TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

AUSTRIAN INSTITUTE OF TECHNOLOGY

Deep Learning for Stereo Vision – Guest lecture

LVA Stereo Vision (188.513)
Summer Semester 2024
Andreas Kriegler

Modalities

- Contacts:
  - andreas.kriegler@tuwien.ac.at
  - margrit.gelautz@tuwien.ac.at
- Slides will be available in the TUWEL course
- Machine learning builds on decades of mathematical frameworks – here it is packed into ~45 mins

If you want to help in keeping the number of errors at a reasonable level send any that you find to andreas.kriegler@tuwien.ac.at

Literature

- ML & DL use applied statistics, linear algebra & calculus:
  - Mathematical foundations and many common algorithms of machine learning – **the ML "bible":** [1]
  - The application of deep learning in neural networks: [2], [3]
  - Artificial intelligence in general and multi-agent theory: [4]
- Lectures:
  - The problems of high dimensional learning – lecture 2 of the AMMI 2021 GDL100 course: [5]
  - TU Wien - 194.100 Theoretical Foundations and Research Topics in Machine Learning [6]
  - Stanford - CS229 Machine Learning [7]
  - **Stanford - CS231n** Convolutional Neural Networks for Visual Recognition [8]
  - MIT - Deep Learning and Artificial Intelligence Lectures [9]
- Videos:
  - **Deep Learning Series, 3Blue1Brown** [10]
  - Mathematics for Machine Learning, Ulrike von Luxburg [11]

Topics covered

I. Introduction to machine learning (ML), deep learning (DL) and Convolutional neural networks (CNN) ~45mins

II. Deep learning for stereo vision ~5mins

III. Recent deep stereo vision methods ~35mins

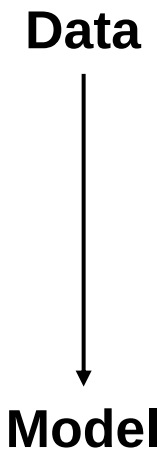
IV. If time allows it: detour to transformers ~5mins

Theory

Machine learning basics

- In classical programming we define rules for input → output relations

- In machine learning (ML) we use data to
  - 1. Generate our model (**learning**)
  - 2. Apply it to new observations (**inference**/prediction)



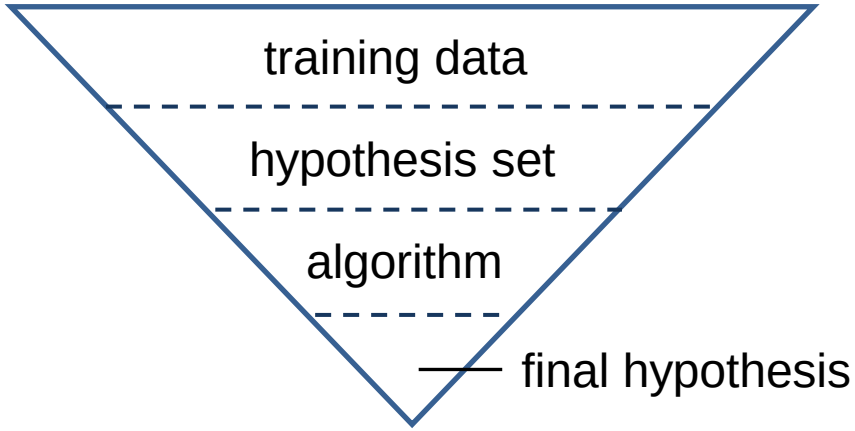


Figure recreated from [12]

Introduction machine learning

- ML often based on calculating the **gradient** of a (convex) **loss-function**
- „No magic“ in ML/DL: find **minima** of objective/cost/error/target/loss function
- Try to find the **weights (parameters)** of the model that minimizes the cost

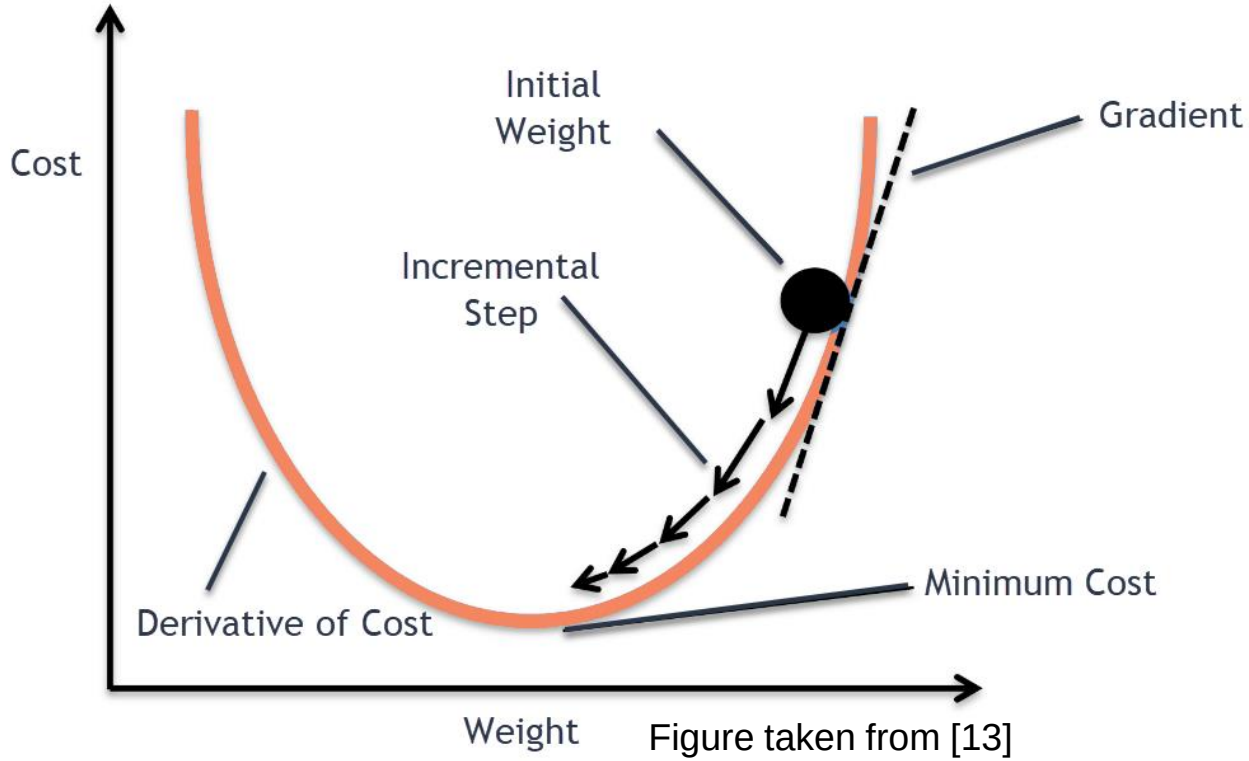


Figure taken from [13]

ML basics – supervised classification

ML example transcribed from [1]

- We have: a **training set** of N observations of x, $\boldsymbol{x} := (x_1, \dots, x_N)^\mathsf{T}$ with corresponding **target** observations t, $\boldsymbol{t} := (t_1, \dots, t_N)^\mathsf{T}$
- We want: predict $\hat{t}$ for new $\hat{x}$, using parameters $\boldsymbol{\theta}$
- **Regression** if $\boldsymbol{t} \subset \mathbb{R}$ or n-way **classification** if $\boldsymbol{t}$ is categorical $t \in \{0, \dots, n-1\}$

- We can, for example, try to fit a polynomial curve

$$f = y(x, \boldsymbol{\theta}) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_M x^M = \sum_{j=0}^{M} \theta_j x^j$$

- We can calculate any loss/error function often L1 or L2

$$L1(\boldsymbol{\theta}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \boldsymbol{\theta}) - t_n\} \qquad L2(\boldsymbol{\theta}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \boldsymbol{\theta}) - t_n\}^2$$

ML basics – supervised classification

ML example transcribed from [1]

$$L(\boldsymbol{\theta}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{\theta}) - t_n\}^2$$

- This will **overfit**, i.e. perform well on training but poorly on test samples
- To prevent this we add a term (λ) for **regularization**

$$\tilde{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{\theta}) - t_n\}^2 + \frac{\lambda}{2}||\boldsymbol{\theta}||^2$$

- Now we can update – **optimize** – our parameters $\boldsymbol{\theta}$ with e.g. curve fitting methods

This was a simple example to introduce common terminology - in typical ML problems we want a convex loss function so we can use **gradient descent** techniques [14]

In ML the hat $\hat{x}$ is typically used to denote targets and the snake $\tilde{x}$ for estimates of some variable x
Also, convex functions tend to arise naturally when dealing with data from an exponential probability distribution family [73]

Deep learning – artificial neural networks

- Deep learning with multilayer perceptrons (MLP) since 1965 [15]
- Dealing with vanishing gradients since 1991 [16, 17], deep since 2012 [18]
- Now models with up to 10^{12} parameters trained on cloud-based tensor or graphical processing unit (TPU/GPU) clusters [19]

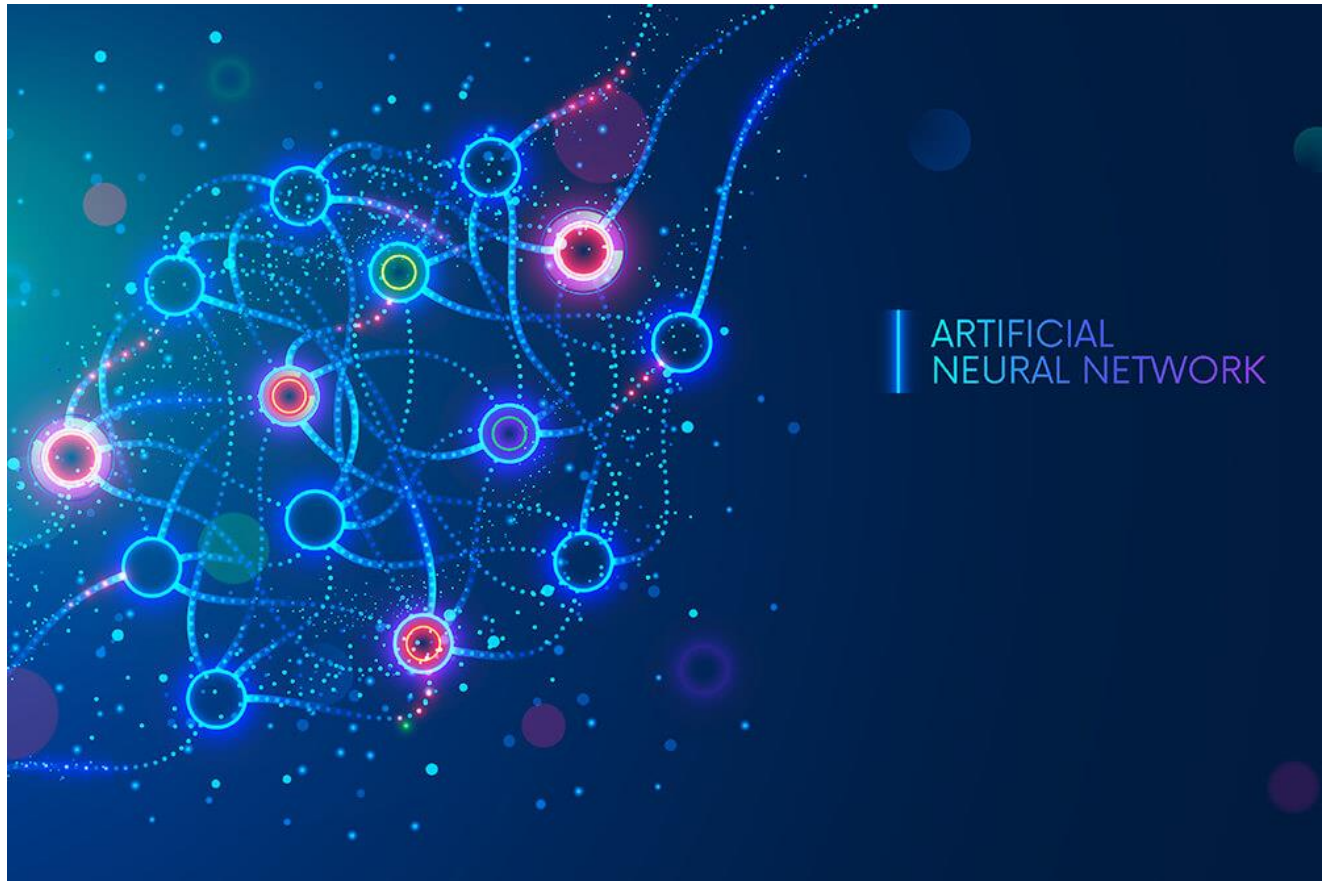


Figure taken from [20]

Deep learning – 1-hidden layer neural network

- Activations f in hidden neurons
  - Sigmoid (old): $\phi(z) = \frac{1}{1+e^{-z}}$
  - **ReLU** (rectified linear unit): $\mathrm{ReLU(z)} = \max(0, \mathrm{z})$
- **Weights** w scale the function and **bias** b shifts it
- N, M: number of input neurons – here 5 and 4
- Multiple output neurons form a output vector

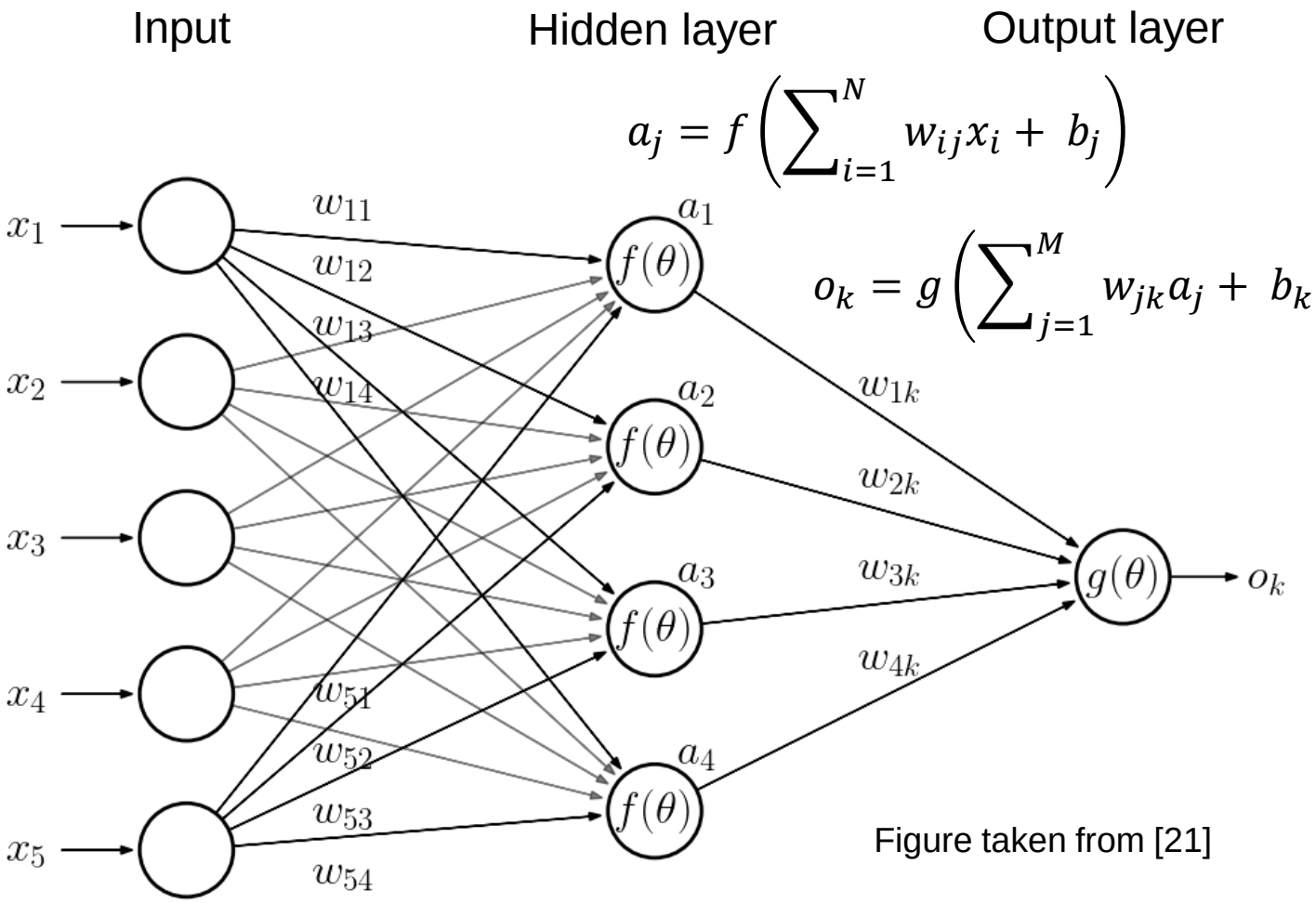
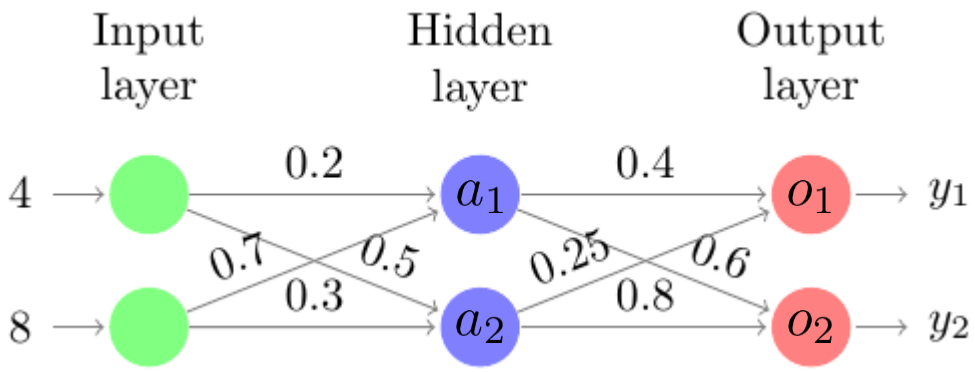


