation. In: ICIP, pp. 3424–3428 (2016)
5. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis, XVII, p. 482. Wiley, New York (1973)
6. Fischer, P., et al.: FlowNet: learning optical flow with convolutional networks. In: CoRR, pp. 2758–2766 (2015)
7. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: CVPR, pp. 3354–3361 (2012)
8. Geiger, A., Roser, M., Urtasun, R.: Efficient large-scale stereo matching. In: Kimmel, R., Klette, R., Sugimoto, A. (eds.) ACCV 2010. LNCS, vol. 6492, pp. 25–38. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19315-6_3
9. Park, J., Lee, J.: A cost effective estimation of depth from stereo image pairs using shallow siamese convolutional networks. In: IRIS, pp. 213–217, October 2017
10. Reyes-Figueroa, A., Rivera, M.: Deep neural network for fringe pattern filtering and normalisation (2019). arXiv:1906.06224)
11. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
12. Scharstein, D., et al.: High-resolution stereo datasets with subpixel-accurate ground truth. In: Jiang, X., Hornegger, J., Koch, R. (eds.) GCPR 2014. LNCS, vol. 8753, pp. 31–42. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11752-2_3
13. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comp. Vision **47**(1), 7–42 (2002). https://doi.org/10.1023/A:1014573219977
14. Tamura, H., Mori, S., Yamawaki, T.: Textural features corresponding to visual perception. IEEE Trans. Sys. Man Cybern. **8**, 460–473 (1978)
15. Zabih, R., Woodfill, J.: Non-parametric local transforms for computing visual correspondence. In: Eklundh, J.-O. (ed.) ECCV 1994. LNCS, vol. 801, pp. 151–158. Springer, Heidelberg (1994). https://doi.org/10.1007/BFb0028345

Chapter 2

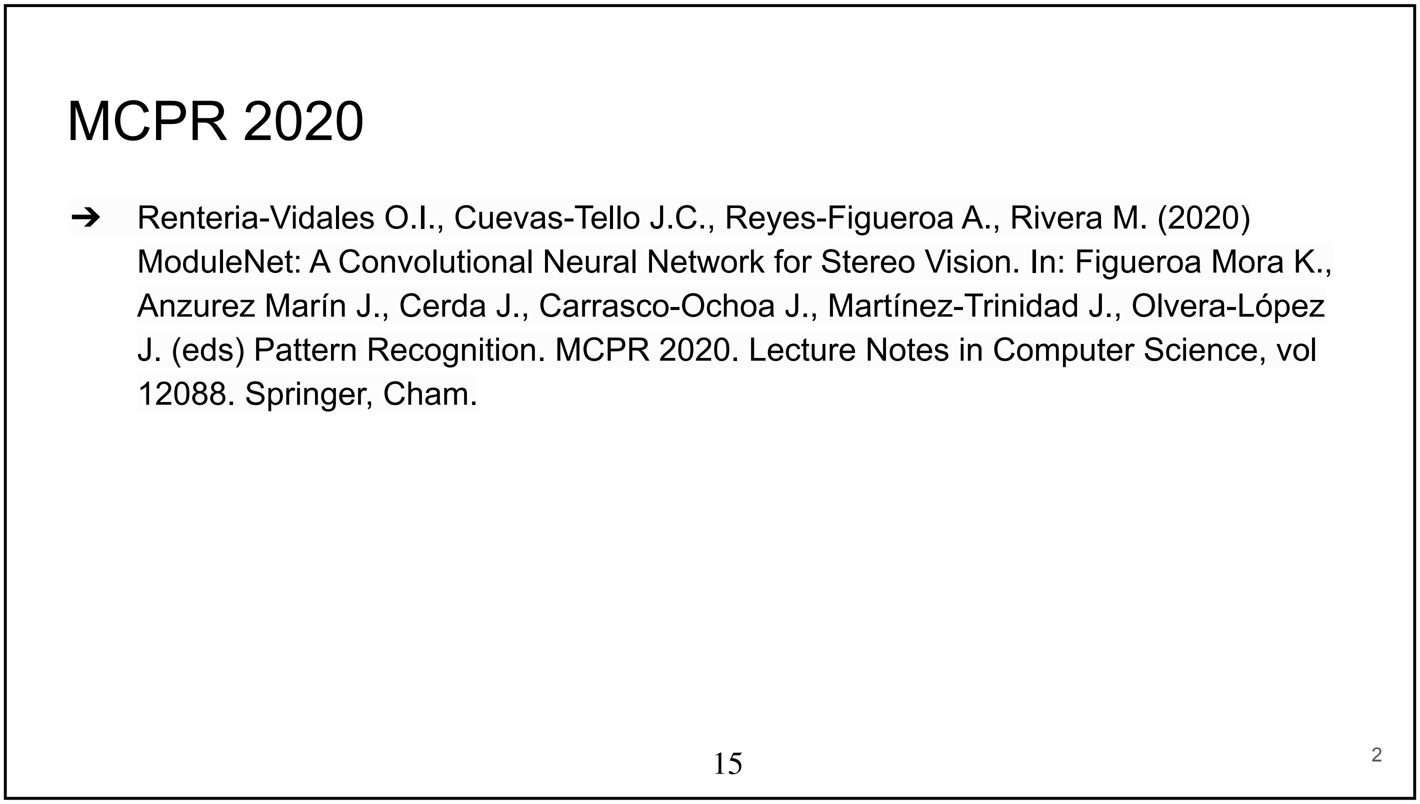
Presentation

This chapter contains the slides of the paper presentation at the conference. Some slides were added after the previous exam in order to improve the presentation.



ModuleNet: A Convolutional Neural Network for Stereo Vision

O. I. Renteria-Vidales, Juan Carlos Cuevas-Tello, A. Reyes-Figueroa, and M. Rivera



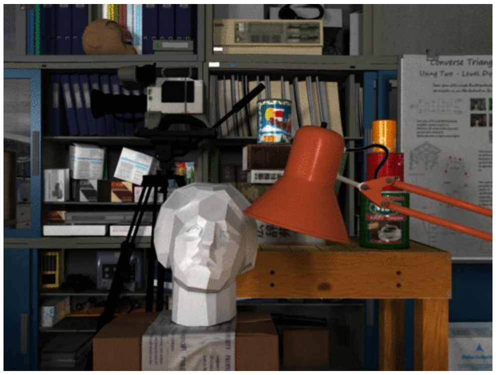
MCPR 2020

➔ Renteria-Vidales O.I., Cuevas-Tello J.C., Reyes-Figueroa A., Rivera M. (2020) ModuleNet: A Convolutional Neural Network for Stereo Vision. In: Figueroa Mora K., Anzurez Marín J., Cerda J., Carrasco-Ochoa J., Martínez-Trinidad J., Olvera-López J. (eds) Pattern Recognition. MCPR 2020. Lecture Notes in Computer Science, vol 12088. Springer, Cham.

Introduction

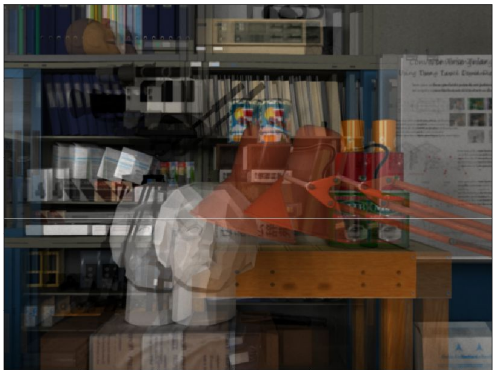
Introduction

The purpose of an stereo system is to estimate the scene depth by computing horizontal disparities between corresponding pixels from an image pair (left and right) and has been intensively investigated for several decades

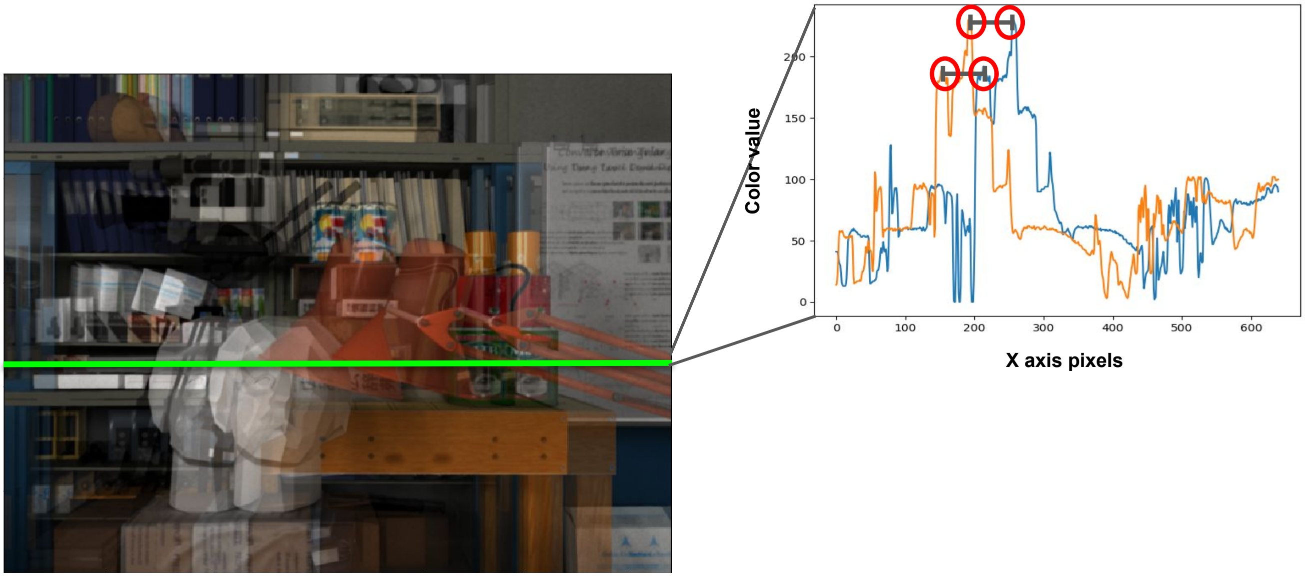


Introduction

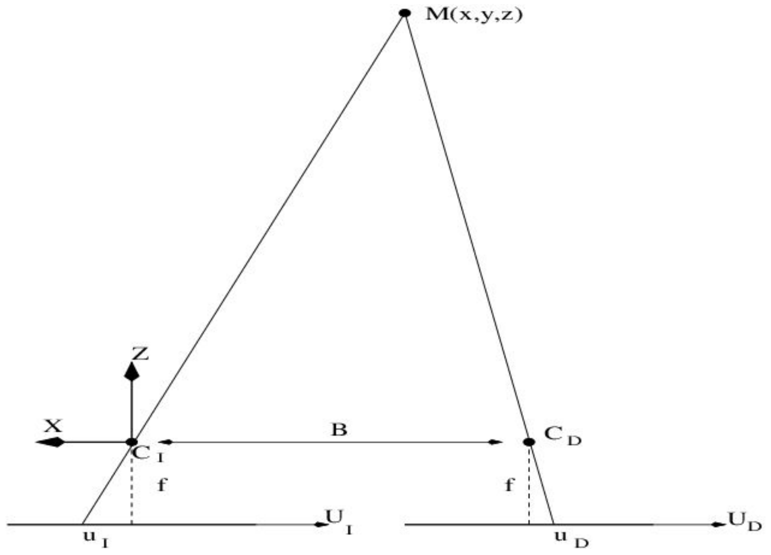
The purpose of an stereo system is to estimate the scene depth by computing horizontal disparities between corresponding pixels from an image pair (left and right) and has been intensively investigated for several decades



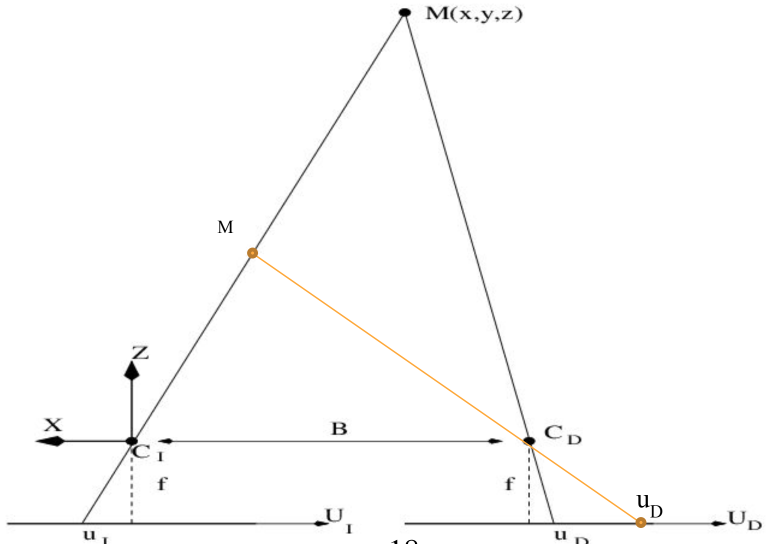
Introduction



Disparity and depth

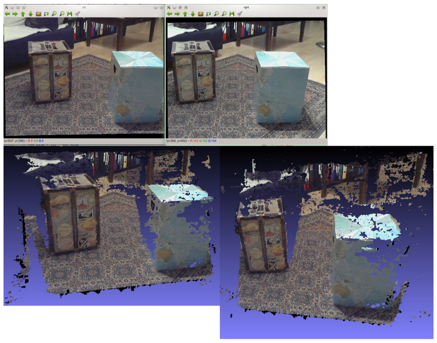


Disparity and depth

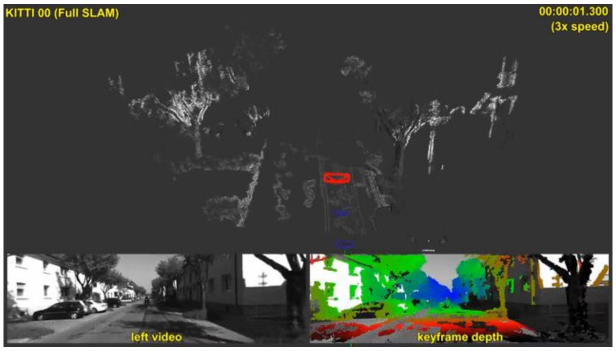


Applications

With these disparities, the depth of each pixel can be calculated to generate a 3D scene. Multiple applications include the scanning of 3D objects, reconstruction of navigation maps for autonomous robots, among others.



3D escene



Navigation using 3D information

Disparity calculation taxonomy

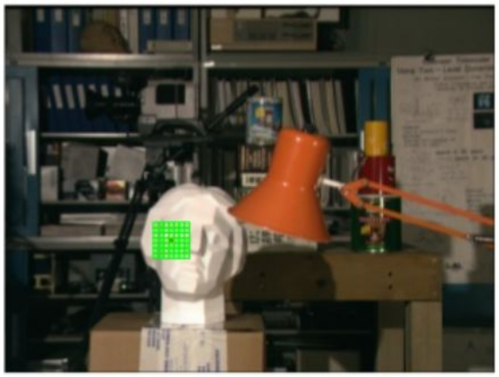
Scharstein and Szeliski propose a block taxonomy to describe this type of algorithms:

- Matching cost calculation
- Matching cost aggregation
- Disparity calculation
- Disparity refinement

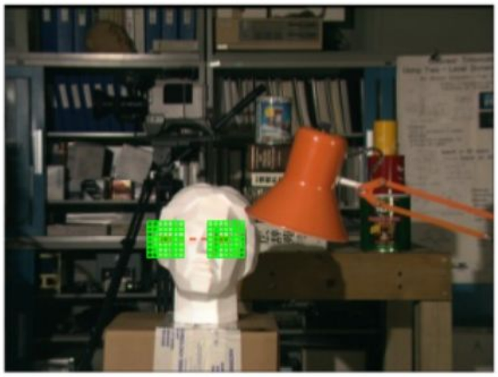
Scharstein, D., Szeliski, R. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. 2015

Cost Calculation

Cost calculation



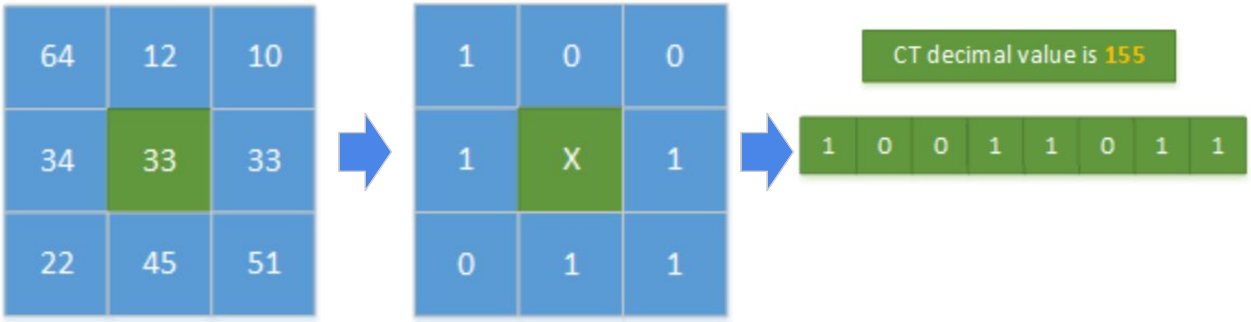
Reference



Target

Cost calculation

A very popular cost function due to its low computational level, is the transformed census:



64	12	10
34	33	33
22	45	51

1	0	0
1	X	1
0	1	1

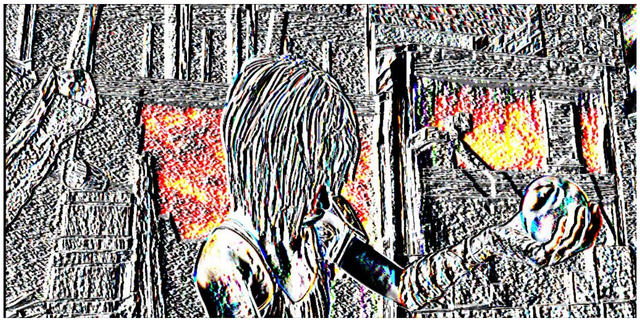
CT decimal value is 155

1	0	0	1	1	0	1	1

Zabih, R., Woodfill, J.: Non-parametric local transforms for computing visual correspondence. 1994

Cost calculation

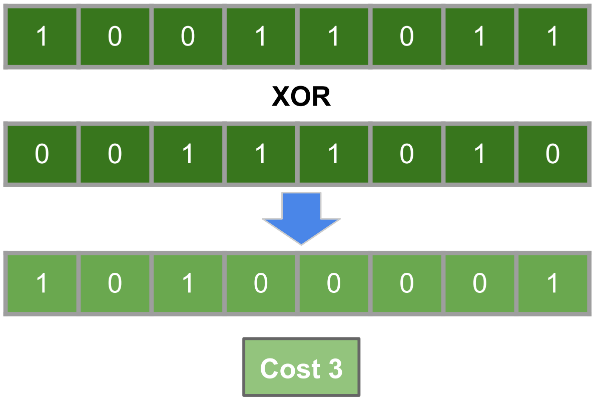
Input image



Census transform

Cost calculation

The hamming distance is used to obtain the cost map, consist in to count the number of different bits



1 0 0 1 1 0 1 1

XOR

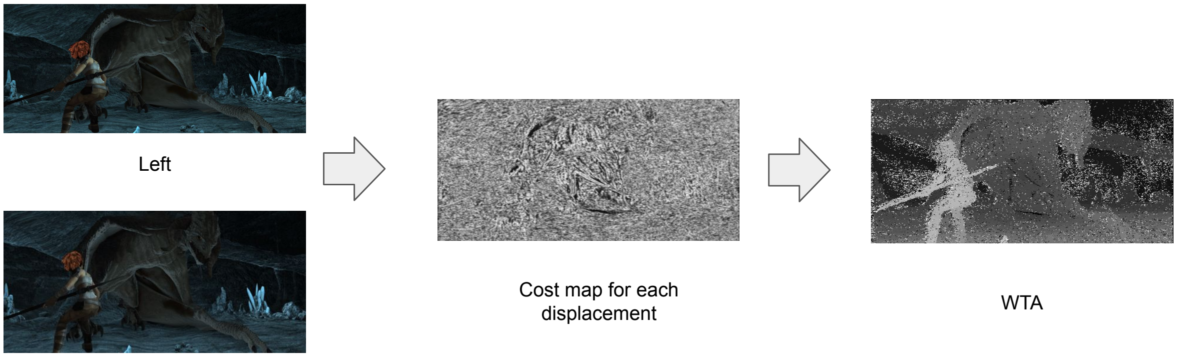
0 0 1 1 1 0 1 0

1 0 1 0 0 0 0 1

Cost 3

WTA

The final displacement map consists of taking the pixel with the lowest cost in each map D, this technique called WTA (winner takes all).



Left

Right

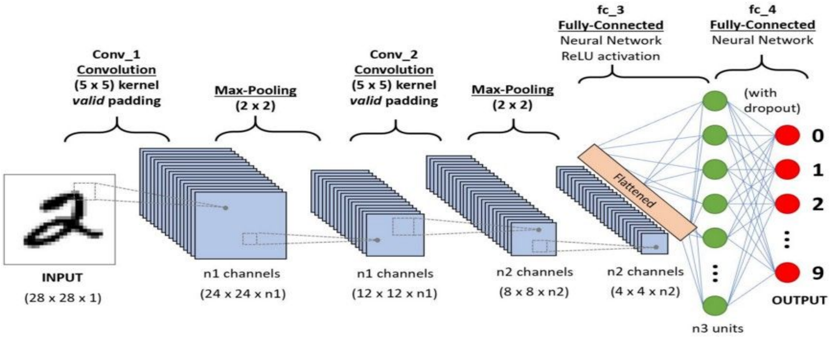
Cost map for each displacement

WTA

Convolutional Neural Networks

Convolutional Neural Networks

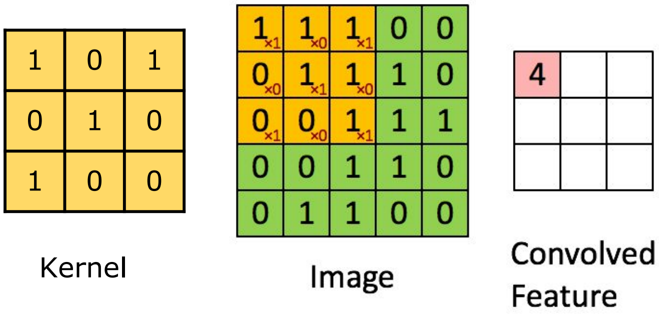
A Convolutional Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.



A CNN sequence to classify handwritten digits

Convolutional Neural Networks

The origin of de CNN is the "neocognitron". Introduced by Kunihiko Fukushima in 1980. The neocognitron introduced the two basic types of layers in CNN: convolutional layers, and downsampling layers. A convolutional layer contains units whose receptive fields cover a patch of the previous layer. The weight vector (the set of adaptive parameters) of such a unit is often called a filter.



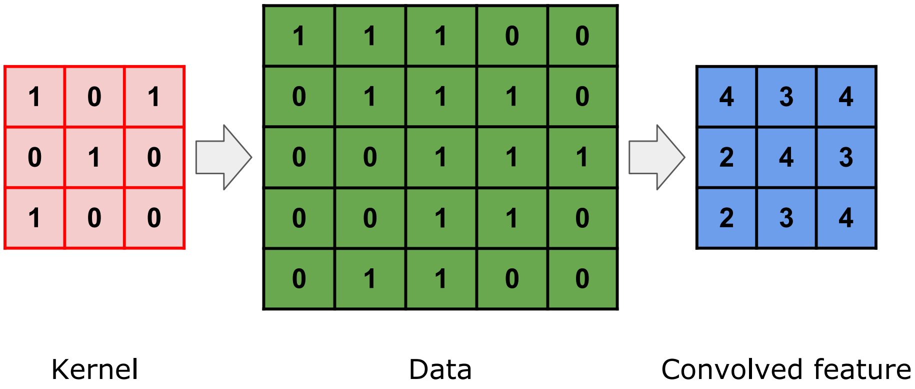
1	0	1
0	1	0
1	0	0

$1_{\times1}$	$1_{\times0}$	$1_{\times1}$	0	0
$0_{\times0}$	$1_{\times1}$	$1_{\times0}$	1	0
$0_{\times1}$	$0_{\times0}$	$1_{\times1}$	1	1
0	0	1	1	0
0	1	1	0	0

4		

Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybernetics* 36, 193–202 (1980).

Convolutional Neural Networks



1	0	1
0	1	0
1	0	0

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

4	3	4
2	4	3
2	3	4

Convolutional Neural Networks

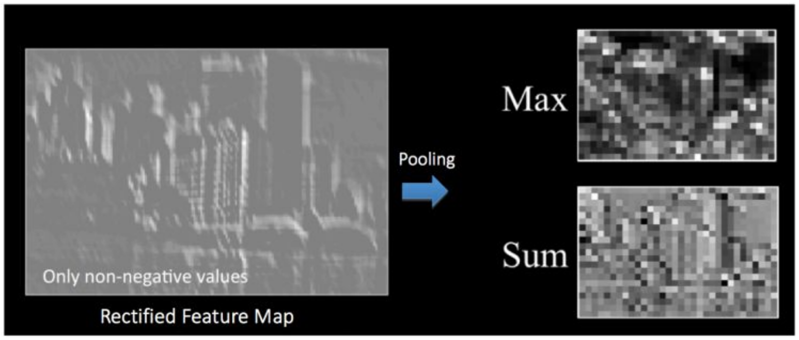
The origin of de CNN is the "neocognitron". Introduced by Kunihiko Fukushima in 1980. The neocognitron introduced the two basic types of layers in CNN: convolutional layers, and downsampling layers. A convolutional layer contains units whose receptive fields cover a patch of the previous layer. The weight vector (the set of adaptive parameters) of such a unit is often called a filter.

Convolution example

Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybernetics* 36, 193–202 (1980).

Convolutional Neural Networks

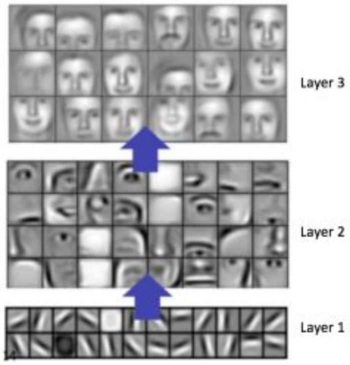
Spatial Pooling (also called subsampling or downsampling) reduces the dimensionality of each feature map but retains the most important information. Spatial Pooling can be of different types: Max, Average, Sum etc.



Pooling example

Convolutional Neural Network

In general, the more convolution steps we have, the more complicated features our network will be able to learn to recognize.