Figure taken from [21]

NN forward-pass example

- Given the input, weights, active bias neurons in the hidden and output layer and standard sigmoid(x) activation function and softmax(x) squashing, what is the output y_1 and y_2



- Prediction would be the label associated to y_2

$$a_1 = sig(\sum_{i=1}^{2}(w_{i1}x_i) + 1) \approx 0.838$$

$$a_2 = sig(\sum_{i=1}^{2}(w_{i2}x_i) + 1) \approx 0.808$$

$$o_1 = \sum_{j=1}^{2}(w_{j1}a_j) + 1 \approx 1.537$$

$$o_2 = \sum_{j=1}^{2}(w_{j2}a_j) + 1 = 2.15$$

$$y_1 = softmax(o_1) = \frac{e^{o_1}}{\sum_{k=1}^{2} e^{o_k}} = 0.351$$

$$y_2 = softmax(o_2) = \frac{e^{o_2}}{\sum_{k=1}^{2} e^{o_k}} = 0.649$$

Training NNs – obtaining the loss

- The output vector of the NN is squashed to predictions $\hat{y} \in (0, 1]$ using **softmax**

$$\sigma(\boldsymbol{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

- This „distribution over categorials“ allows probabilistic reasoning [74]
- We apply **cross-entropy loss** (simplified for NNs) using $\hat{\boldsymbol{y}}$ and ground truth $\boldsymbol{y}$

$$L = -\boldsymbol{y} \cdot \log(\hat{\boldsymbol{y}})$$

- NNs are typically updated using a variation of gradient descent: stochastic gradient descent (**SGD**)

Training NNs – gradient descent

- Stochastic gradient descent (**SGD**)

$$\boldsymbol{\theta} \coloneqq \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \tilde{L}(\boldsymbol{x}; \boldsymbol{t}) \quad \eta \ \ldots \text{ learning rate}$$

- Instead of the entire dataset, we only compute the gradient for small sub-sets
- Configuring η (as well as any other model hyperparameters) is its own science
- Backpropagation as a special case of (reverse) automatic differentiation [22] – based on Euler-LaGrange equations [23]

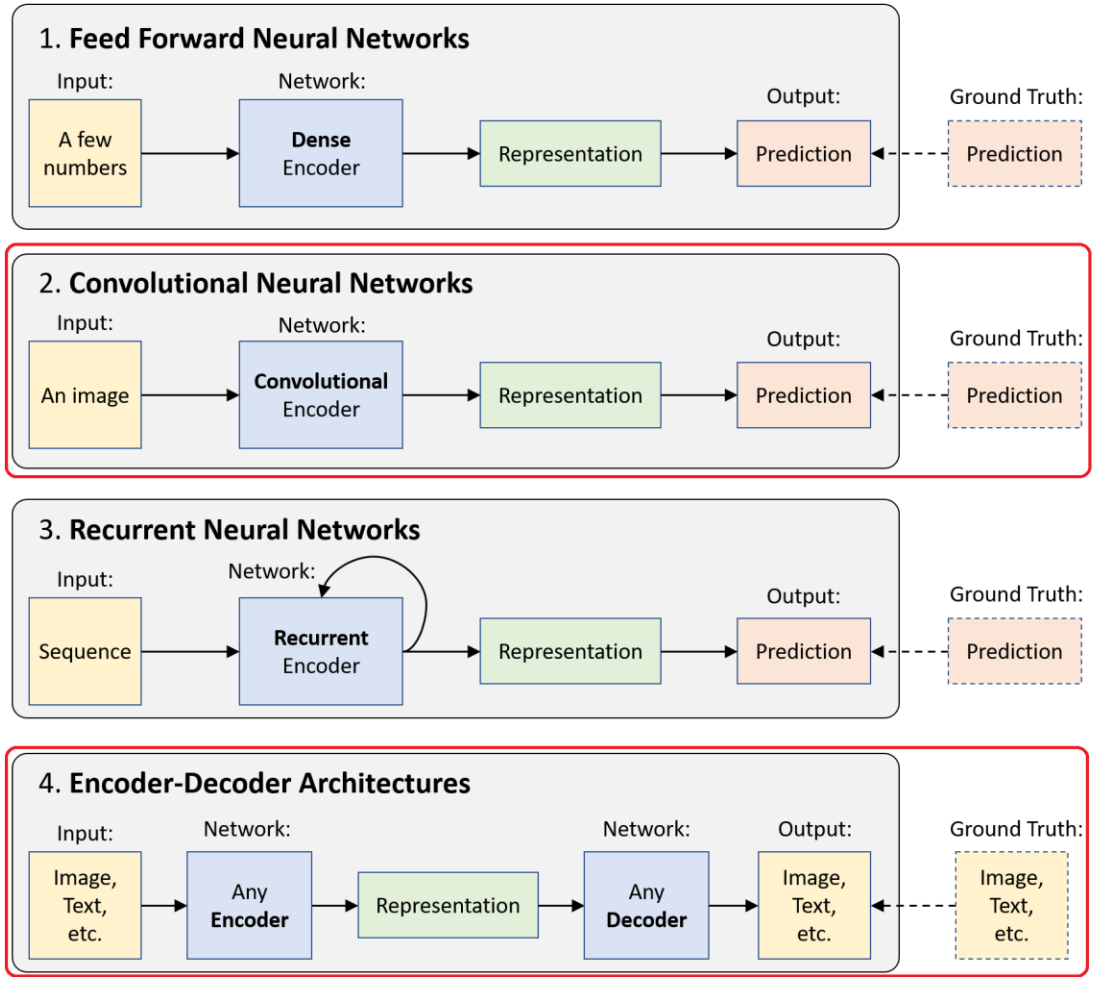
Training NNs – backpropagation via SGD

03:07 – 11:26 of part 3 of 4 from 3Blue1Brown‘s deep learning series [10]

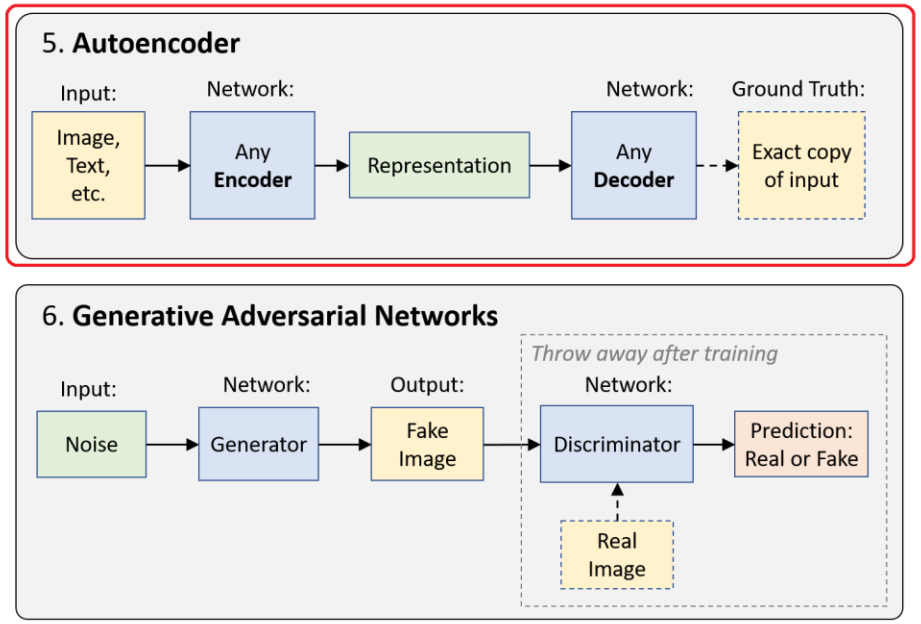
https://www.youtube.com/watch?v=aircAruvnKk

Learning families and neural networks

Supervised Learning



Unsupervised Learning



Reinforcement Learning

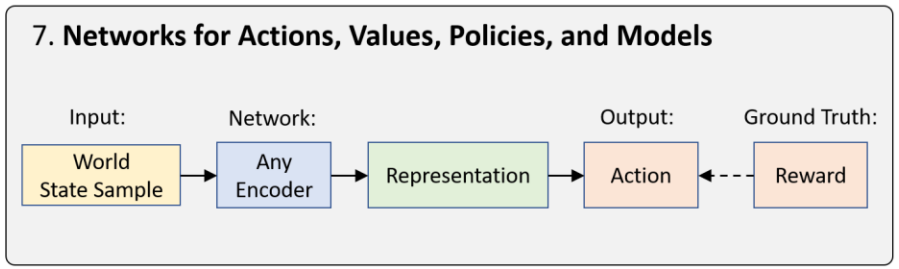


Figure slightly modified and taken from [9]

Supervised Convolutional Neural Networks (CNNs)

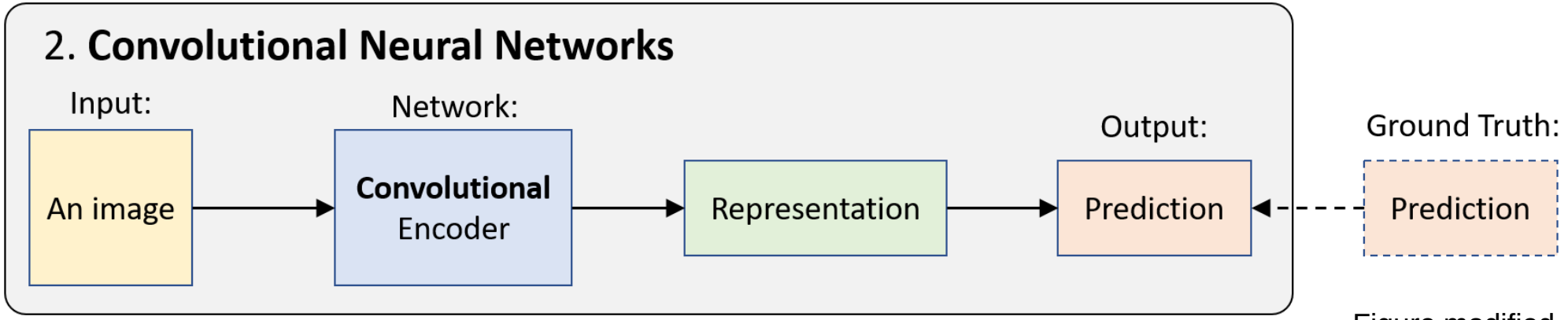
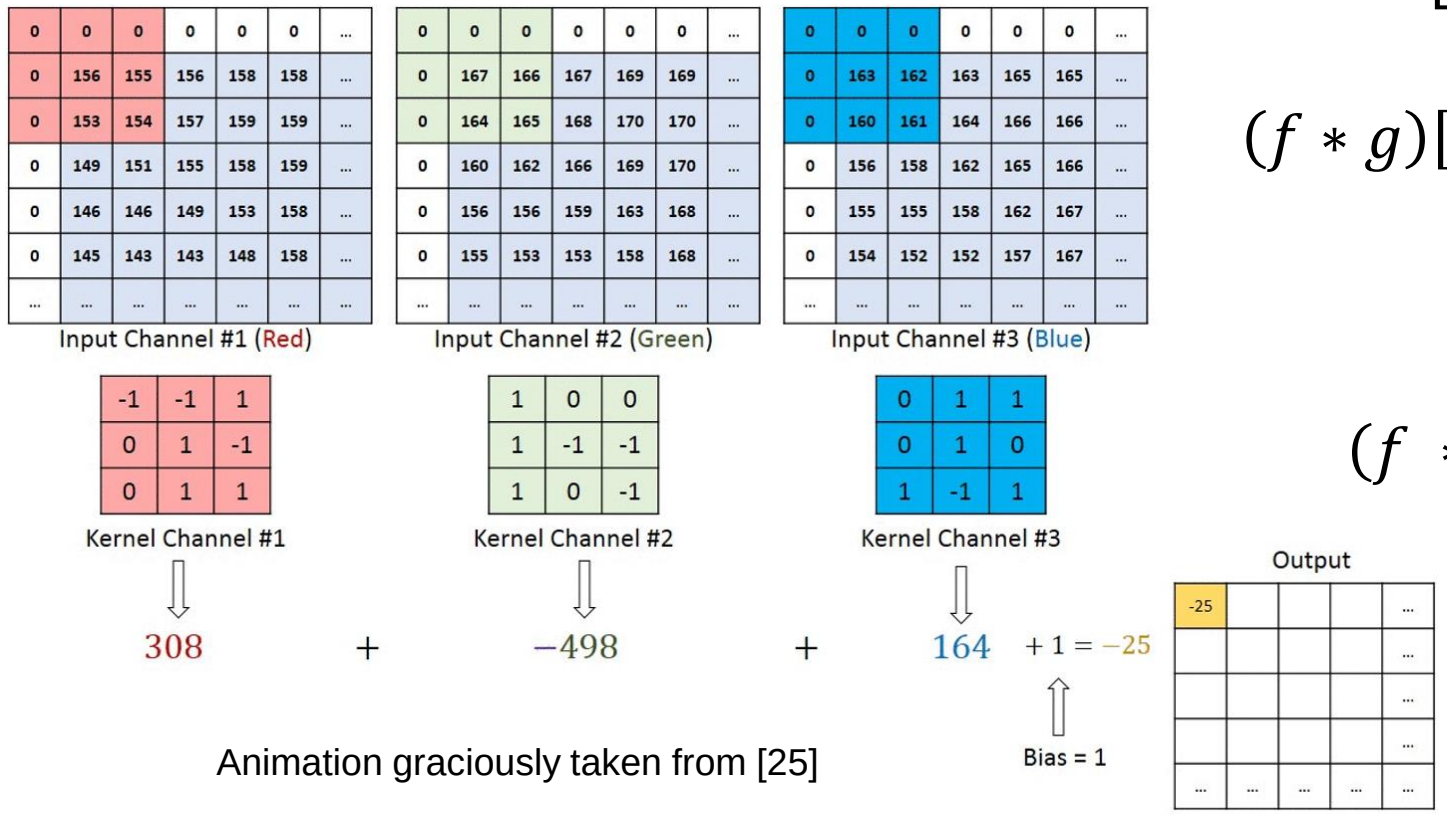