Learned features from a Convolutional Deep Belief Network

Honglak Lee, *et al*, “Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations”

Convolutional Neural Networks Evolution

- Neocognitron (1980)
- LeNet (1990s)
- AlexNet (2012)
- ZF Net (2013)
- GoogLeNet (2014)
- VGGNet (2014)
- ResNets (2015)
- DenseNet (2016)
- YOLO V2 (2017)
- YOLO V3 (2018)
- EfficientNet (2019)

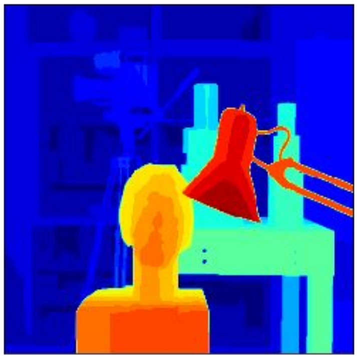
Cost calculation using Convolutional Neural Networks

Cost calculation using CNN

The main focus in various CNN proposals is the calculation of similarity costs, that is to find the relevant pixels for each displacement. This can be seen as a segmentation problem:

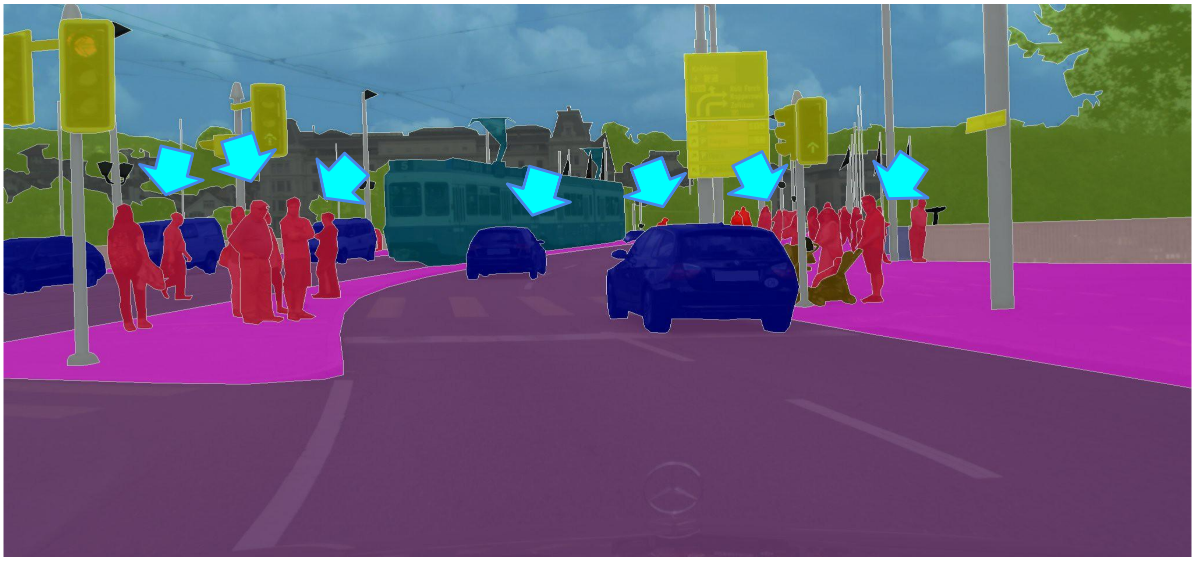
Left

Right



Disparity levels

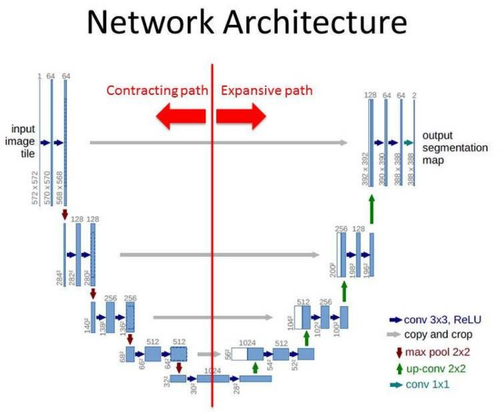
Semantic Segmentation



U-Net

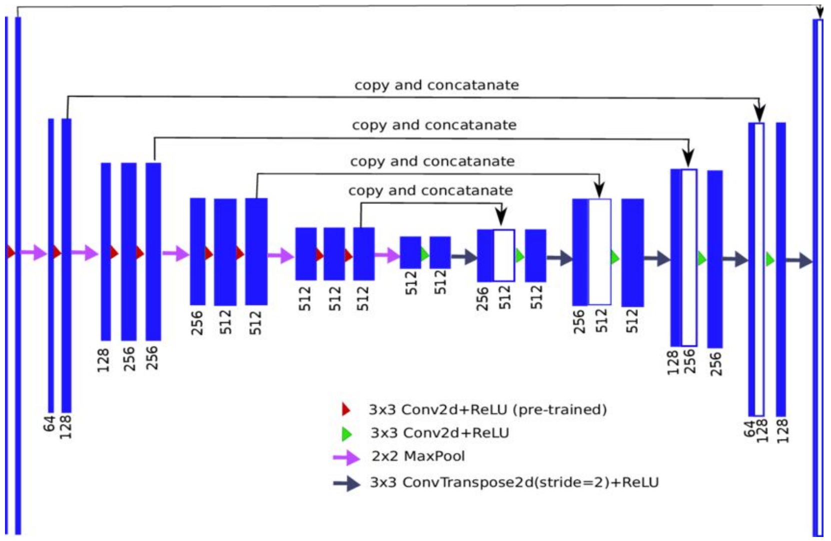
U-Net architecture is separated in 3 parts:

1. The contracting/downsampling path
2. Bottleneck
3. The expanding/upsampling path



Ronneberger, Olaf; Fischer, Philipp; Brox, Thomas (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation".

U-Net



Ronneberger, Olaf; Fischer, Philipp; Brox, Thomas (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation".

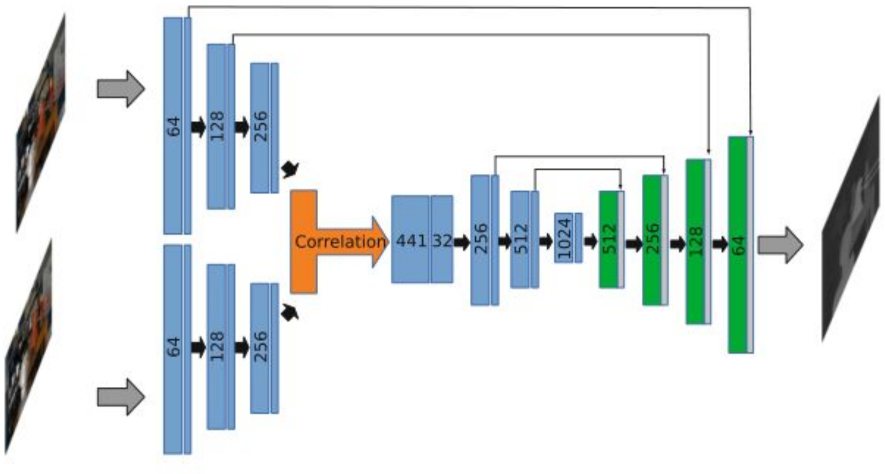
Cost calculation using CNN

We take inspiration in a U-Net like netwok called FlowNetC.

This architecture utilizes two independent streams for the images and to combine them at a later stage, With this architecture the network is constrained to first produce meaningful representations of the two images separately and then combine them on a higher level. To aid the network to find correspondences between images, the network utilizes a custom ‘correlation layer’ that performs multiplicative patch comparisons between two feature maps.

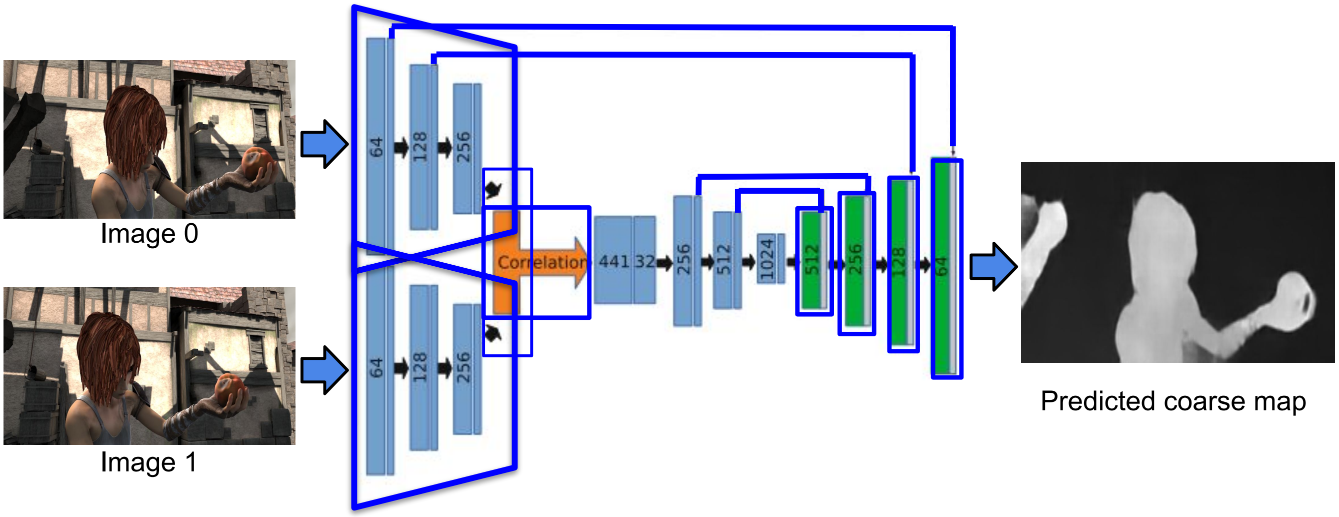
A. Dosovitskiy et al., "FlowNet: Learning Optical Flow with Convolutional Networks," 2015

Cost calculation using CNN



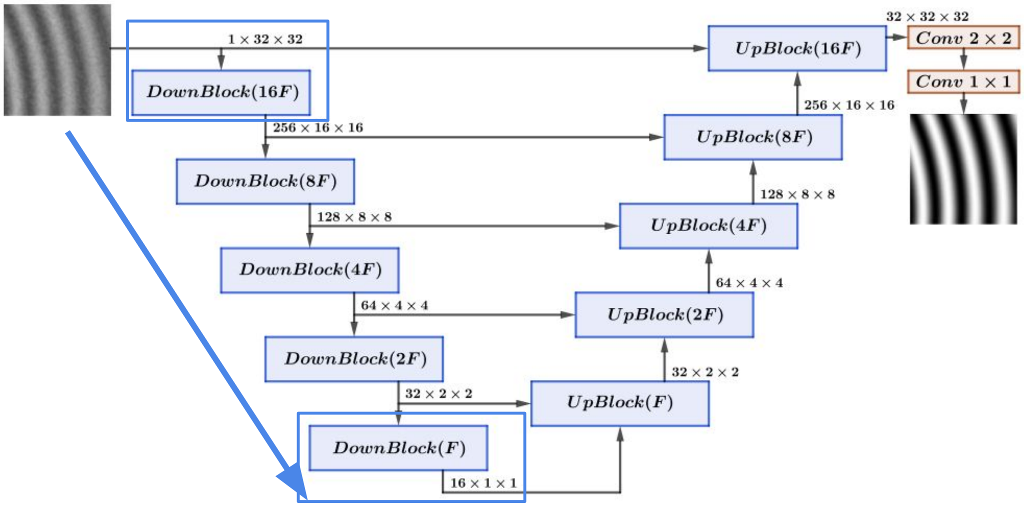
A. Dosovitskiy et al., "FlowNet: Learning Optical Flow with Convolutional Networks," 2015

FlowNetC



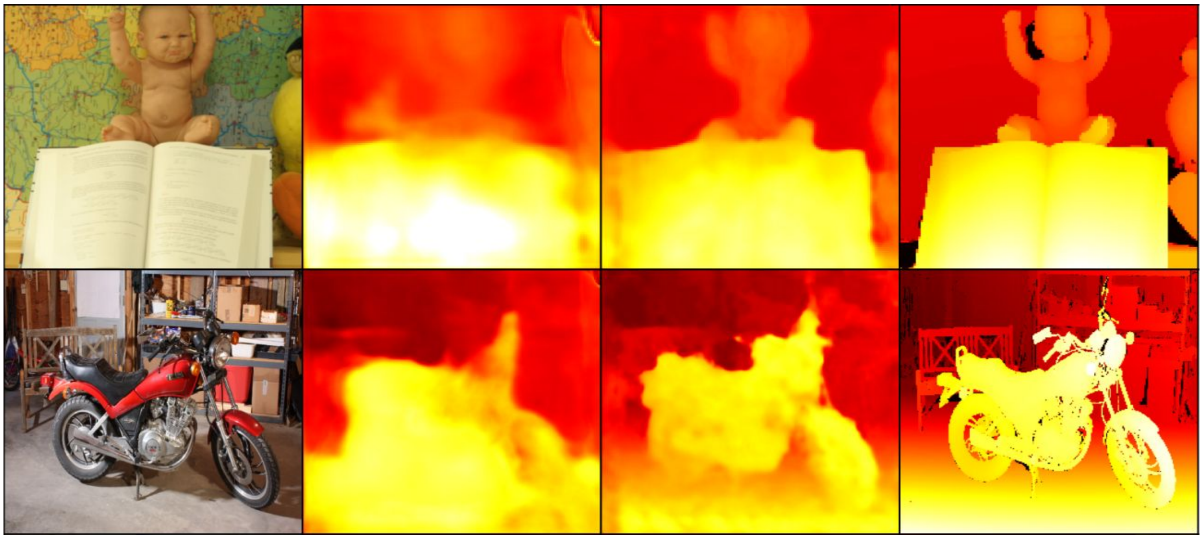
ModuleNet

V-Net



Reyes-Figueroa, A., Rivera, M.: Deep neural network for fringe pattern filtering and normalisation (2019)

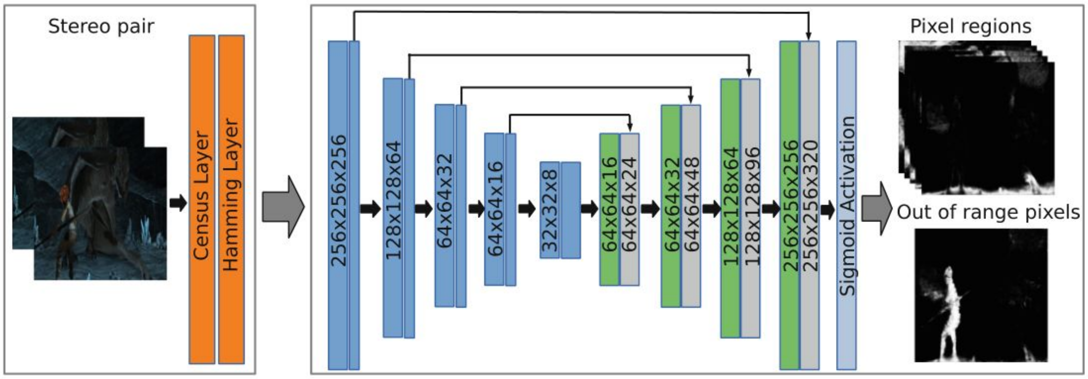
V-Net



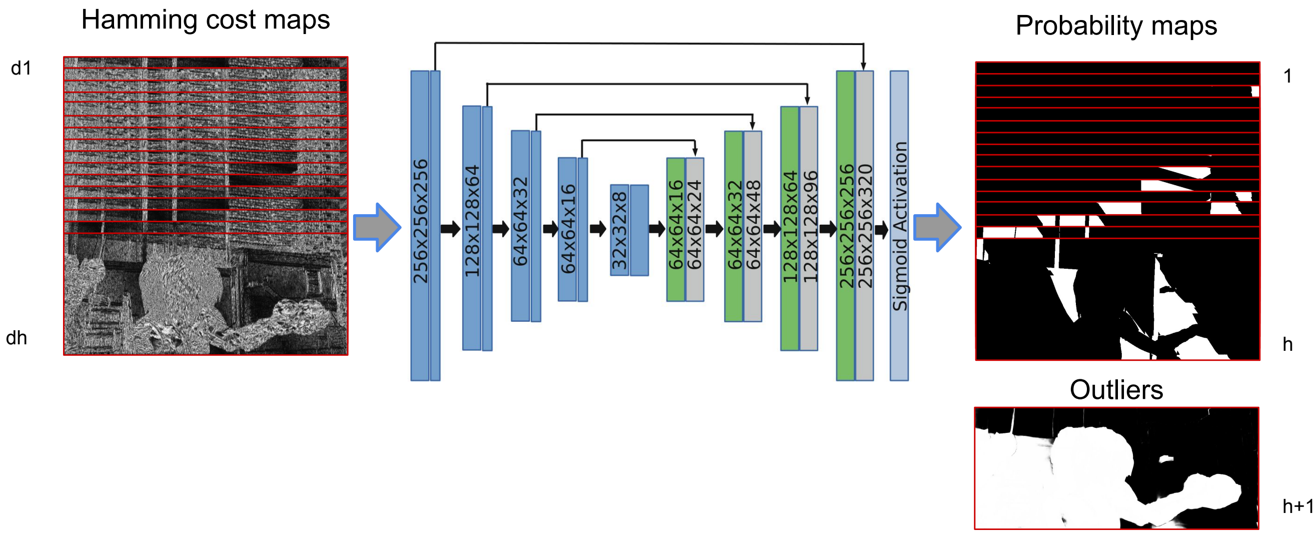
Left image U-Net V-Net Ground–truth

General Block: U-Net U-Net Module

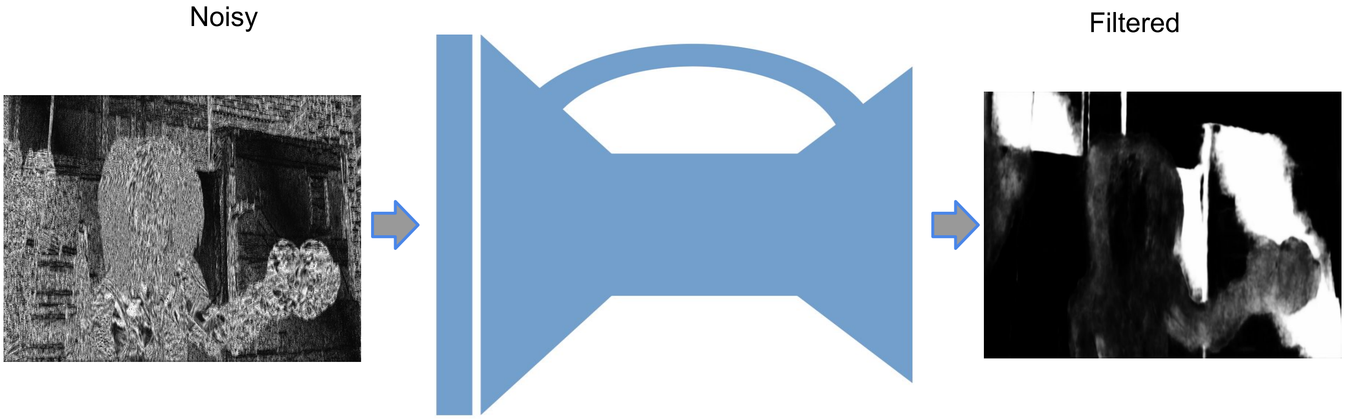
The most basic construction block in our proposal it is a Encoder-Decoder network (U-Net type) where the Disparity Map D is computed with the Hamming distances between the Census transformed patches:



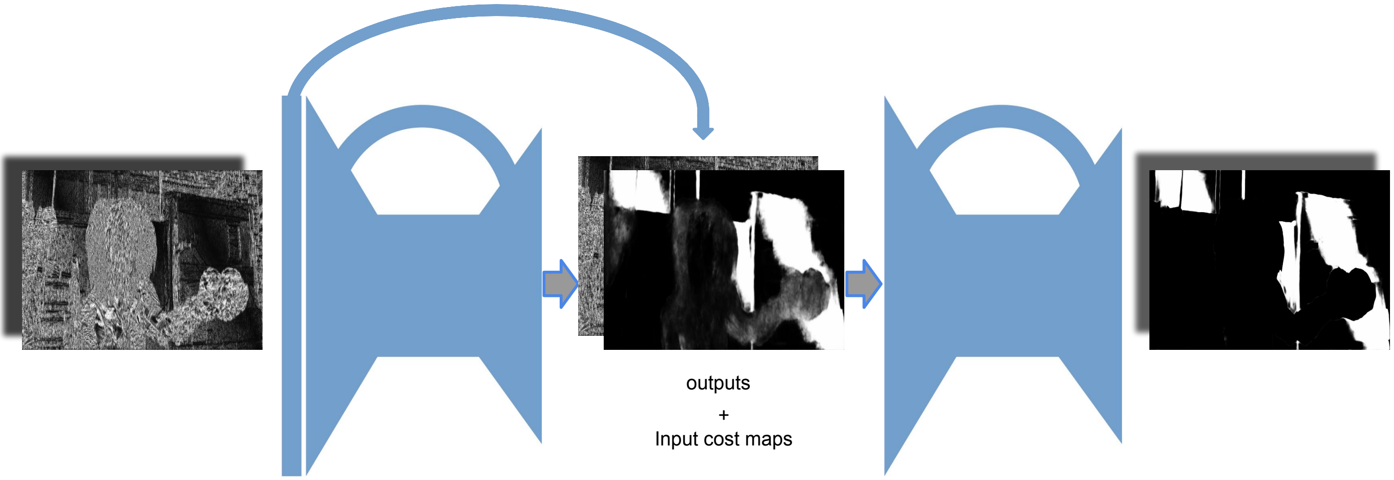
General Block: U-Net U-Net Module



General Block: U-Net U-Net Module



General Block: U-Net U-Net Module



Hamming cost maps

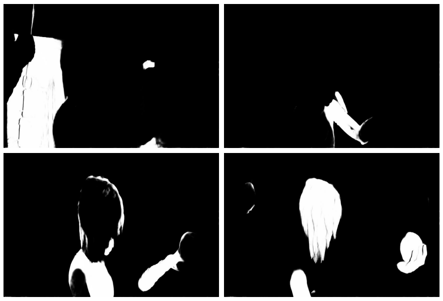
Primary U-Net

Second U-Net

Refined maps

General Block: U-Net U-Net Module

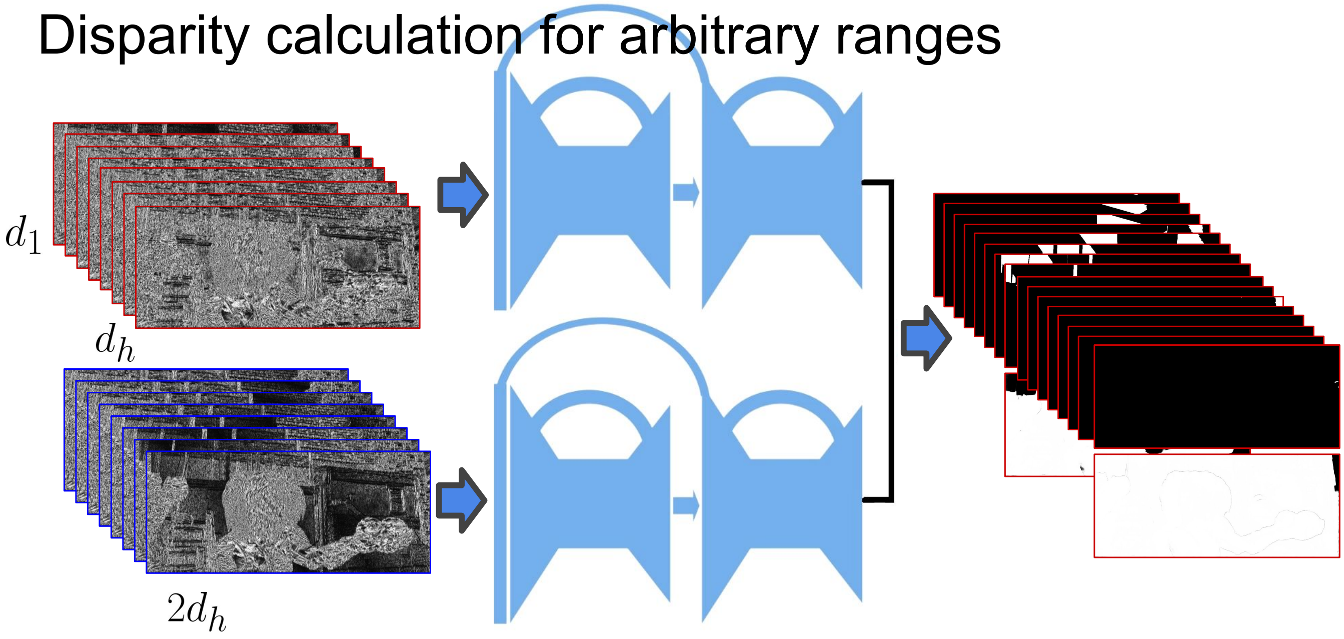
Differently to FlowNet which computes directly a real valued map of disparities, we estimate the probability that a particular candidate disparity δ is the actual one at each pixel.



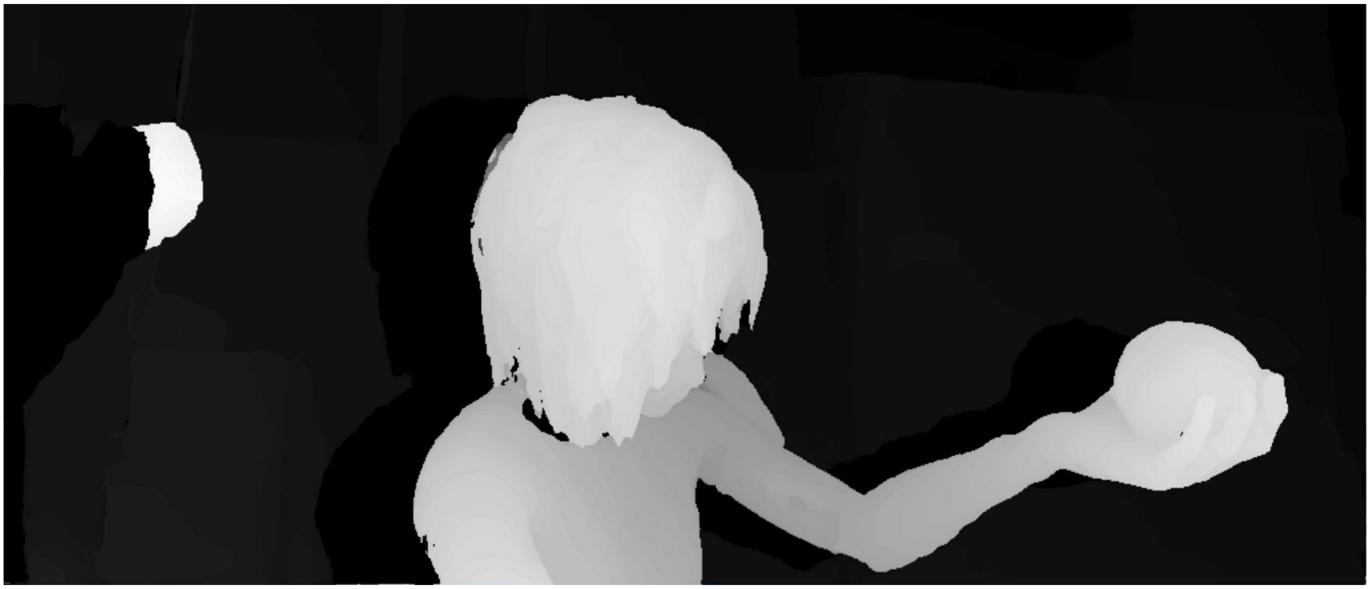
Disparity calculation for arbitrary ranges

Assume, we have a trained basic module for the disparities into the interval [d_1, d_h] and the actual range of disparities, in the stereo pair, lays into the interval [d_1, 2 d_h]. We can reuse our basic model for processing of such a stereo pair if we split the calculations for the disparities sets.