Figure modified and taken from [9]

1. We operate on images, particularly the **RGB** channels
2. The **convolutional layers** act as **encoders** for feature representations
3. From those representations we obtain predictions
4. We compare predictions with the **ground truth (gt)** via a loss function
5. Backpropagate the loss using gradient descent
6. Update our weights, i.e. **kernel entries**

Convolution operation in CNNs

- Invented in 1979 [24] – see [8] for the Stanford course
- We learn kernel entries using backpropagation



Animation graciously taken from [25]

Discrete convolution

$$(f * g)[n] = \sum_{m=-M}^{M} f[n-m]g[m]$$

Easily differentiable

$$(f * g)' = f' * g = f * g'$$

Discrete convolution example

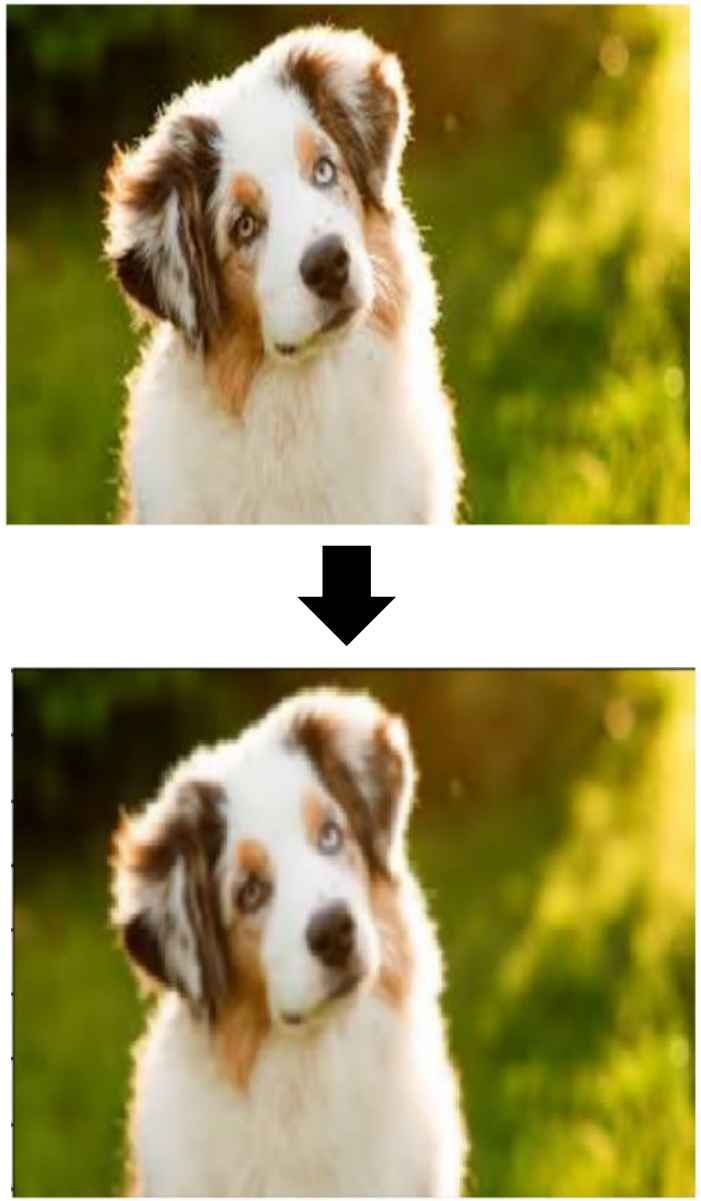
- Given below input patch and the kernel, what is the result of a discrete convolution?
- What would this filter do to an entire image?

128	190	125
170	130	140
187	220	150

Input

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Kernel



- 1/9 * (128 + 190 + 125 + 170 + 130 + 140 + 187 + 220 + 150) = **160**
- It would **blur** an image, since it takes the average

Images taken from [73]

Typical CNN architecture

- Convolutional layers, activations, pooling etc. → feature-maps
- Fully connected (FC) layer followed by softmax
- Obtain a score [0, 1] at every neuron → maximum-a-posteriori rule for classification
- Differentiable cross-entropy loss → we can use gradient descent

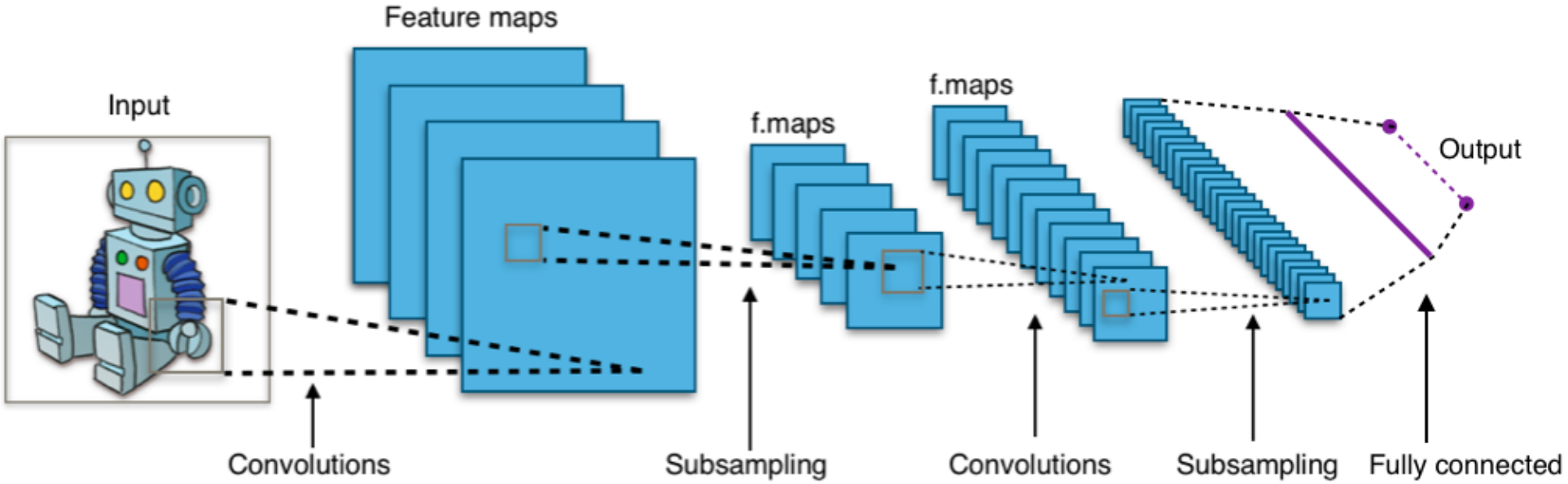


Figure taken from [26]

CNNs as powerful feature extractors

- Augmented RGB images
- The features become increasingly complex
- Visualizations on the right [27] are „projected activations of selected feature maps“
- No heatmaps but highlights of contributing features

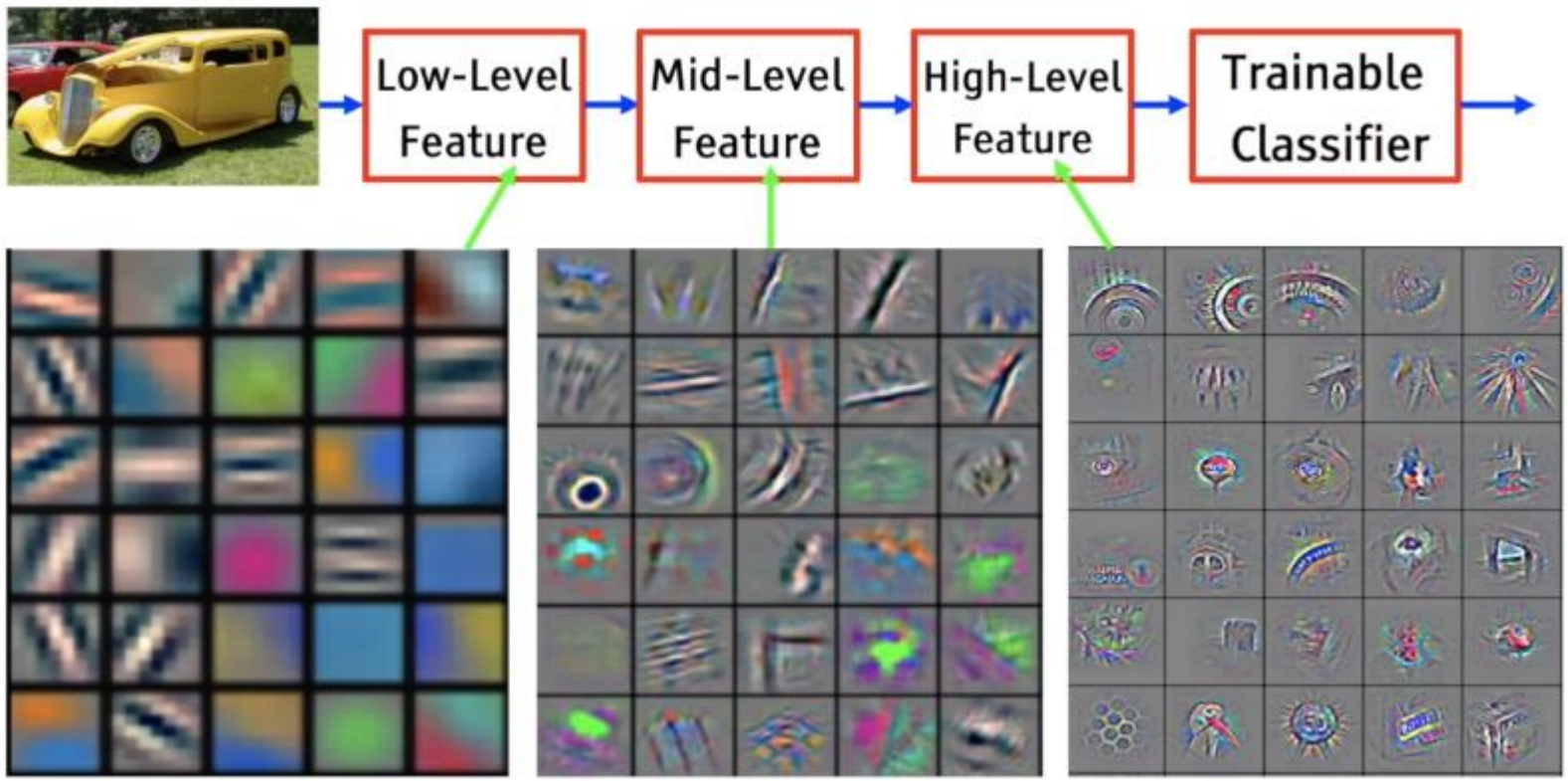
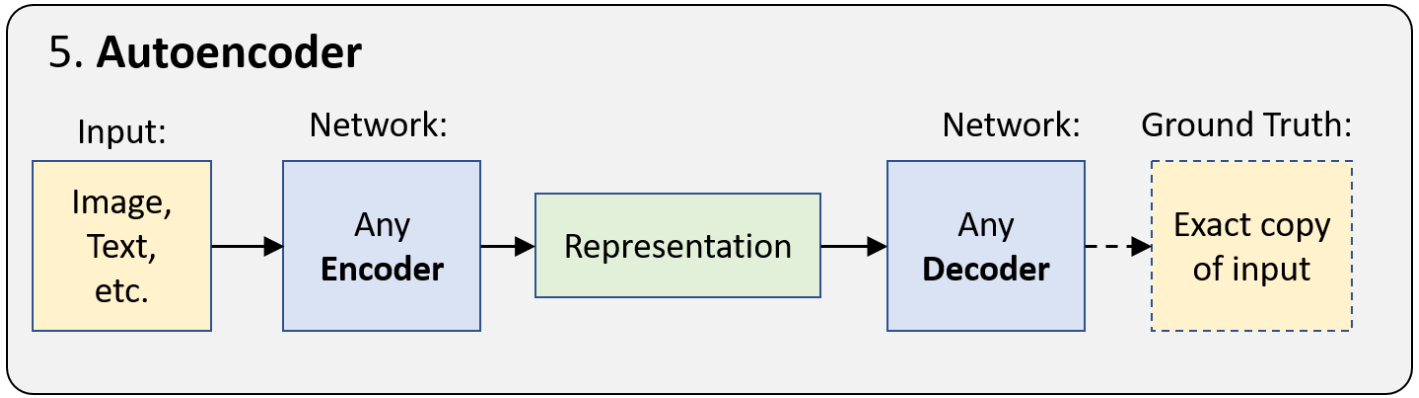
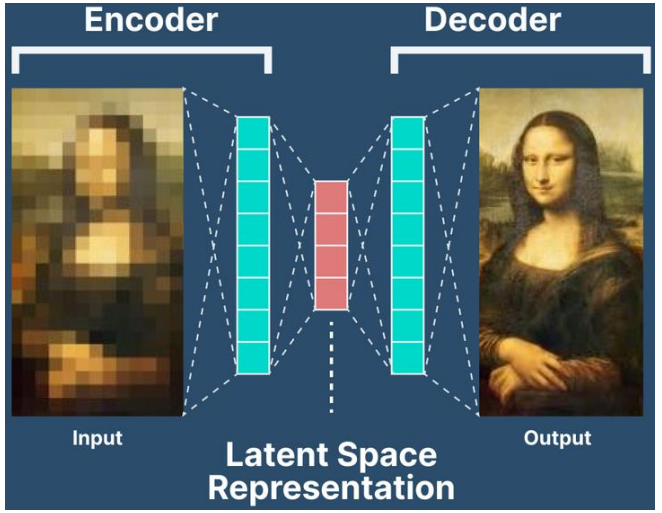


Figure taken from [27]

Autoencoder

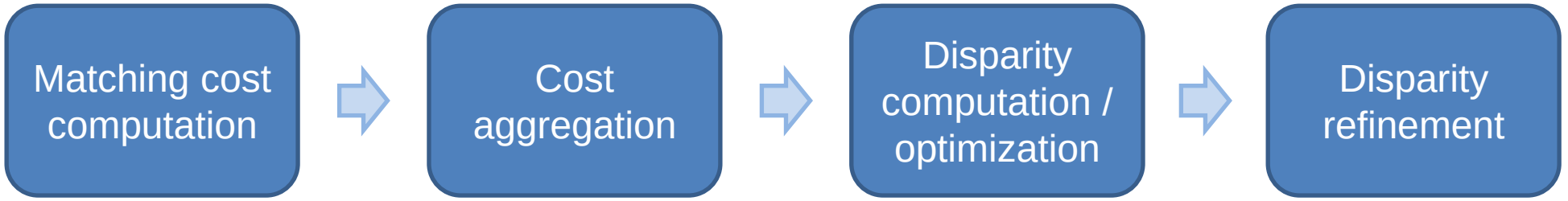
→ Figure modified and taken from [28]
↓ Figure modified and taken from [9]





1. **Encoder-Decoder** architectures use encoded, learned representations to decode to a high-dimensional output (not prediction) that can be of different modality
2. Autoencoders are **unsupervised** models trying to **recreate the input exactly**
3. A highly meaningful yet **low-dimensional representation** is learned
4. Its ground truth is the input, so **no human annotation** is required
5. Applications use these unsupervised embeddings in **representation learning**

4 steps of stereo vision



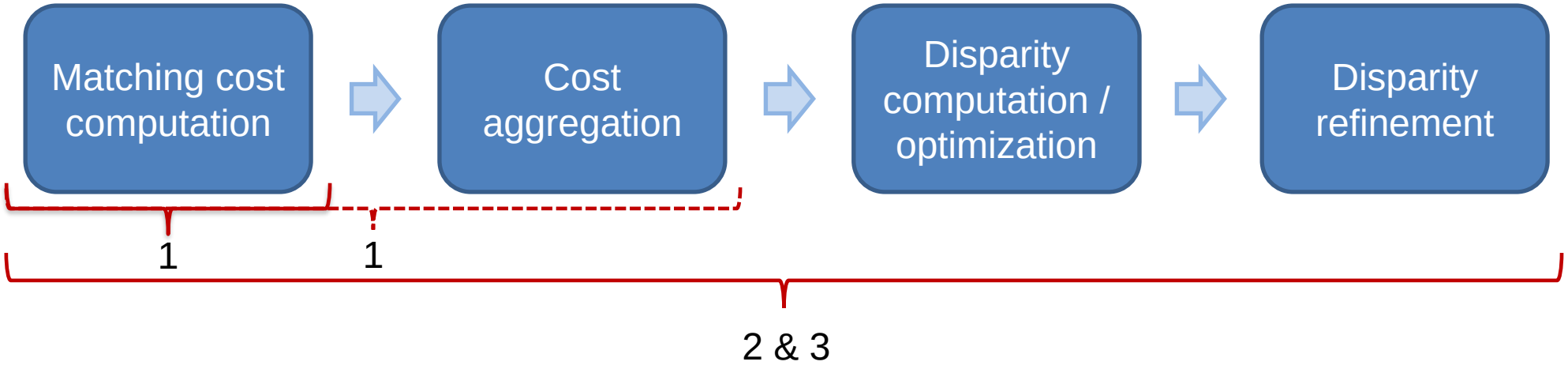
4-step stereo vision methodology from [30] – figure inspiration from [31]

1. Matching cost as difference between pixel intensity (classical) or image patches (CNNs)
2. Cost aggregation via window-based summation, global cost, cross-based or filters
3. Winner-takes-all, i.e. lowest cost for all possible disparity map values
4. Slanted plane smoothing, left-right-consistency or gaussian filters for smoothing

- CNNs replace one or more of these components

CNNs as replacements

Figure inspiration from [31] – below 3-level taxonomy following [32]



1) **Non end-to-end**: MC-CNN [33], improved with multiscale-fusion [34] or multilabel distribution [35]
   Uses feature maps to construct matching costs using e.g. dot product or fully-connected -> sigmoid

2) **End-to-end**: all steps jointly optimized, using a) **2D encoder-decoder** [36] or b) **3D convolutions** [37]
   Methods for optical flow and stereo matching but stereo search space limited due to epipolar constraint

3) **Unsupervised**: rely on minimizing photometric warp error – difference between true and warped image
   DeepStereo generates new views using nearby images [38] – [39] include pixel-wise reconstruction loss

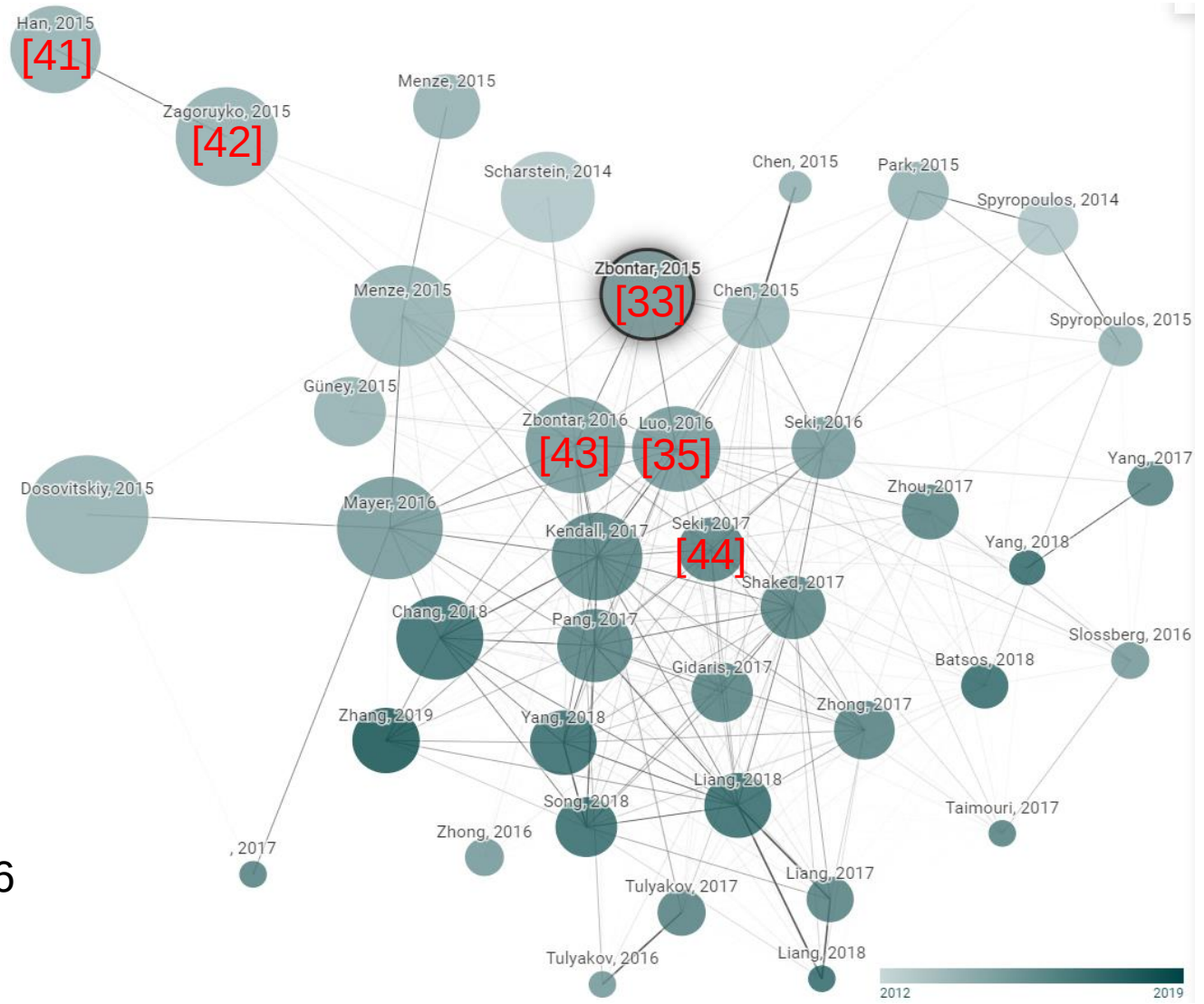
Non-end-to-end deep stereo matching methods

General similarity models

[41]: MatchNet – Han, 2015
[42]: ZagoruykoNet – Zagoruyko, 2015

Stereo matching and disparity estimation

[33]: MC-CNN [33] – Zbontar, 2015
[43]: MC-CNN variants [43] – Zbontar, 2016
[35]: Content-CNN [35] – Luo, 2016
[44]: SGM-Net [44] – Seki, 2017

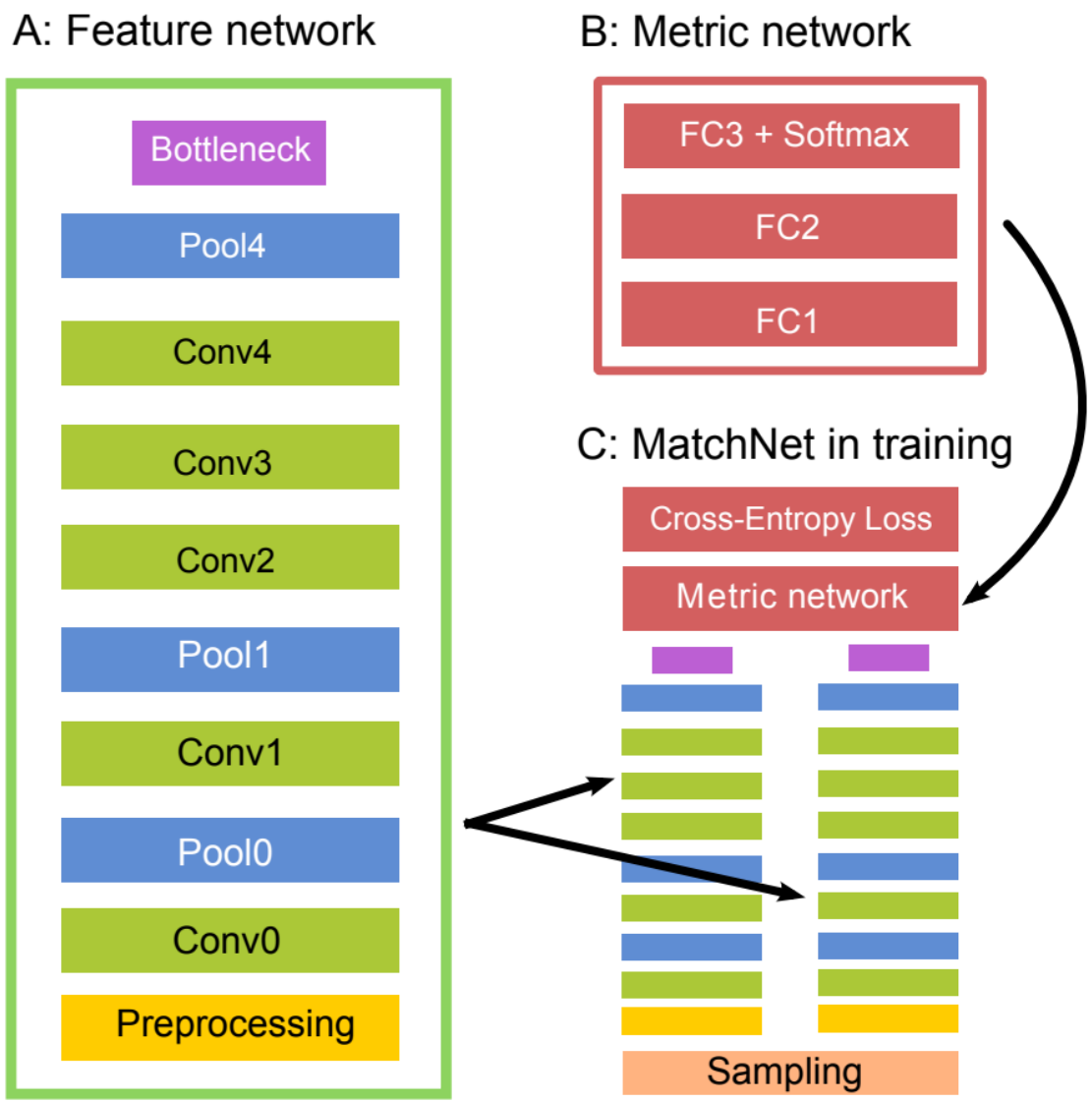


Slightly modified ConnectedPapers [40] graph for [33] as of May 2021

- **MatchNet** [41] use two AlexNet [15] models for feature extraction and a metric network (3 FC + softmax) on image-pairs

- Mimize cross-entropy with y_i as 0/1 label for input pair x_i

$$E = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

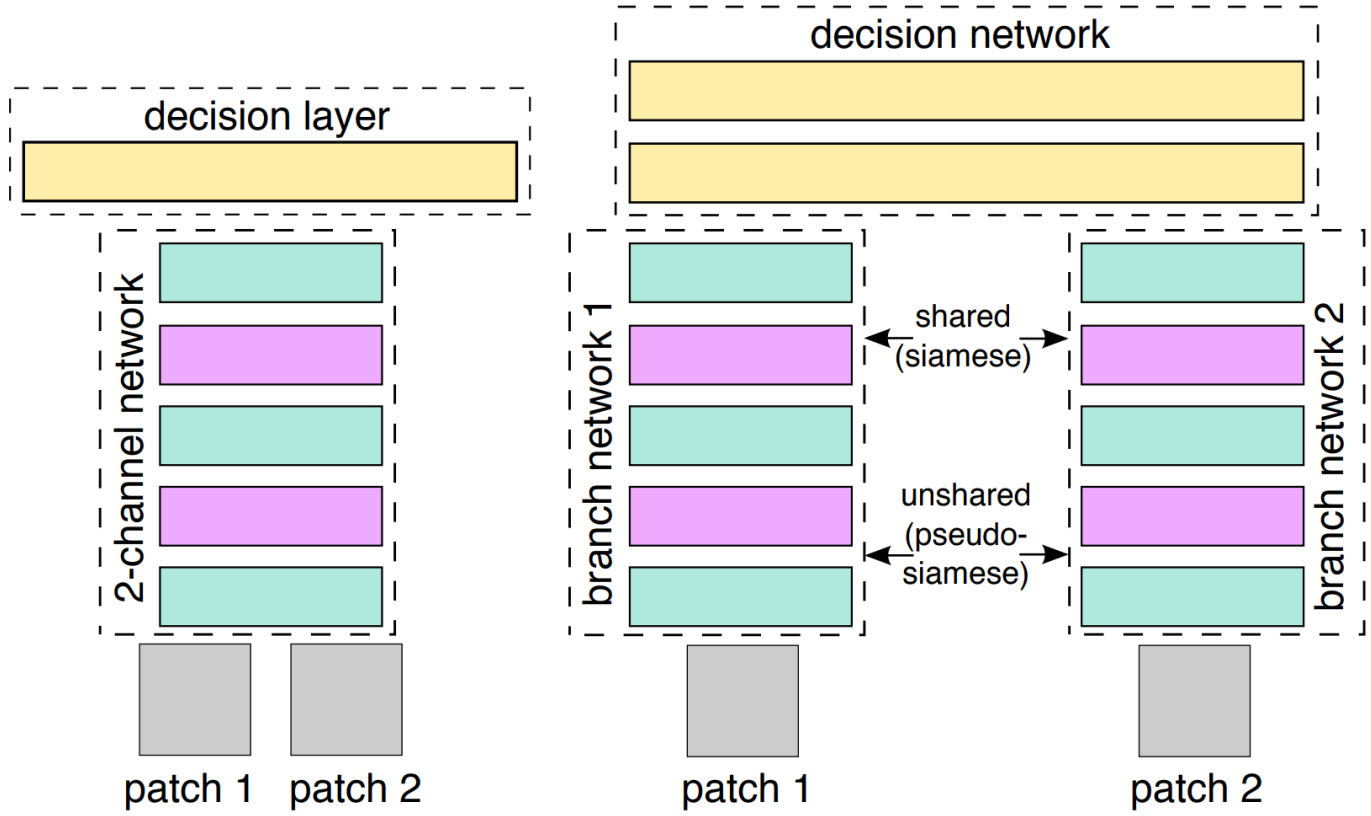


The architecture of MatchNet - figure taken from [41]

- **ZagoruykoNet** [42] learn a general similarity function for two image patches via
- a) 2 channels: each patch is an image channel
- b) siamese network: two branches with shared weights
- c) pseudo-siamese: two branches with unshared weights