Disparity calculation for arbitrary ranges



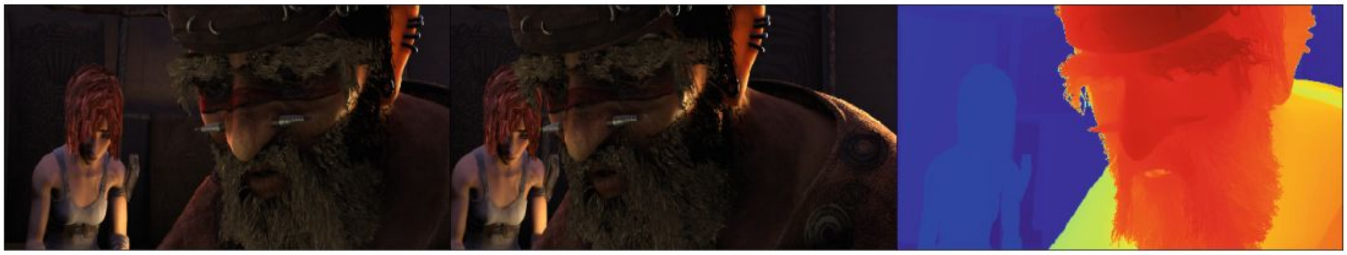
Disparity calculation for arbitrary ranges



Dataset and training

Dataset and Training Parameters

The MPI Sintel-stereo dataset is a benchmark for stereo, produced from the open animated short film Sintel produced by Ton Roosendaal and the Blender Foundation. For our experiments, we use the clean subset pairs that consist of 1064 pairs; 958 for training and 106 for testing.



Pair example with ground truth

Dataset and Training Parameters

We trained our basic block with 16 disparities, during 2000 epochs with mini-batches of size 8.
The ADAM optimization algorithm was used with fixed learning ***lr*** = 0.0001 and ***β*** = [0.9,0.999]

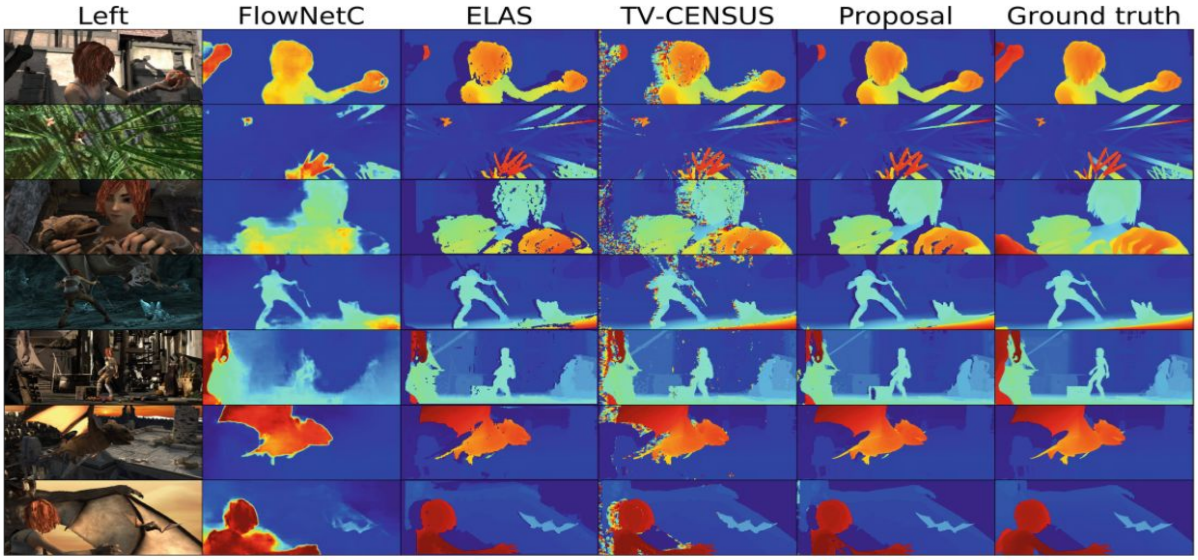
Results

Results

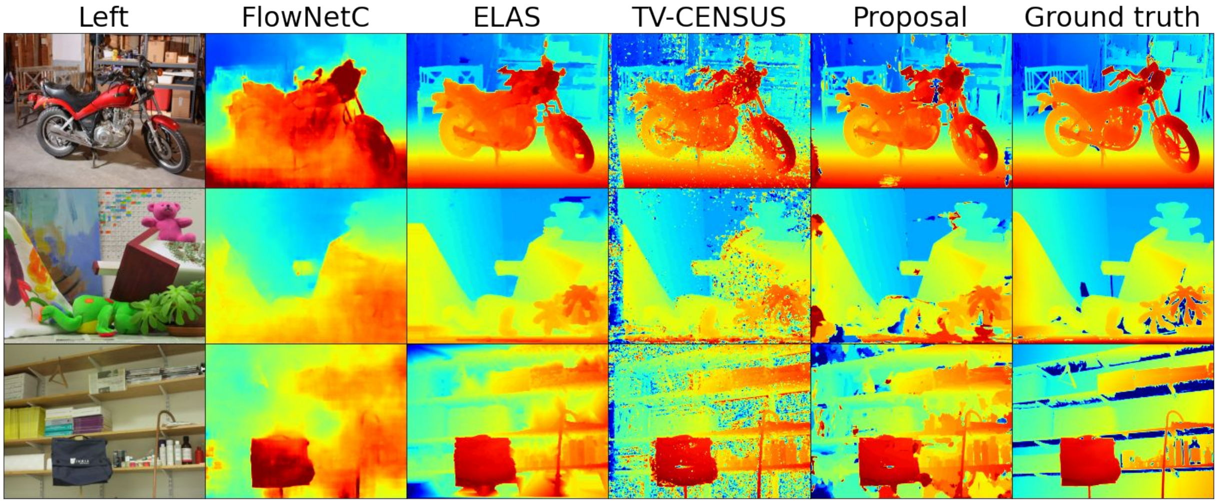
We compare results against FlowNetC and Census-Hamming with total variation denoising (TV-Census). We also compare against Efficient Large-Scale Stereo Matching (ELAS) which builds disparities by forming a triangulation on a set of support points.

Chambolle, A. "An algorithm for total variation minimization and applications" (2004).
Geiger A., Roser M., Urtasun R. Efficient Large-Scale Stereo Matching (2011).

Results

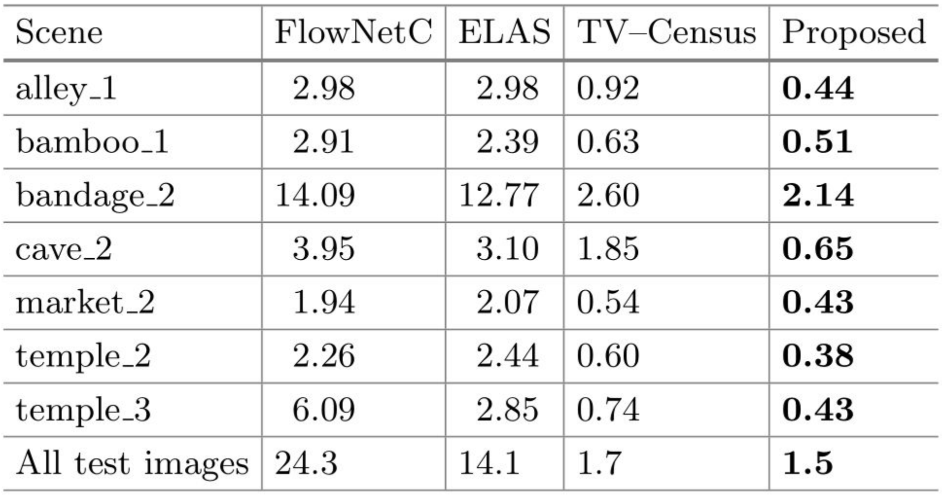


Results



Middlebury Stereo Datasets

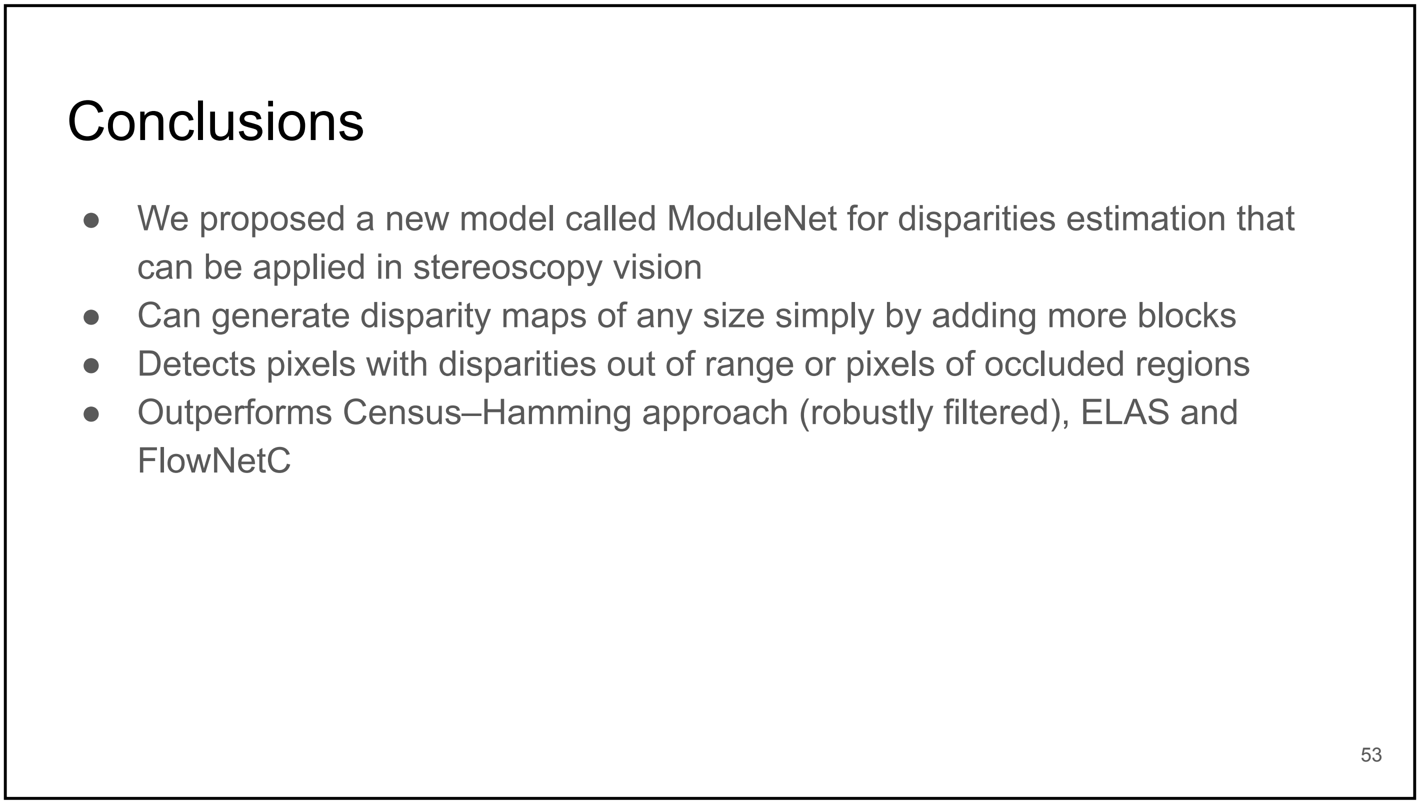
Results



Scene	FlowNetC	ELAS	TV–Census	Proposed
alley_1	2.98	2.98	0.92	**0.44**
bamboo_1	2.91	2.39	0.63	**0.51**
bandage_2	14.09	12.77	2.60	**2.14**
cave_2	3.95	3.10	1.85	**0.65**
market_2	1.94	2.07	0.54	**0.43**
temple_2	2.26	2.44	0.60	**0.38**
temple_3	6.09	2.85	0.74	**0.43**
All test images	24.3	14.1	1.7	**1.5**