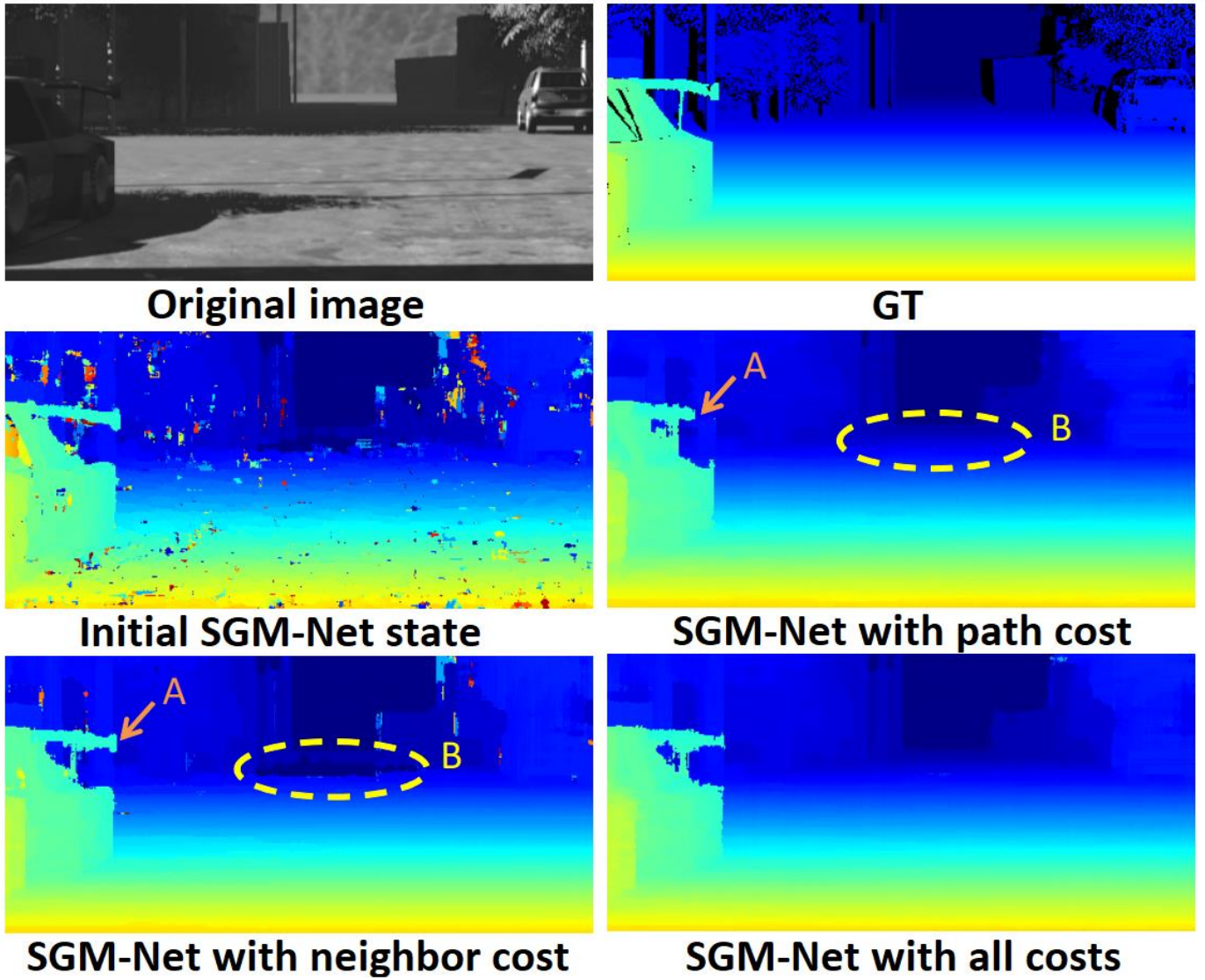
$$y_i \in \{-1, 1\}$$

$$\min_w \frac{\lambda}{2} ||w||_2 + \sum_{i=1}^{N} \max(0, 1 - y_i \hat{y}_i)$$



Three basic architectures a), b) and c) - figure taken from [42]

- **SGM-Net** [44] – regularization via Semi-global matching (SGM) [57]
- CNN is used for matching and learning SGM penalty-params. via
- a) path-cost: path traversing correct disparity at a pixel should have smaller cost than any other paths
- b) neighbor-cost: path traversing correct disparities in consecutive pixels must have smalled cost – this deals with the special cases of borders, slants and flat regions



Quantitative comparison results for loss functions – figure modified and taken from [44]

End-to-end supervised 2D CNNs for stereo vision

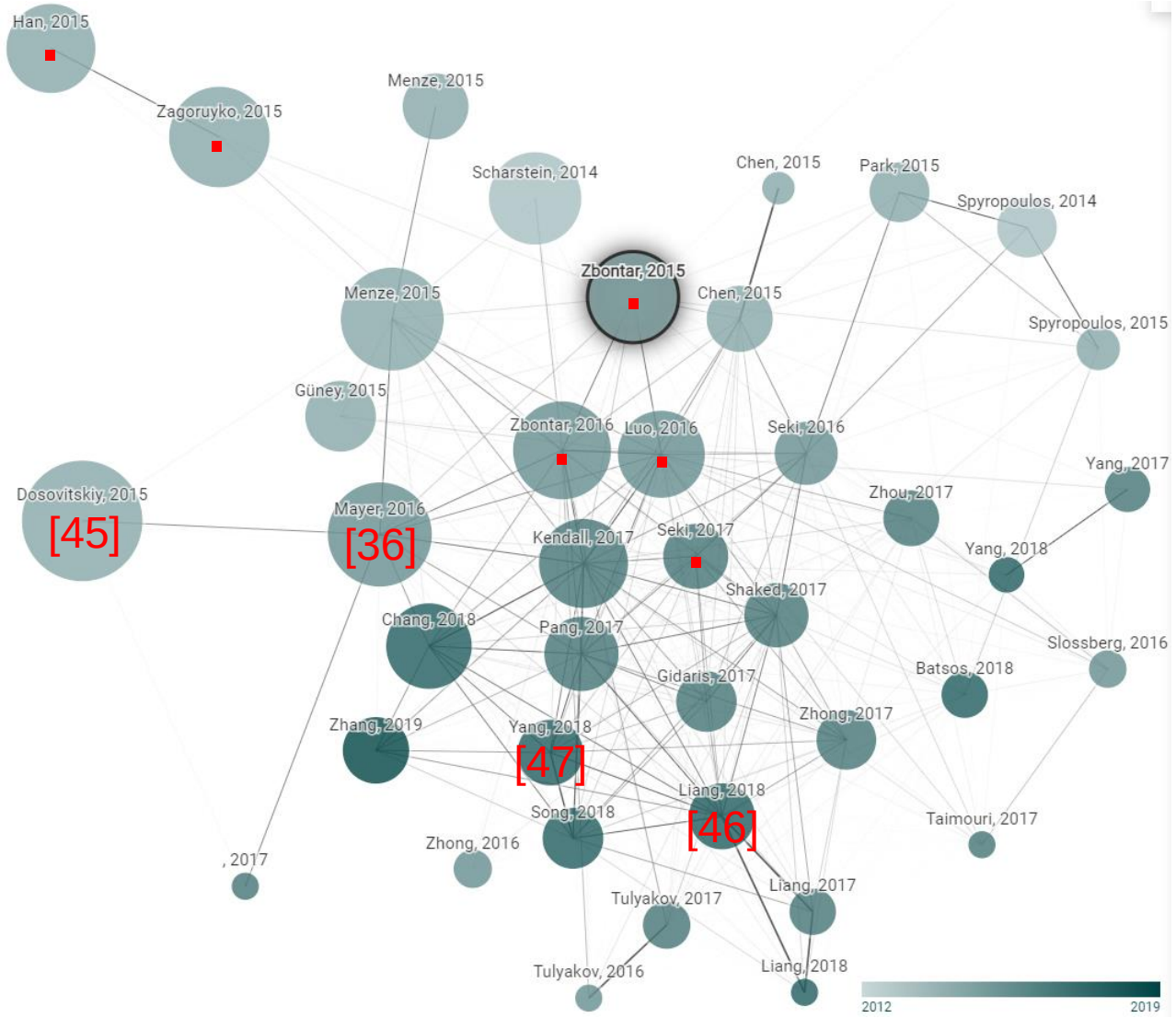
Optical flow works for stereo vision

0: FlowNet [45] – Dosovitskiy, 2015

2D CNN‘s for entire stereo pipeline

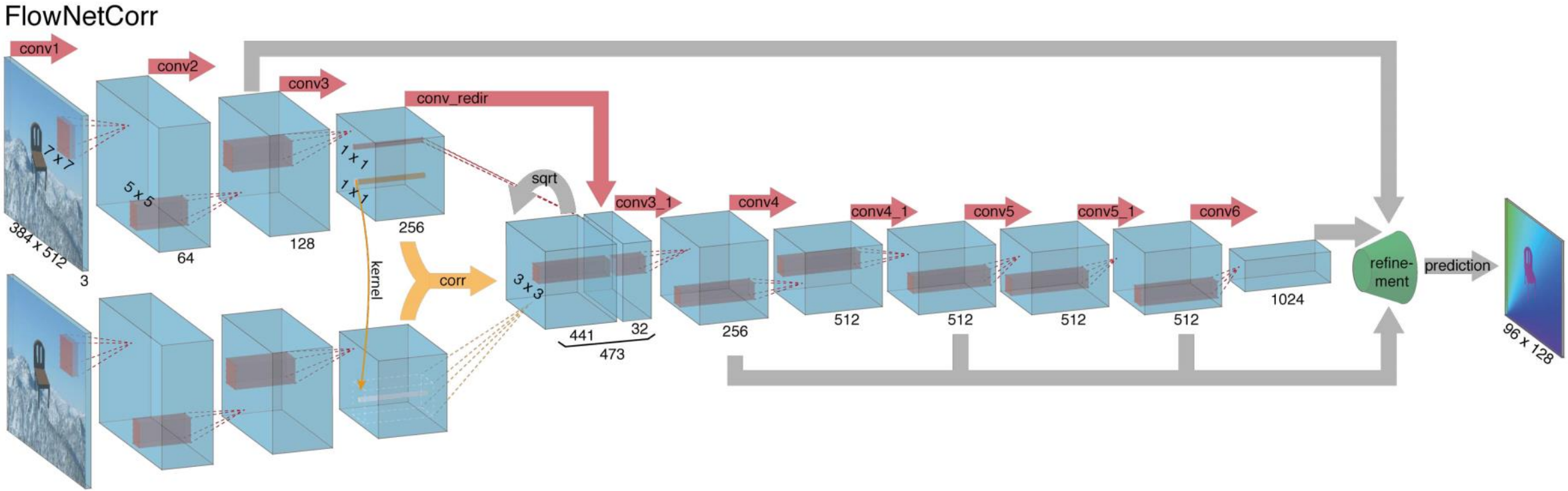
1: DispNet [36] – Mayer, 2016
2: iResNet [46] – Liang, 2018
3: SegStereo [47] – Yang, 2018

[62] HITNet as top-performer
(Slide 48f., preprint for CVPR 2021)



Slightly modified ConnectedPapers [40] graph for [33] as of May 2021

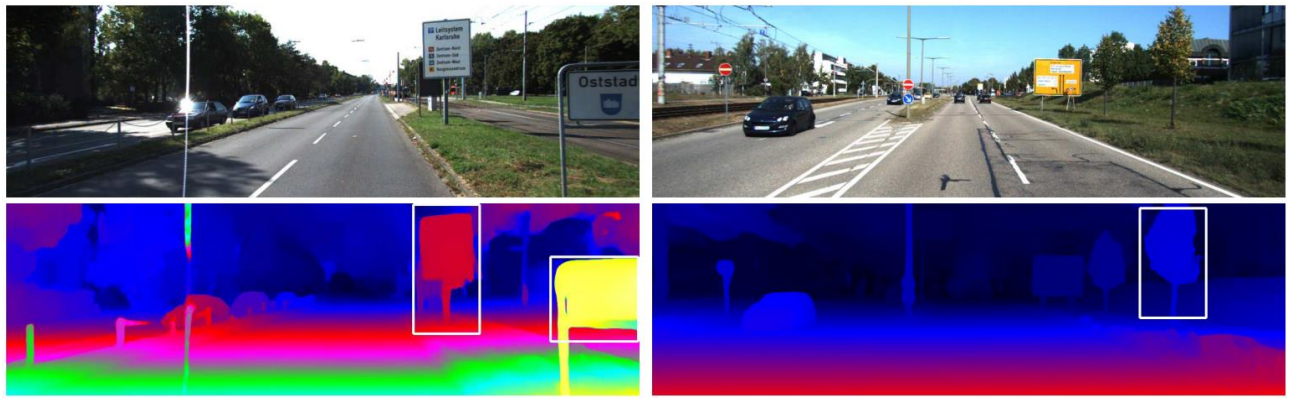
- **Flownet [45]** – first end-to-end network for stereo matching, proposed for optical flow
- Provides basic 2D encoder-decoder, matching via „correlation“ of two patches: convolution but data is convolved with other data
- Decoding (upsampling) takes place in the refinement



Red steps are convolutions, in yellow the correlation is obtained, while green modules provide the refinement – figure taken from [45]

- **DispNet [36]** – inspired by FlowNet, propose dataset & networks for disparity estimation, optical flow and scene flow estimation
- First end-to-end CNN method explicitly for disparity estimation
- Architecture very similar to FlowNet with additional convolutional layers

- **iResNet** [46] – network in three parts
  1. 1st calculates multi-scale shared features from left/right images
  2. 2nd performs cost computation, aggregation and disparity calculation
     - Calculate „feature constancy“ using initial disparity and features
  3. 3rdDisparity and feature constancy fed into subnetwork for refinement



Top row: images from KITTI [48]
Bottom row: iResNet disparity prediction
Figure taken from [46]

- **HITNet [62]** – state-of-art results on ETH3D [63], Middlebury-v3 [64] and KITTI [48]
- Uses a very small U-Net [60] for multi-resolution feature extraction $E = F(\boldsymbol{I}; \theta_F)$
- Initialization stage to extract initial disparity d^{init} and feature vector $\boldsymbol{p}^{init}$
- Slanted plane hypothesis with disparity d, gradients d_x, d_y and descriptor $\boldsymbol{h} = [d, d_x, d_y, \boldsymbol{p}]$
- Propagation stage to refine disparity hypotheses with slanted support windows

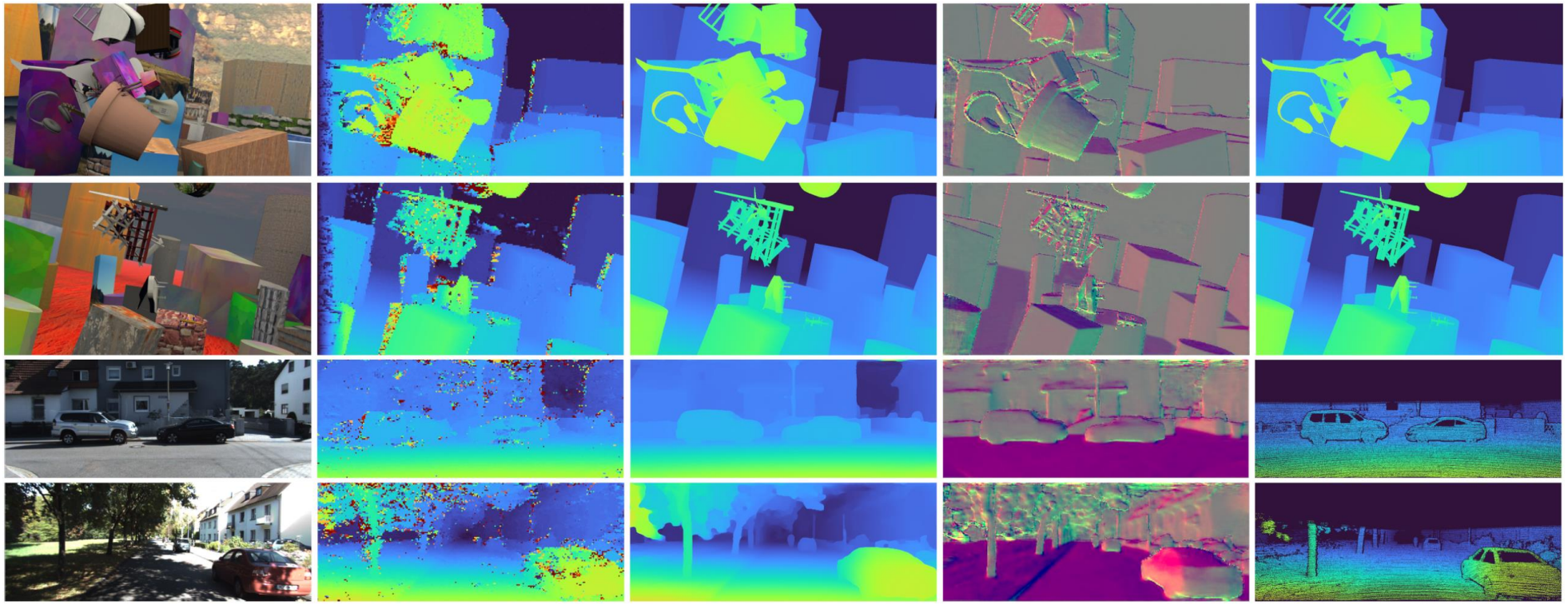


Figure taken from & results of HITNet [62] on SceneFlow [36] and KITTI [48]

Left-to-right: input image, initialization, disparity result, predicted slant and ground truth

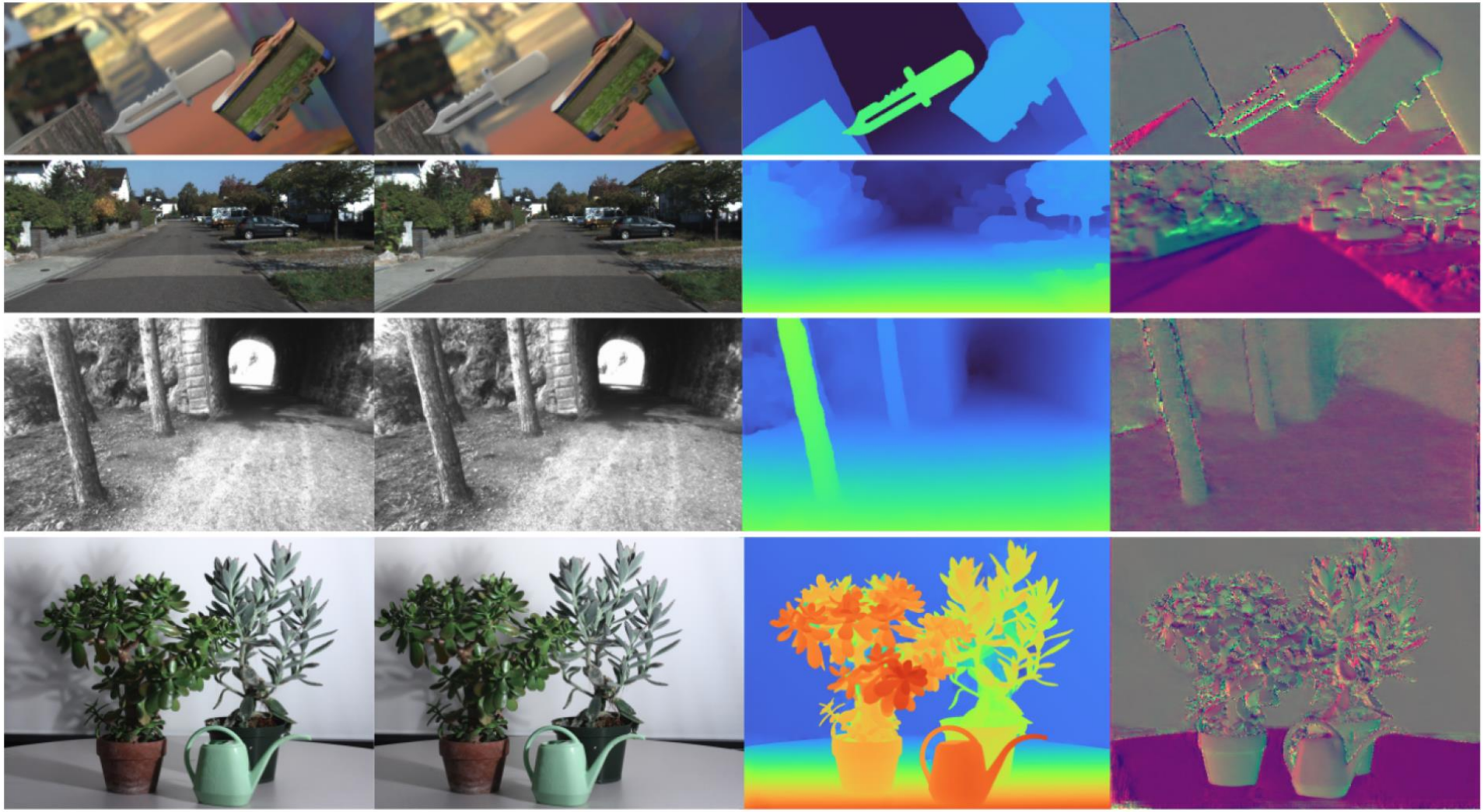


Figure taken from [62] showing example HITNet results on SceneFlow [36], KITTI [48], ETH3D [63] and Middlebury-v3 [64]
Using a Titan V GPU it runs 19ms per frame for ETH3D and KITTI (0.5Mpixel) and 108ms per Mpix for Middlebury

End-to-end super-vised 3D CNNs

3D convolutions to construct
4D cost volume

[33]: GC-Net – Kendall, 2017
[52]: StereoNet – Khamis, 2018
[53]: PSMNet – Chang, 2018

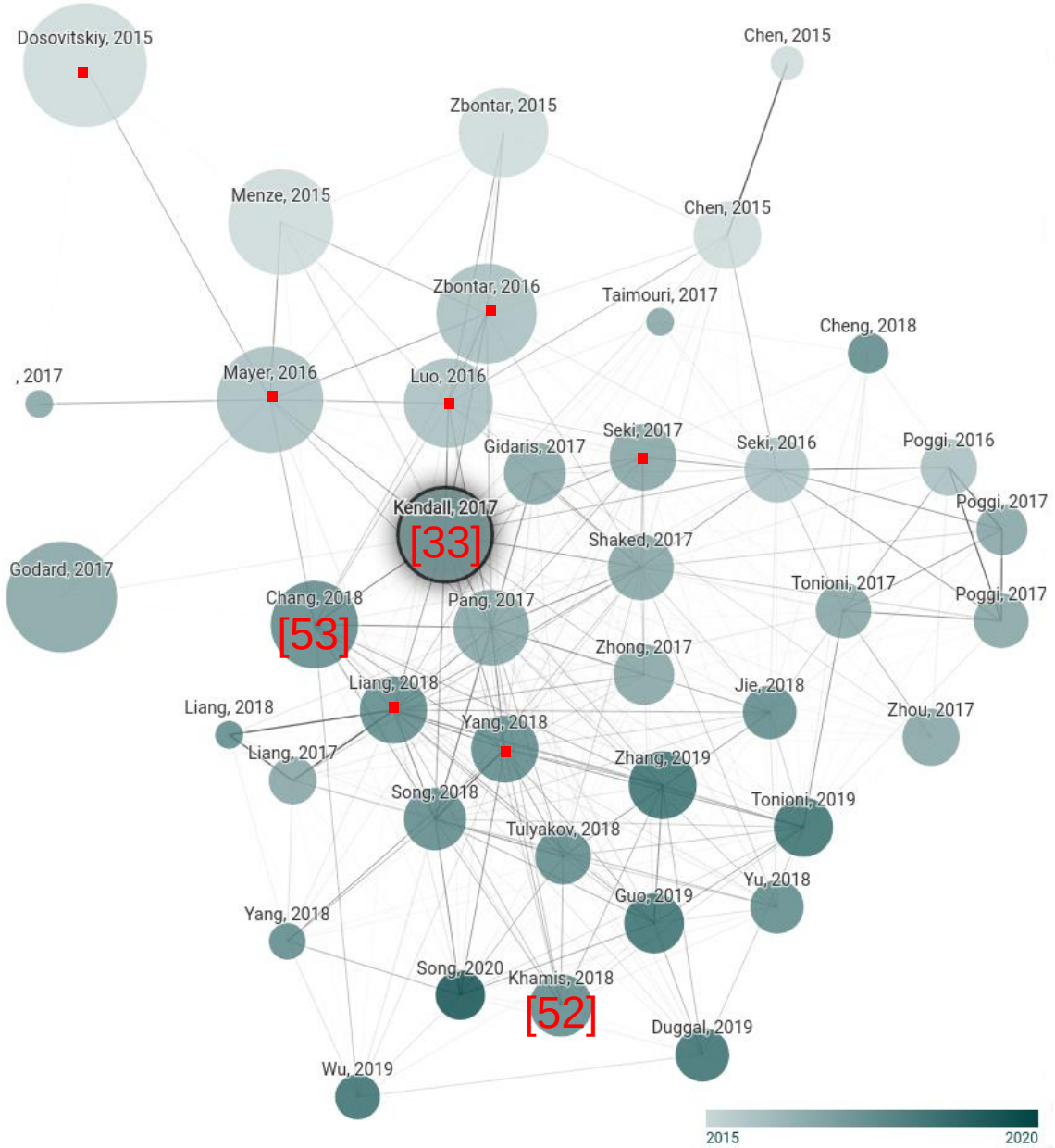


Figure on the right: slightly modified ConnectedPapers [40] graph for [33] as of May 2021

- **GCNet [33]** – first to propose 3D convolutions

1. 2D convolutions create featuremaps as usual
2. Each feature map is concatenated with all featuremaps across all disparities for the other image to form a „4D“ cost volume
3. 3D convolution then learns to regularize for width W, height H and disparity D
4. 3D deconvolutions yields final cost volume $\mathrm{H} \times W \times D$
5. Soft ArgMin, fully differentiable, regresses to disparities

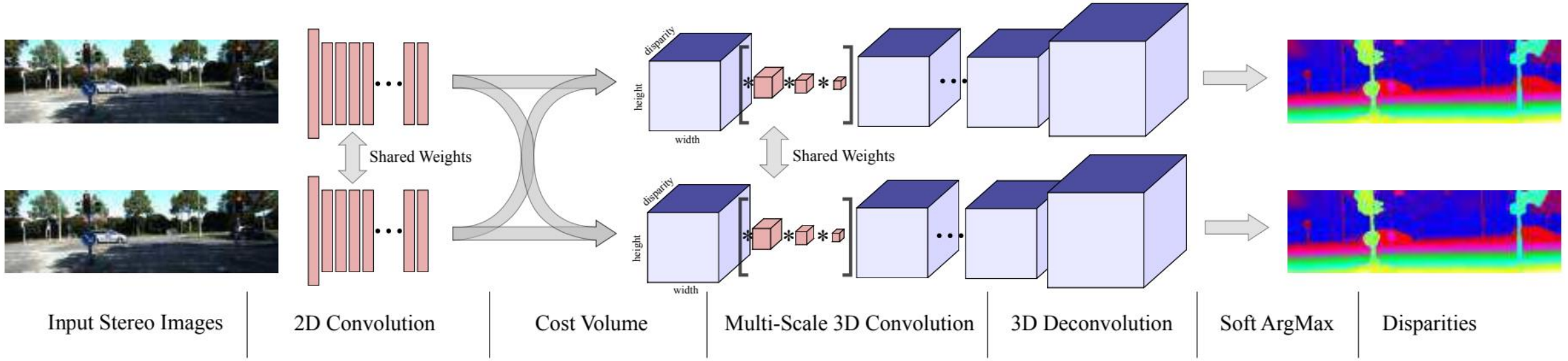
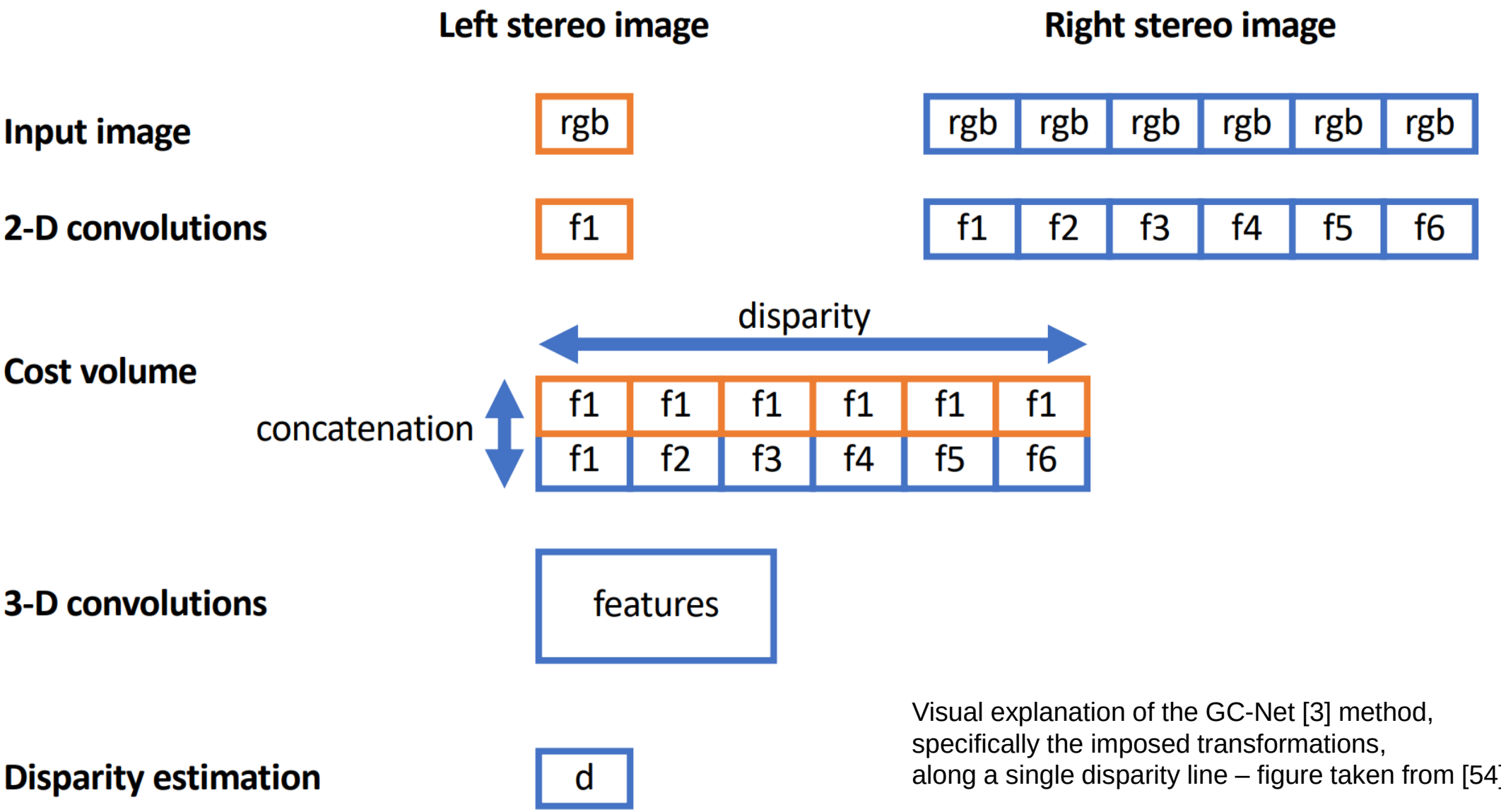
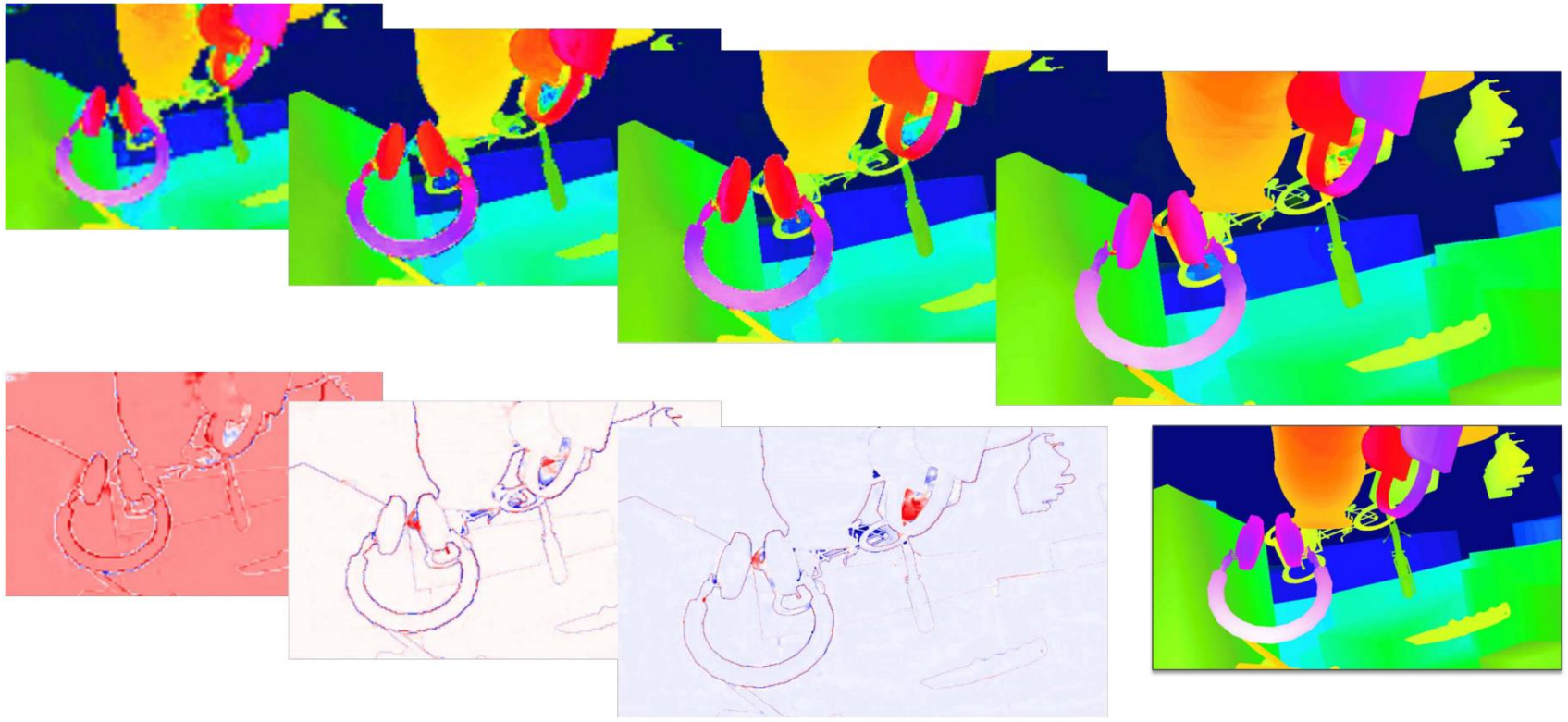