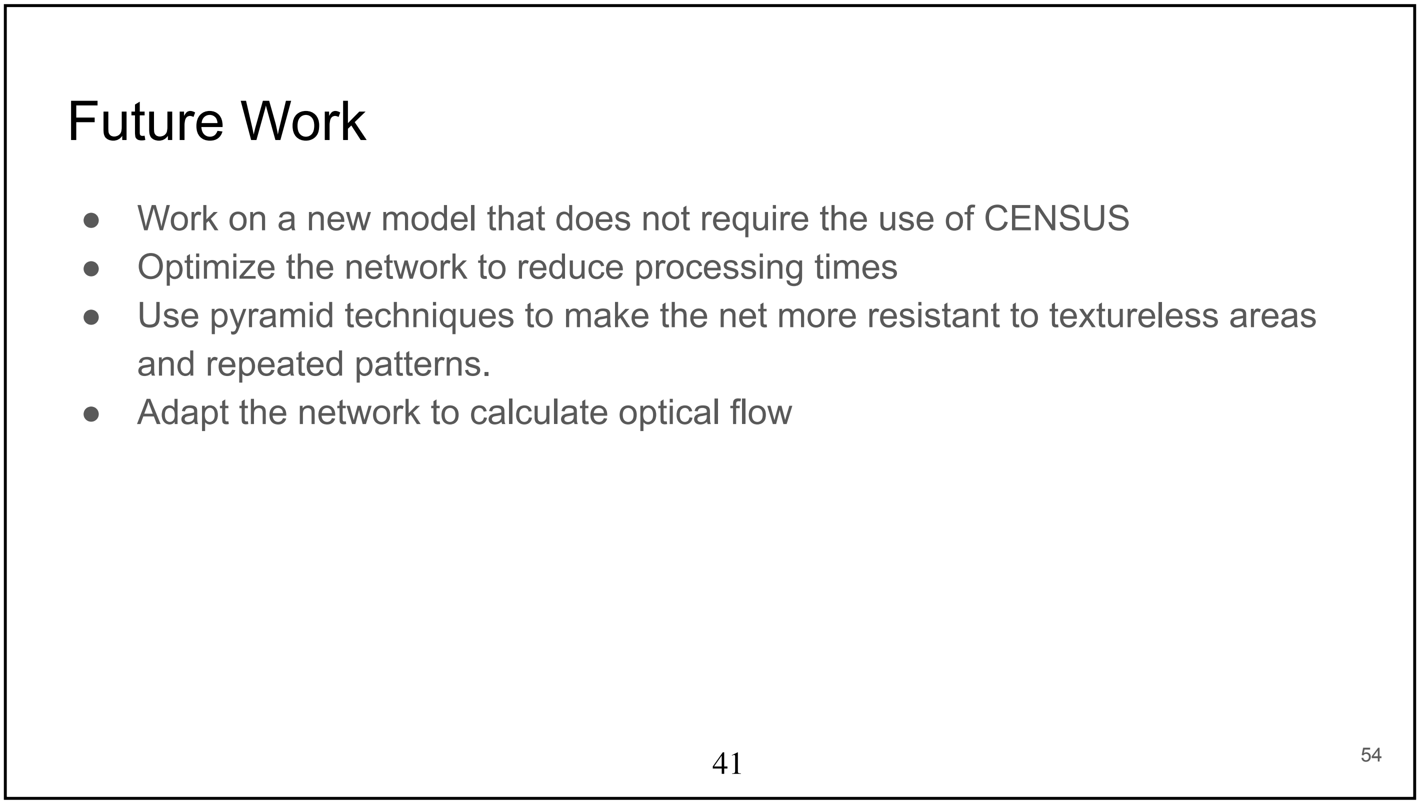
Mean Absolute Error

Conclusions



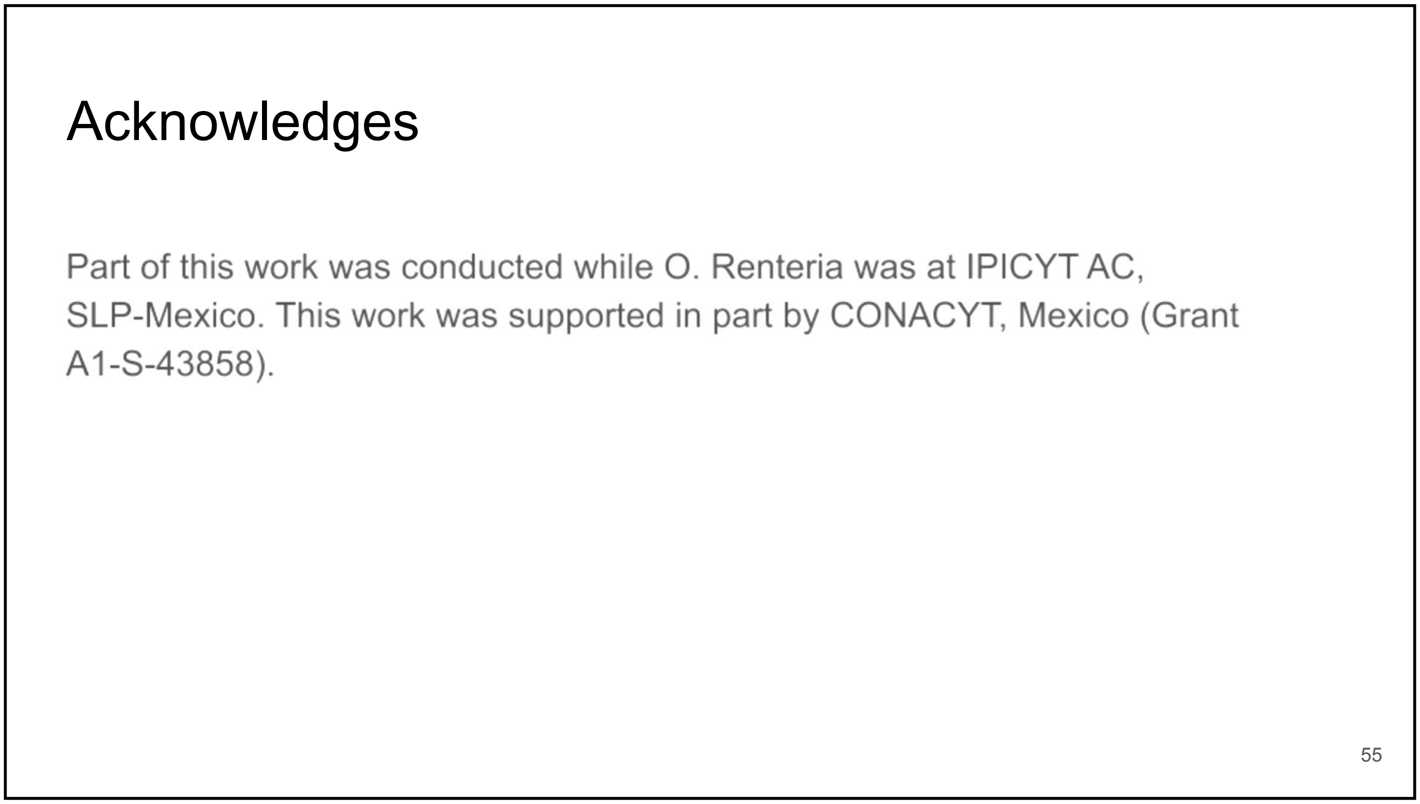
Conclusions

- We proposed a new model called ModuleNet for disparities estimation that can be applied in stereoscopy vision
- Can generate disparity maps of any size simply by adding more blocks
- Detects pixels with disparities out of range or pixels of occluded regions
- Outperforms Census–Hamming approach (robustly filtered), ELAS and FlowNetC



Future Work

- Work on a new model that does not require the use of CENSUS
- Optimize the network to reduce processing times
- Use pyramid techniques to make the net more resistant to textureless areas and repeated patterns.
- Adapt the network to calculate optical flow



Acknowledges

Part of this work was conducted while O. Renteria was at IPICYT AC, SLP-Mexico. This work was supported in part by CONACYT, Mexico (Grant A1-S-43858).

Appendix A

Convolutional Neural Networks

This appendix introduces Convolutional Neural Networks (CNN), U-Net and ModuleNet.

A.1 Introduction

A CNN is a Deep Learning algorithm where an image is the input, then CNN assign importance (learnable weights and biases) to various aspects/objects in the image and is able to differentiate one from the other. The pre-processing required in a CNN is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The design of CNN was natural inspired and clearly based on the seminal work of Hubel and Wiesel [6] that study the presence of receptive fields and functional architecture in the visual cortex of cats. In this work, electrical signals were captured from the brain of cats to study how their visual cortex respond to stimulus, see Fig. A.1. In fact, the study showed that the visual cortexes of cats are composed of areas specialized in responding to particular visual stimulus, the so-called biological receptive fields, see Fig. A.2. The mentioned work showed that each portion of the visual cortex is responsible for responding to specific local regions of the visual fields. In other words, a particular neuron is unaware of what happens to visual signals that are not is its local areas of particular interest

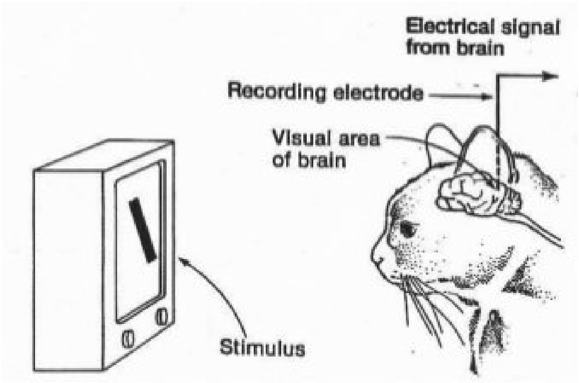


Figure A.1 Experimental setup for study of the visual cortex of cats [6]

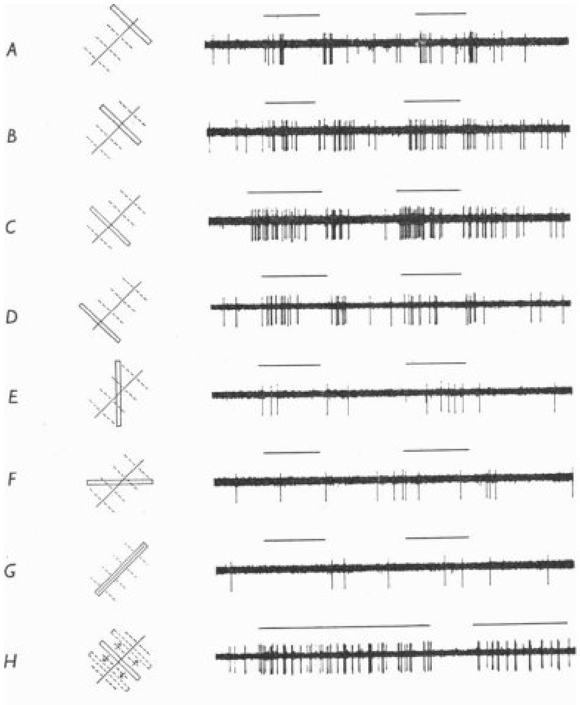


Figure A.2 Demonstration of the existence of receptive fields in visual cortex of cats. Only specific neurons activate each time (right) while the bar moves (left) [6]

One of the first work in neural networks inspired by this papers was the Neocognitron proposed by Kunihiko Fukushima [5]. The Neocognitron (see Fig. A.3) consists of multiple types of cells, the most important of which are called S-cells and C-cells. The local features are extracted by S-cells, and these features' deformation, such as local shifts, are tolerated by C-cells. Local features in the input are integrated gradually and classified in the higher layers.

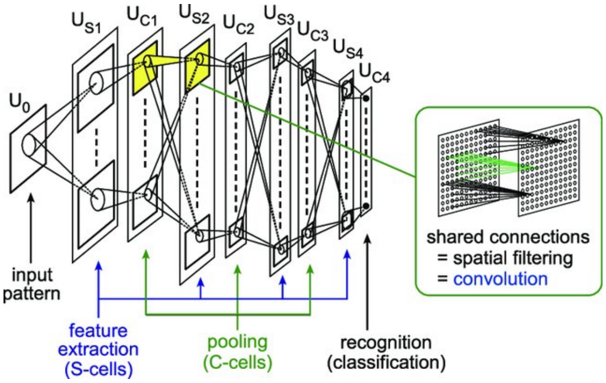


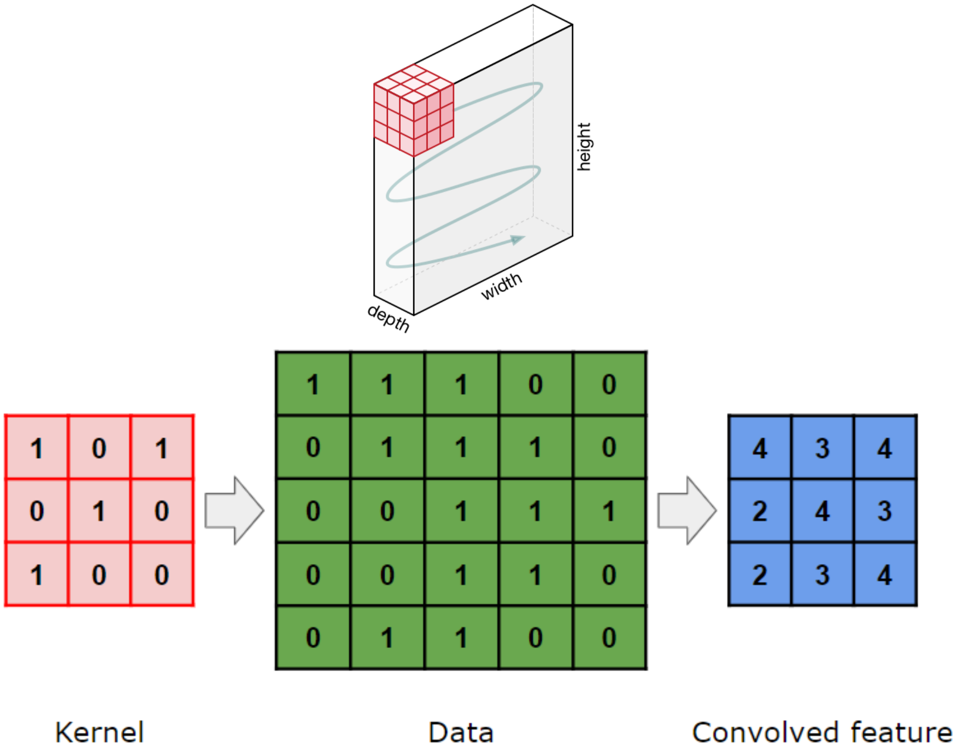
Figure A.3 The architecture of the Neocognitron [5].

The Neocognitron established the two basic layers of any CNN, the convolutional layer (local feature extraction) and the pooling layer (global feature extraction)

A.2 Convolution Layer

The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. CNN need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc.

Convolution is a mathematical operation on two functions (f and g) that produces a third function $f * g$ that expresses how the shape of one is modified by the other. The operation is fairly simple, the function f (named kernel) slides over the function g (data), performing an elementwise multiplication with the part of the input it is currently on, and then summing up the results into a single output, see Fig. A.4.



Kernel

1	0	1
0	1	0
1	0	0

Data

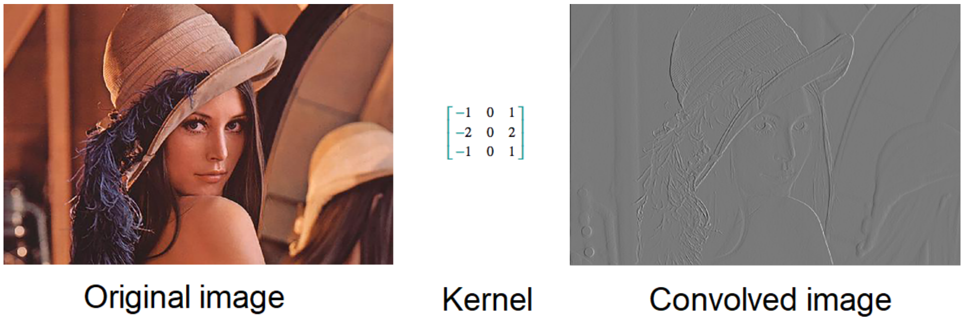
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Convolved feature

4	3	4
2	4	3
2	3	4

Figure A.4 Convoluting a $5 \times 5 \times 1$ image with a $3 \times 3 \times 1$ kernel to get a $3 \times 3 \times 1$ convolved feature

The kernel repeats this process for every location it slides over, converting a 2D matrix of features into yet another 2D matrix of features. The output features are essentially, the weighted sums (with the weights being the values of the kernel itself) of the input features located roughly in the same location of the output pixel on the input layer, see Fig. A.5.