Figure taken from and architecture of GCNet [33]



Visual explanation of the GC-Net [3] method, specifically the imposed transformations, along a single disparity line – figure taken from [54]

- **StereoNet [52]** – first end-to-end stereo matching algorithm for **real-time** (60fps, 1xTitan X)
- Subsample the features to $\frac{W}{2^k} \times \frac{H}{2^k} \times \frac{(D+1)}{2^k}$ with K downsampling layers
- Also use 3D convolutions followed by soft argmin but disparities are coarse
- For refinement disparities are dilated/eroded via a network that learns pixel-to-pixel mapping
- Intuition that the network learns an edge-aware upsampling function with a RGB image guide



Hierarchical refinement steps in StereoNet [52]
Top row gives the result and bottom row the refinement output at each stage
Ground truth in the bottom right
Figure taken from [52]

Unsupervised Mono-Depth

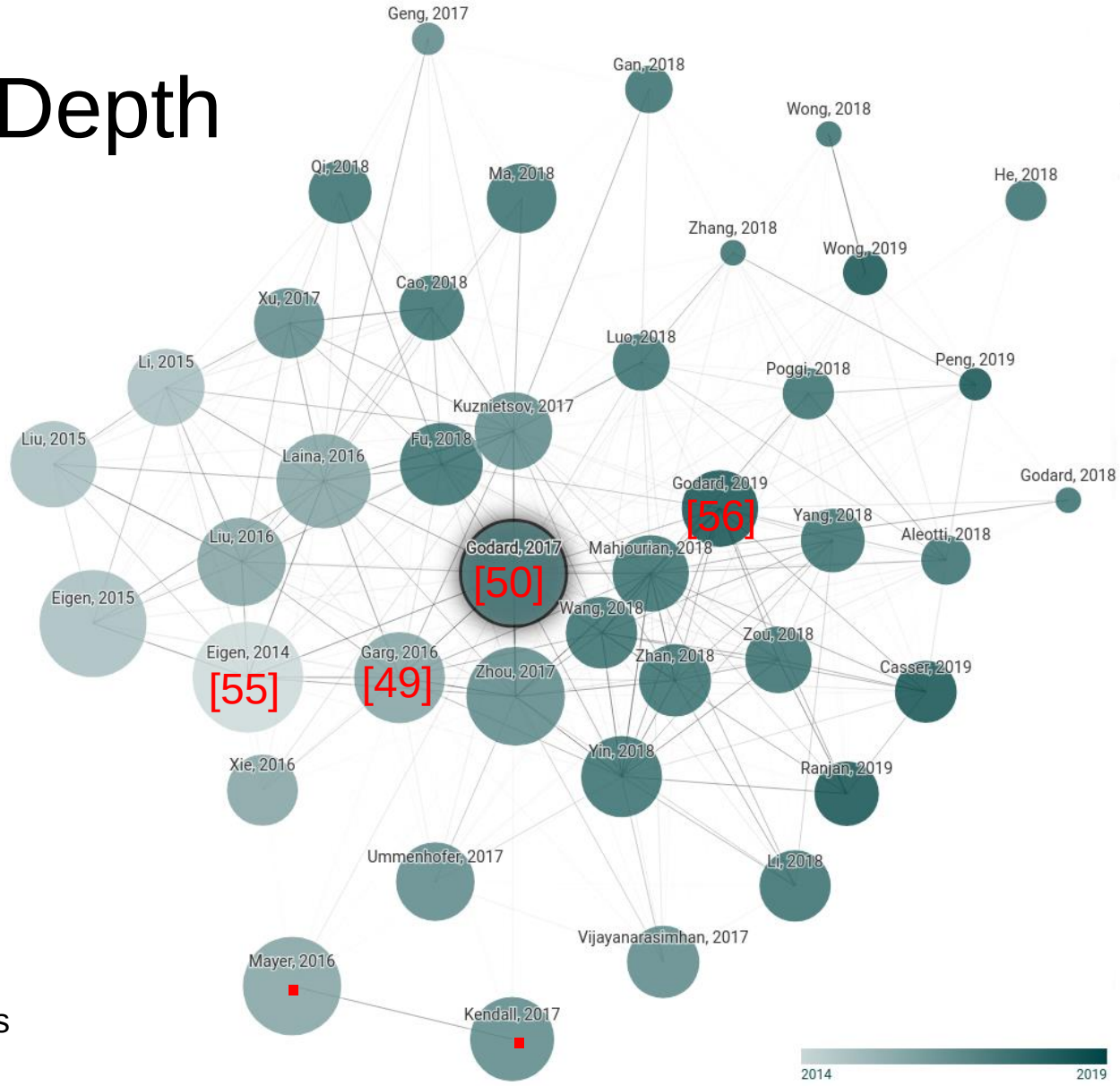
First depth maps from single image in 2014 [55] with (comparatively) poor results

Figures taken from [54] showing an input and result image for their model

Photometric reprojection error [49] as the driving force for unsupervised (monocular) depth estimation

[49]: GargNet – Garg, 2016
[50, 56]: Monodepth1/2 - Godard, 2017/9

Figure on the right: slightly modified ConnectedPapers [40] graph for [50] as of May 2021



- **GargNet [49]** – unsupervised depth estimation, left image as source and right as target

1. Train encoder to predict depth map for source image $I^l_{i,j}$
2. Inverse warp the target image with depth and e.g. baseline to $\tilde{I}^l_{i,j}$
3. Optimize photometric error (see slide 35)

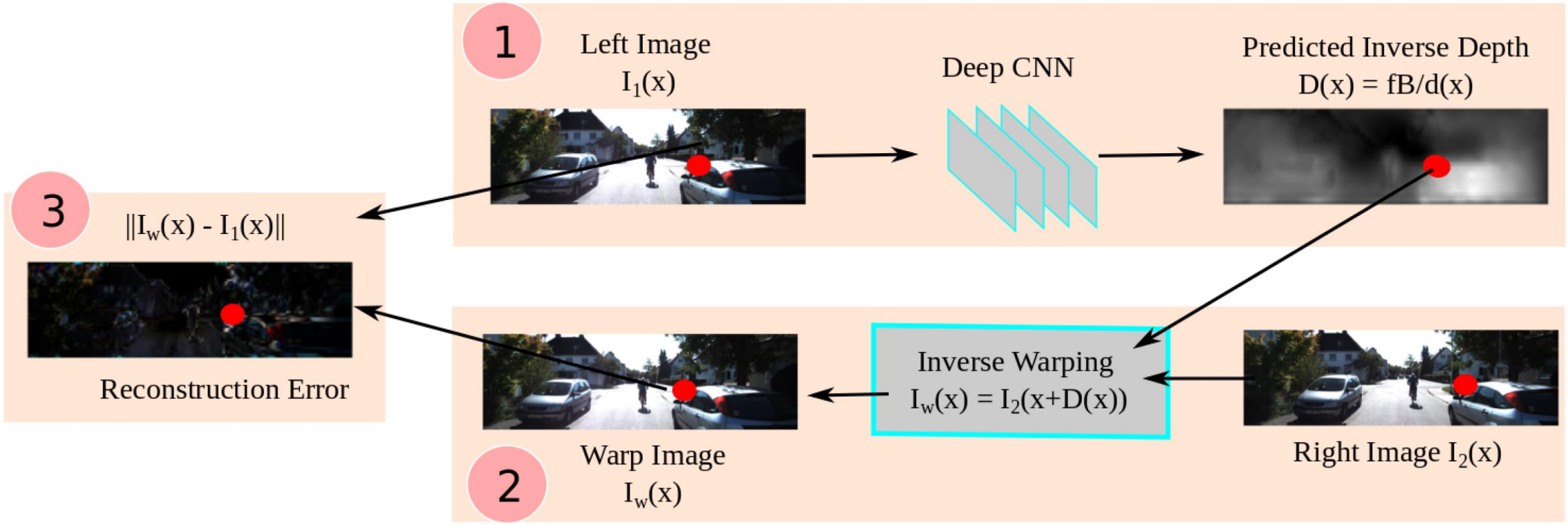


Figure taken from and architecture of GargNet [49]

- **GargNet [49]** as autoencoder (see slide 14) uses upsampling for last layers
- (-) Coarse and blurry predictions
- (+) Provides 100% coverage since no pixels are occluded (compared to true stereo)
- SGM → CNN denotes a standard SGM algorithm [57] for proxy ground-truths to train a CNN
- HS → CNN similarly first uses the HS algorithm [58]

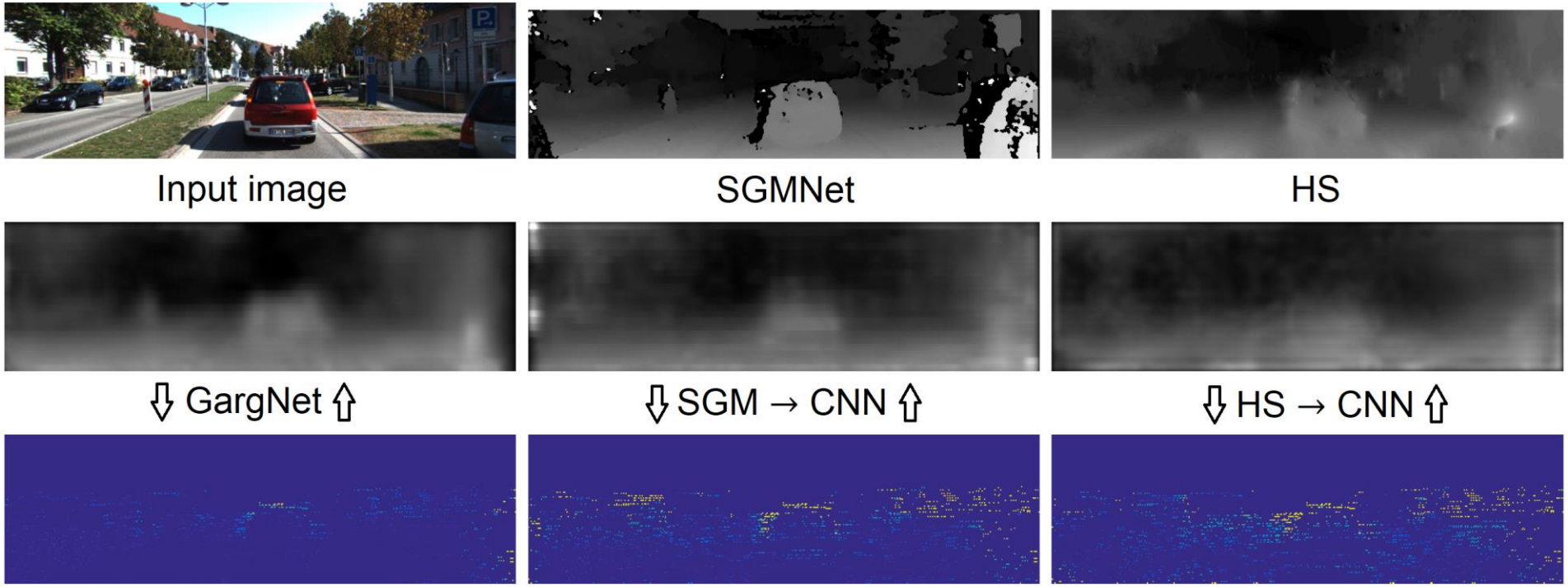
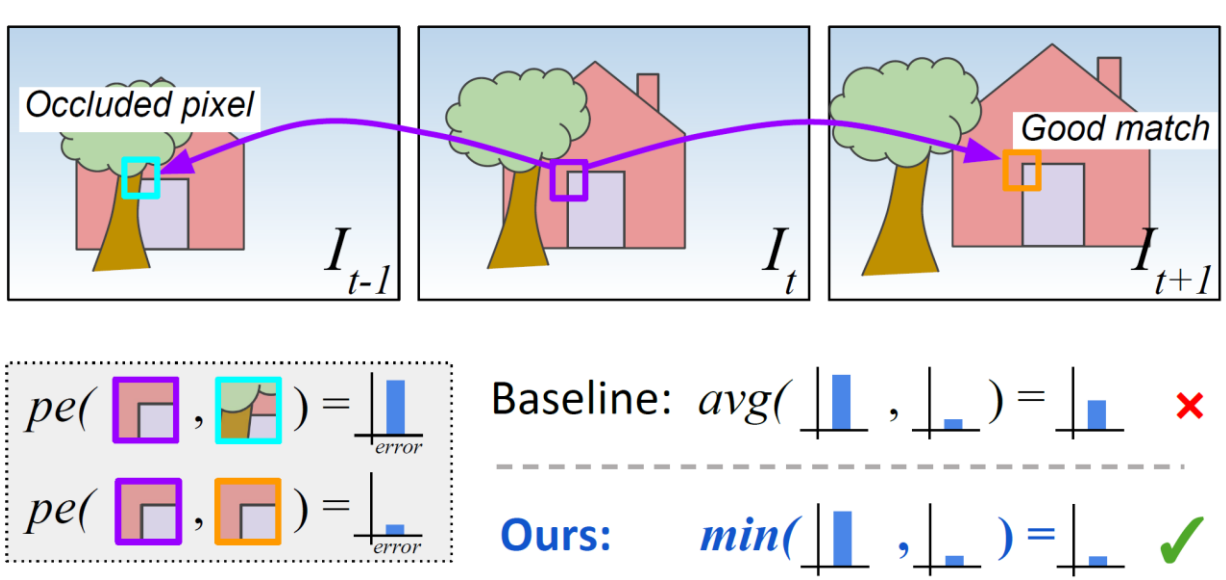
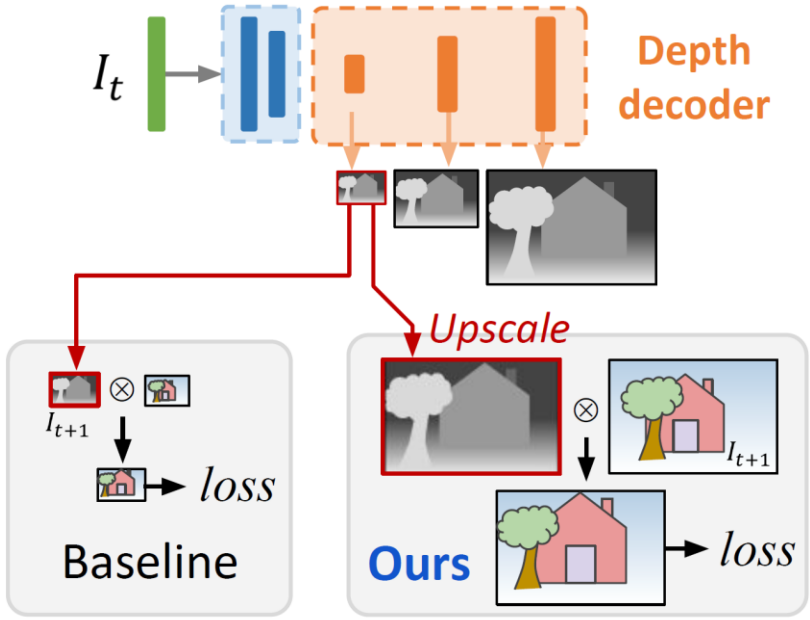