$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Figure A.5 Example of the image Lena [1] with a edge detector kernel

The kernel (also called filter) generates a filtered representation of the input data, the values of the kernel are learned during training and specialize on filter features relevant to the model. This features are locally because the value of a pixel depend of the values of their neighbors, therefore the size of a kernel defines the neighbor window around a pixel.

A.3 Pooling Layer

The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features, which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel, see Fig. A.6. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel.

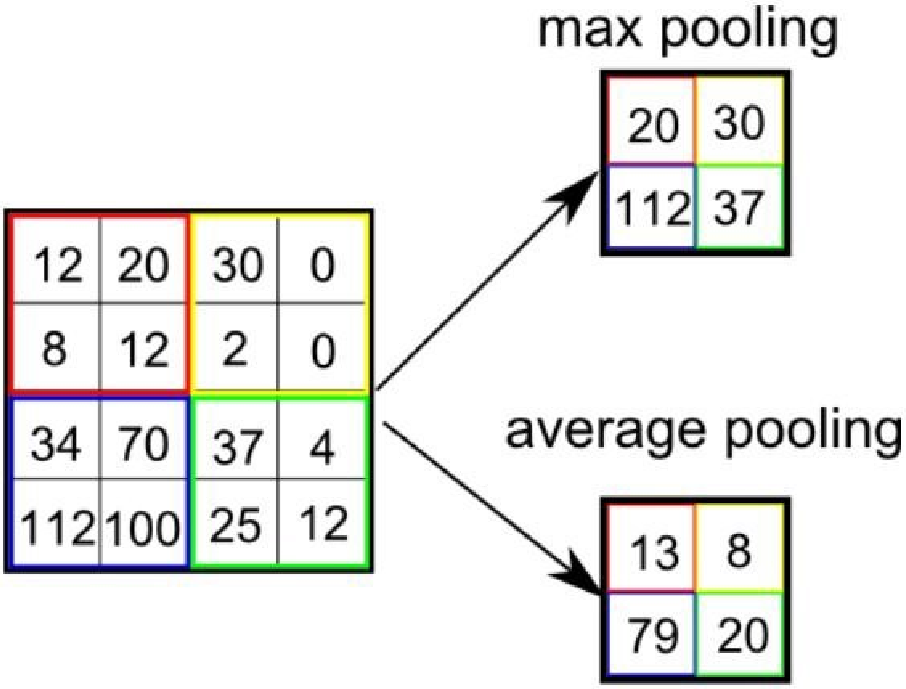


Figure A.6 Types of Pooling

A.4 Conv Block

The Convolutional Layer and the Pooling Layer, together form the *i*-th layer of a Convolutional Neural Network (sometimes called Conv Block). Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, see Fig. A.6, but requiring more computational resources.

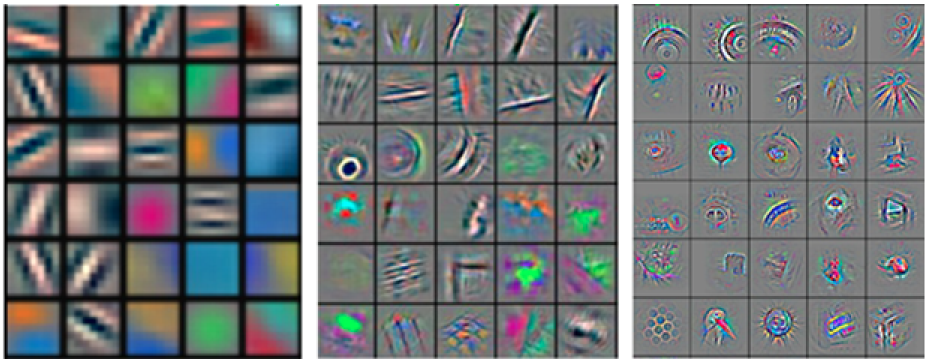


Figure A.7 The combination of more conv blocks derives on more complex features [7]

Now we have a network which is capable of learn how to filter an image in a a way relevant to solve a problem. The output can be used directly for tasks as segmentation or add a fully connected layer (Multi-Layer Perceptron) for classification tasks, see Fig. A.8.

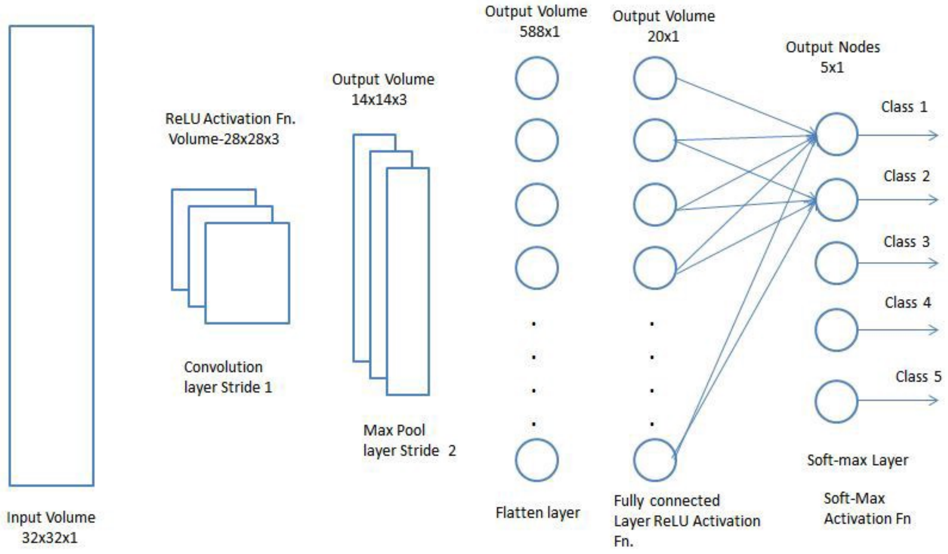


Figure A.8 Minimal model of a CNN for multi-classification [2]

A.5 U-Net

Over the years, many CNN architectures have been proposed to improve the image classification problem, as well as novel proposals to solve other types of vision tasks, such as segmentation.

In Image Segmentation, the machine has to partition the image into different segments, each of them representing a different entity, see Fig. A.9.

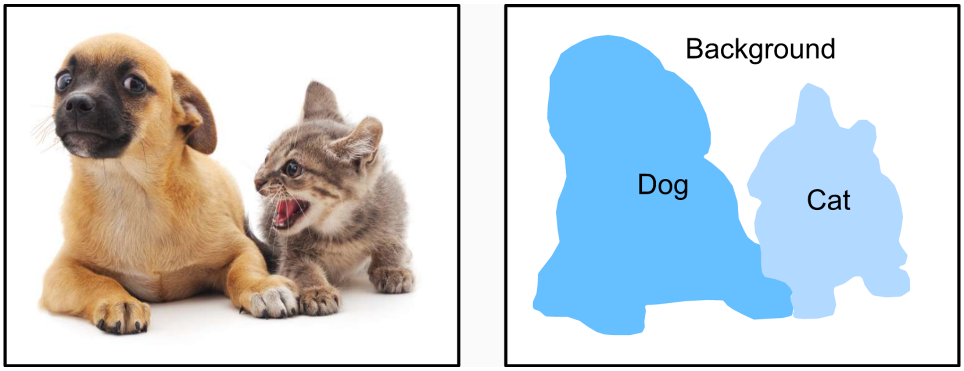


Figure A.9 Semantically-segmented image, with areas labeled “dog”, “cat” and “background” [4]

Image segmentation is useful in many fields from self-driving cars to medical imaging. CNN gave decent results in easier image segmentation problems but it has not made any good progress on complex ones until the apparition of U-Net.

U-Net was designed especially for medical image segmentation, were the subtleties in those images are quite complex and sometimes even challenging for trained physicians. This architecture showed such good results that it used in many other fields.

A.5.1 The idea behind U-Net

The purpose of a CNN is to learn the feature mapping of an image and exploit it to make more nuanced feature mapping. This works well in classification problems as the image is converted into a vector (flatten layer) which used further for classification, see Fig. A.8. But in image segmentation, we not only need to convert feature map into a vector but also reconstruct an image from this vector. This is a very difficult task because it is a lot tougher to convert a vector into an image than vice versa. The whole idea of U-Net is revolved around this problem.

While converting an image into a vector, we already learned the feature mapping of the image so why not use the same mapping to convert it again to image. This is the recipe behind U-Net. Use the same feature maps that are used for contraction to expand a vector to a segmented image. This would preserve the structural integrity of the image which would reduce distortion.

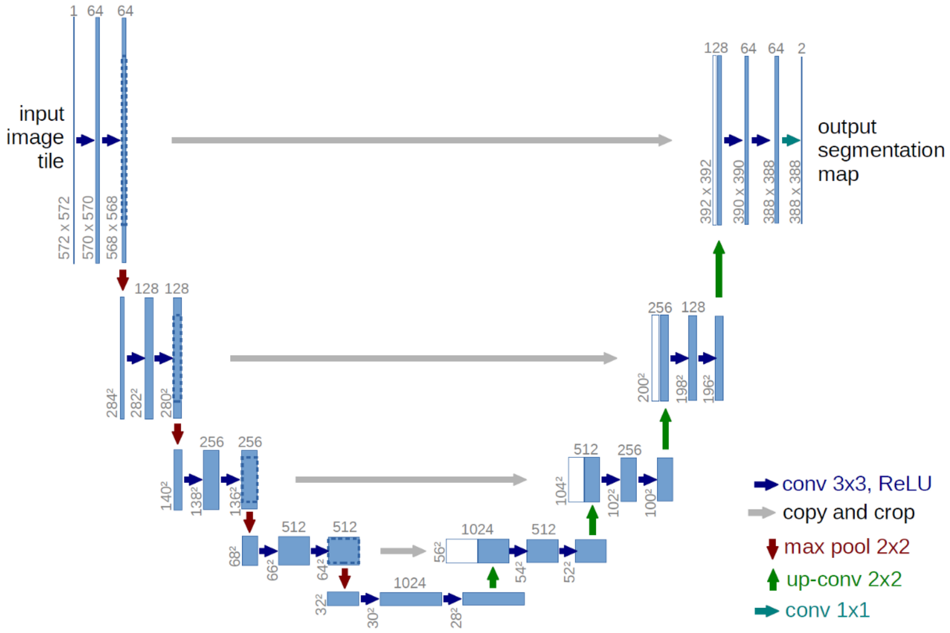


Figure A.10 U-Net architecture as shown in the original paper [9]

The architecture visually looks like a “U” which justifies the name. This architecture consists of three sections: **the contraction**, **the bottleneck**, and **the expansion path**. The contraction path is made of many contraction blocks. Each block takes an input and applies two 3 convolution layers followed by a 2×2 max pooling. The number of kernels or feature maps after each block doubles so that architecture can learn the complex structures effectively. The bottommost layer mediates between the contraction layer and the expansion layer. It uses two 3×3 CNN layers followed by 2×2 up convolution layer.

Similar to contraction layer, the expansion layer it also consists of several expansion blocks. Each block passes the input to two 3×3 CNN layers followed by a 2×2 upsampling layer. Also after each block number of feature maps used by convolutional layer get half to maintain symmetry. However, every time the input is also get appended by feature maps of the corresponding contraction layer. This action would ensure that the features that are learned while contracting the image will be used to reconstruct it. The number of expansion blocks is as same as the number of contraction block. After that, the resultant mapping passes through another 3×3 CNN layer with the number of feature maps equal to the number of segments desired.

A.5.2 Pixel-wise Classification

A pixel-wise softmax is applied on the resultant image which is followed by cross-entropy loss function. So each pixel is classified into one of the classes. The idea is that even in segmentation every pixel have to lie in some category and we just need to make sure that they do. So we just converted a segmentation problem into a multiclass classification, see Fig. A.11.

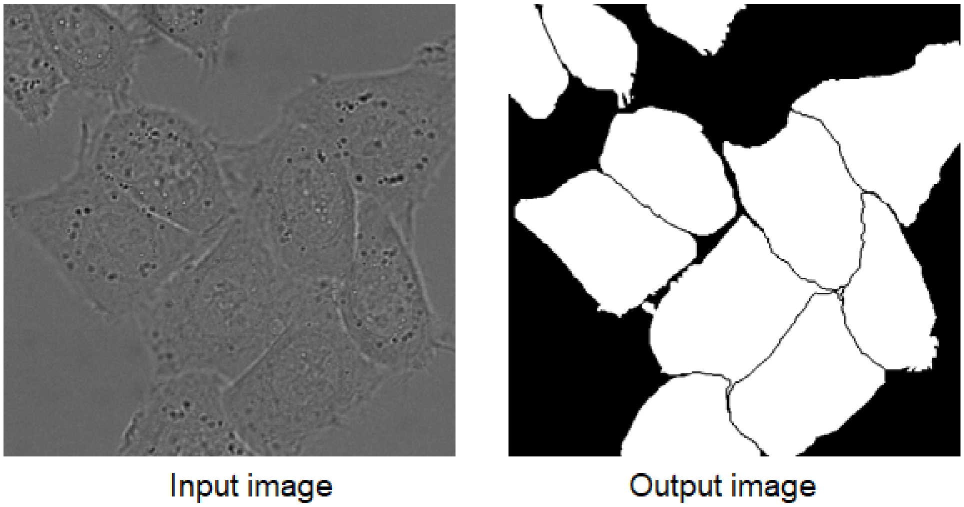


Figure A.11 Example of the prediction of the U-Net (note the borders in the segments) [9]

A.6 ModuleNet

In a stereo system we have two images, each one has a view of the same scene but each view is slightly moved (horizontally). Given the perspective, in order to pair both images it requires different levels of displacement, in each displacement only some regions will match.

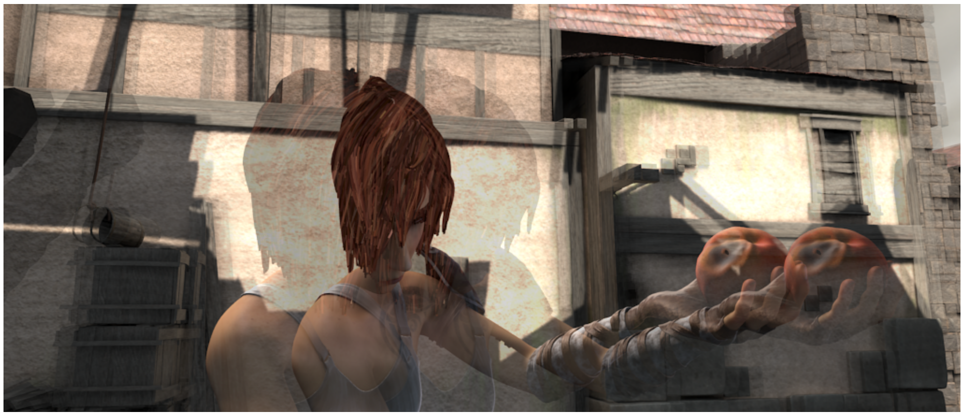
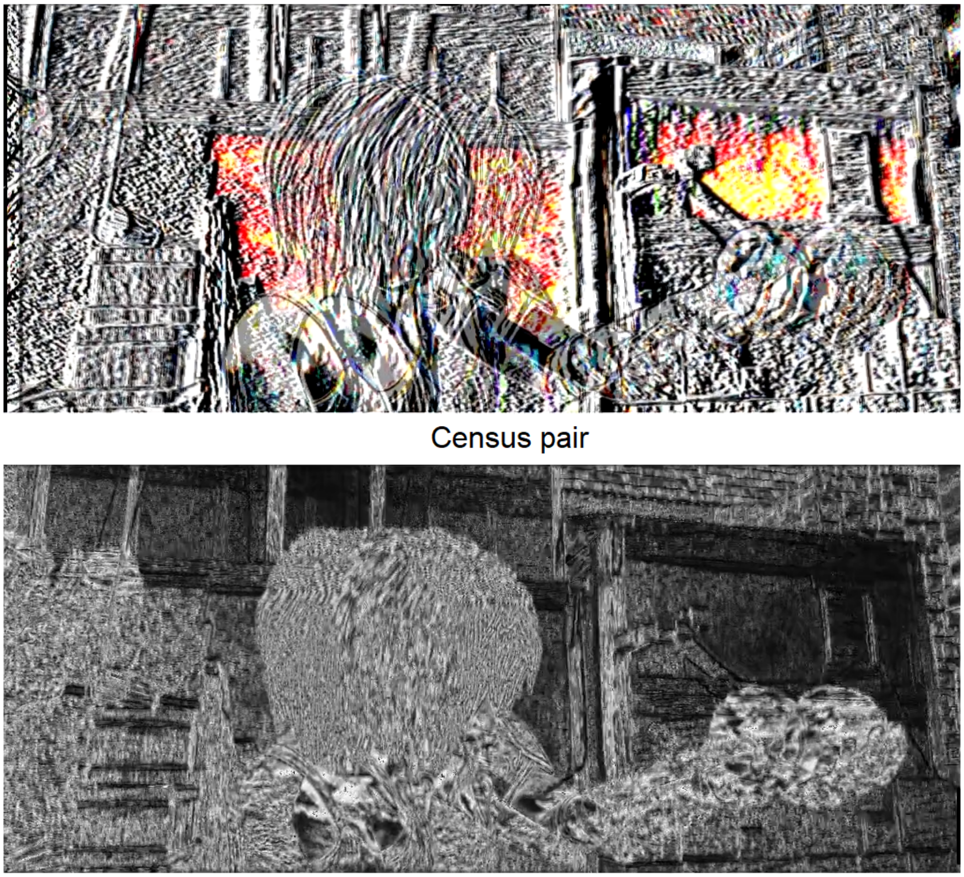


Figure A.12 Example of a binded stereo pair, at different displacements, different regions will match [3]

The level of displacement required to match a region in both images defines how far or close it is to the camera.

There are an infinity of algorithms to measure matching regions between images, in our work we choose the Census tansform combined with the Hamming cost, this algorithm gives acceptable accuracy and requires less computation power. The principal problem it is generates noisy maps, see Fig. A.13.



Census pair

Hamming map

Figure A.13 The Hamming cost produces a map in which the lower the cost (black areas), bigger the match between pixels

A.6.1 Filtering noise with ModuleNet

The basic objective of the ModuleNet model is to filter noisy Hamming maps in a smart way. ModuleNet it is composed of two main parts, the **input generator** and the **U-Net U-Net Module**, the input generator it is composed by two custom layers called Census Layer and Hamming layer, the first layer takes the two images (left and right) and applies them the census transform. the resulted census images enters into the Hamming Layer in which the Hamming cost is calculated. ModuleNet is capable of generate *N* filtered maps (in the paper, we defined 16 maps) so the model requires *N* input Hamming maps, the Hamming Layer generates such maps moving the right census map one pixel at a time to the right, then calculates the Hamming cost between the left static and right moved census images. This process repeats *N* times in order to generate the *N* maps required. The Census and Hamming layers are fixed operations, whereby they do not require training.

The U-Net U-Net Module is composed of two U-Net connected in series, see Fig. A.14, the first U-Net generates a segmented version of the input maps. The network learns how to maximize the high correlation zones, meanwhile ignoring the noisy ones in areas with low correlation, inherent to the Census and Hamming cost.

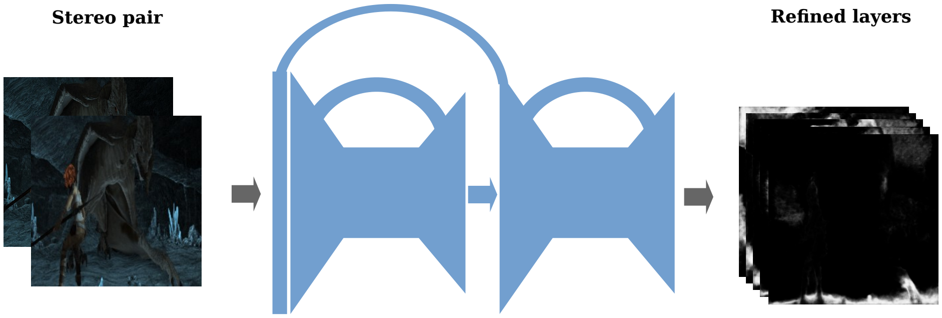
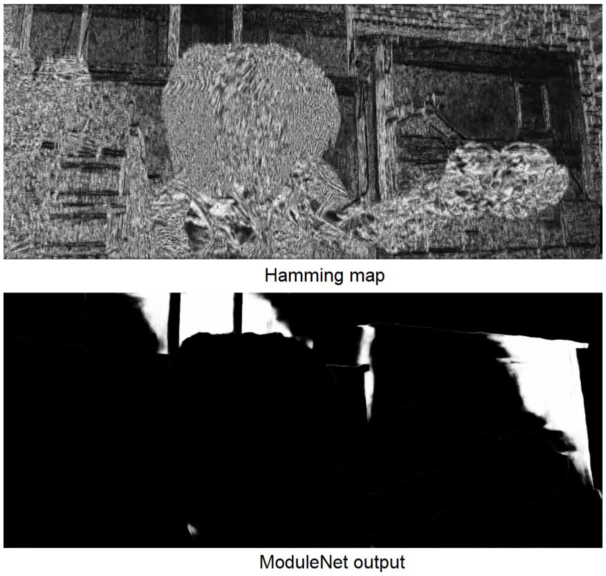


Figure A.14 The main block of the ModuleNet [8]

The second U-Net model refines the outputs, generating high quality segmentations, see Fig. A.15.



Hamming map

ModuleNet output

Figure A.15 The Hamming cost produces a map in which the lower the cost (black areas), bigger the match between pixels [8]

ModuleNet only can generate a fixed amount of disparities maps at a time (16 maps), in some small images is enough, but in bigger images this fixed range is insufficient to cover all the displacements.

To circumnavigate this limitation, we can run several ModuleNet modules in parallel, see Fig. A.16, each modules will take care of process a different range of disparities. Lets say our stereo pair has a disparities range of $R = 64$, and our ModuleNet only can see $N = 16$ disparities, we require R/N modules in parallel in order to cover the entire range (in this example, we need 4 models).

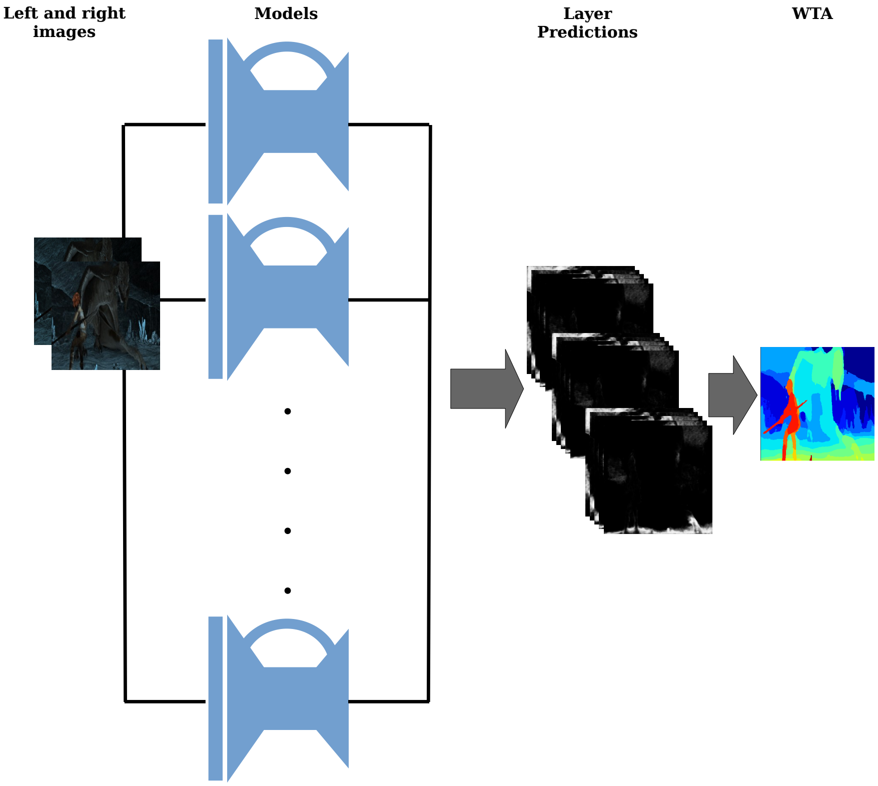


Figure A.16 We can stack several ModuleNet for extended disparities ranges [8]

This is accomplished by moving the right image N pixels to the right before feed it to the network, in the example of 4 modules, the four modules receives the same left image, but the first module receives the right image moved $N * 0$ pixels, the second module the right image moved $N * 1$, the third module the right image moved $N * 2$, etc. At first glance this seems like a waste of memory because, apparently you need to load the module several times in order to cover the needed range. In reality you run only one module, but it is executed in batches. The final part is only stack all the output maps and apply a simple WTA (Winner Takes All) to generate the final Disparities map, see Fig A.17.

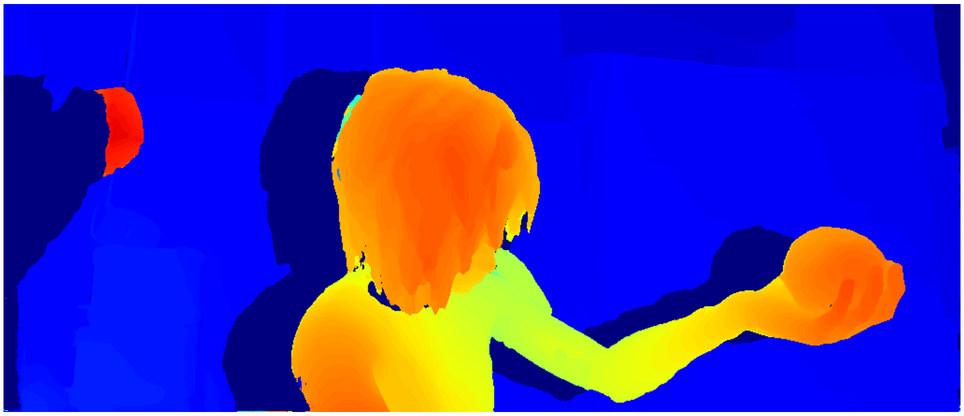


Figure A.17 Final result of WTA colored with Jet palette [8]

References

[1] Image processing: the lena story. https://en.wikipedia.org/wiki/Lenna. Accessed: 2020-12-04.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012.

[5] Kunihiko Fukushima and Sei Miyake. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, 15(6):455 – 469, 1982.

[6] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591, 1959.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.

[8] O.I. Renteria-Vidales, J.C. Cuevas-Tello, A. Reyes-Figueroa, and M Rivera. Modulenet: A convolutional neural network for stereo visio. In Figueroa Mora K. et al. (Eds.), editor, *Pattern Recognition. MCPR 2020. Lecture Notes in Computer Science*, volume 12088, pages 219–228. Springer, 2020.

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.