Figure slightly modified and taken from [49] showing depth results and errors (bottom row) of GargNet [49], SGMNet [44], HS algorithm [58] and SGM/HS derivatives

- **Monodepth2 [56]** – self-supervision from stereo or sequence of images
- Use a standard U-Net [60] to predict depth
- Argue that avg. error matches occluded pixels while minimum does not
- Upscale depth prediction at multiple layers to compute cost at input resolution
- Mask stationary (relative to camera) pixels using temporal sequence



Appearance loss as minimum of projection errors



Upsample at multiple scales to input size

Figure modified and taken from [56] showing an overview of the method minimum reprojection and multi-scale loss calculation

- **Monodepth2 [56]** – strong results using monocular (M) or stereo (S) supervision

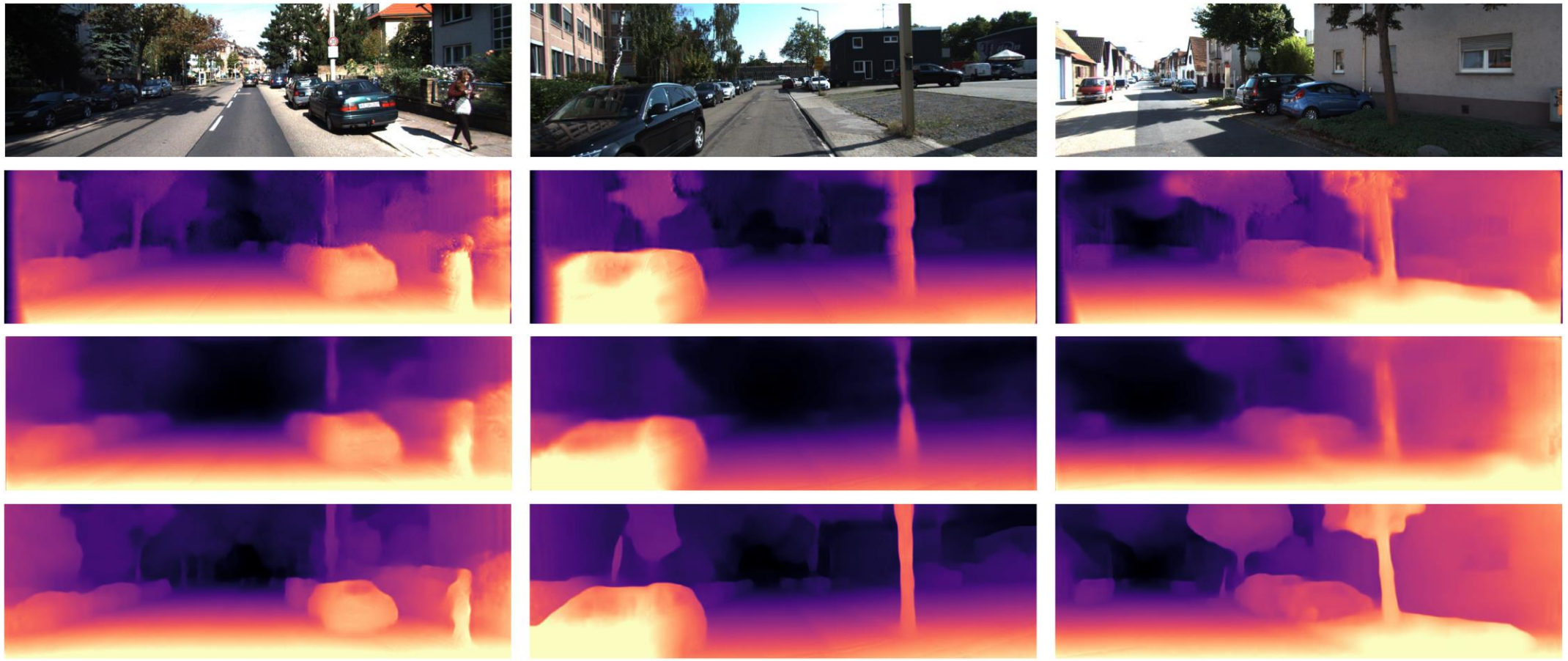


Figure modified and taken from [56] showing (top to bottom) KITTI Eigen [48] input image as well as quantitative results for methods Monodepth [50], EPC++(MS) [61] and Monodepth 2 [56] with monocular (M) supervision

Summary of learning-based stereo vision

- Classical stereo matching
- NN-based cost calculation
- End-to-end 2D CNNs
- End-to-end 3D CNNs
- Unsupervised monodepth

- HITNet[62] already looks really close to ground-truth: is (DL-based) stereo vision solved?

- Struggles with texture-less regions, occlusions, reflections
- Depending on sensor: requires dense LiDAR-data, requires expensive GPU

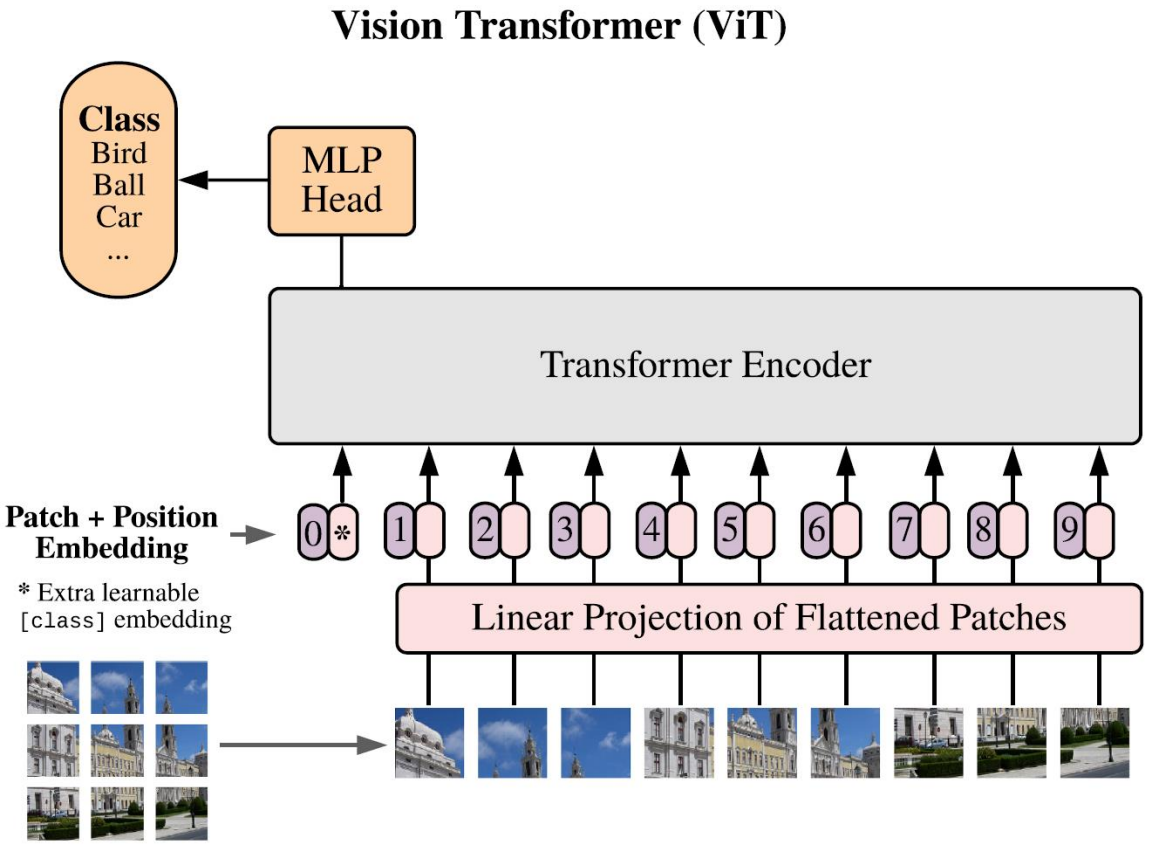
Outlook - Transformers

- Convolutions in CNNs go back to the age-old bias-variance dilemma [65]:
  - Size of convolutional kernel determines receptive field
  - Small (e.g. 3x3) capture local-context, biased
  - Largest reasonable is 7x7 – context can span much larger distances
  - Lack of global understanding, i.e. dependencies between features
  - How to capture long-range relations without increasing complexity?

- **Transformers**
  - Taken from natural language processing (NLP) – embeds words in codebook
  - Two main ideas [66]:
    - **Self-attention** to capture long-range relations between embeddings
    - **Pre-training** on large (un)labelled corpus in (un)supervised manner
      → fine-tuning for target task on small labeled dataset

Transformers in computer vision

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale [67]:



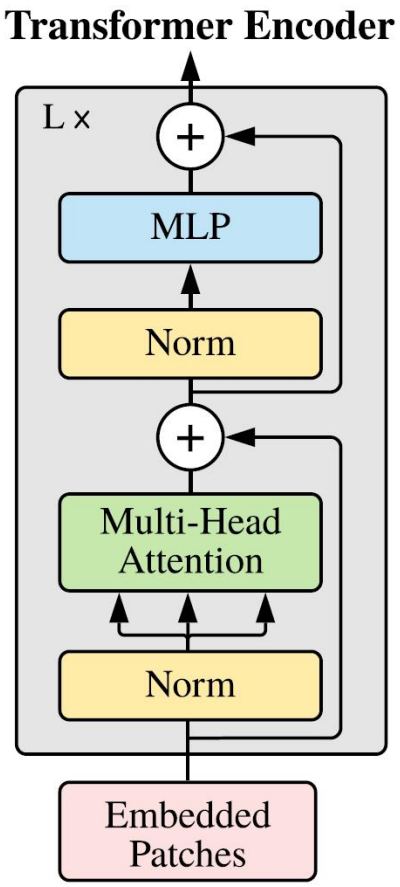


Image taken from [67] where the illustration of the transformer encoder was inspired by [68]

1. Image is split into patches
2. Embedded with position
3. Fed into transformer
4. Attention & dependencies
5. Classification with learnable “classification token”

Stereo vison with transformers: STTR [69]

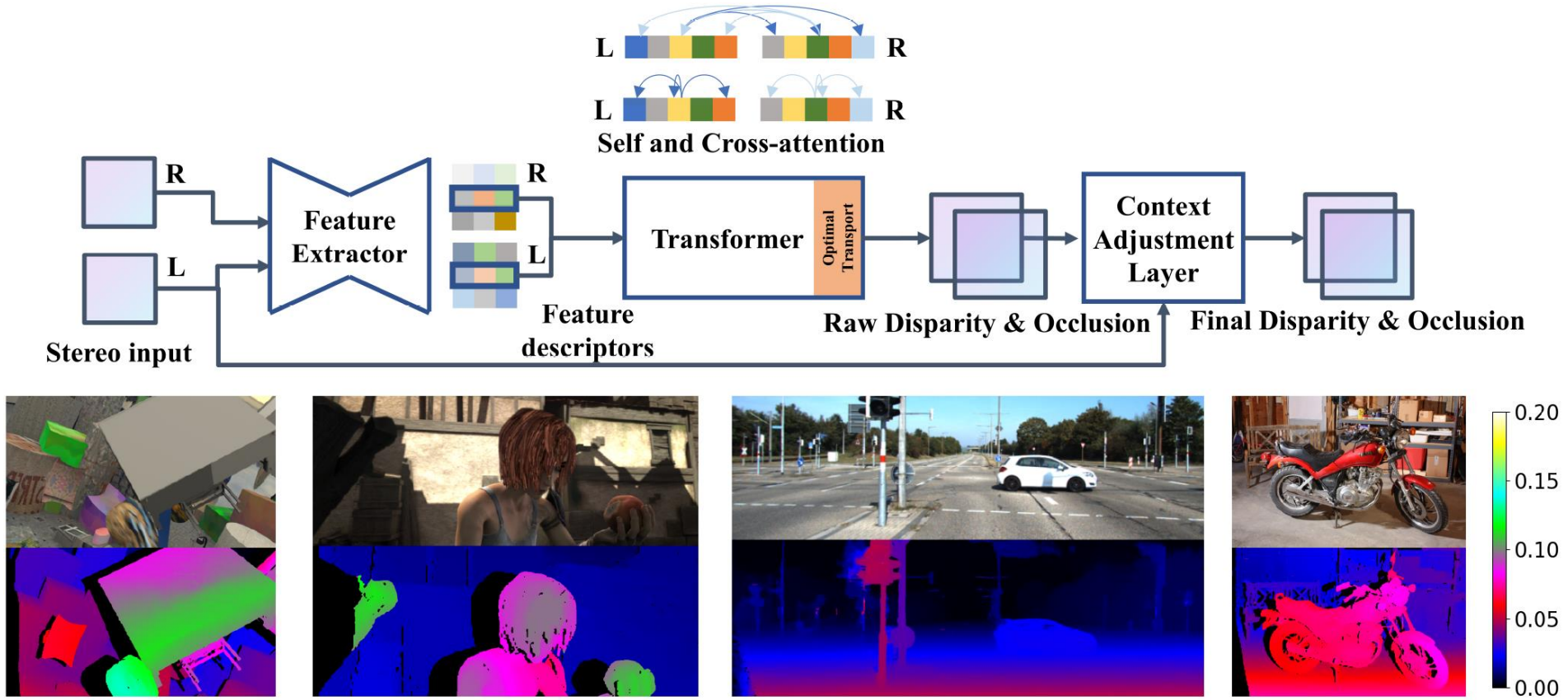


Figure taken from STTR [67] showing the network overview (top), left input images (middle) and inference results (bottom) of a model trained only on SceneFlow [36] - datasets from left to right: SceneFlow [36], MPI Sintel [70], KITTI [48] and Middlebury-v3 [64] - color map is relative to image width with the scale given at the right side

Mono-depth transformers: MiDaS [71] & DPT [72]

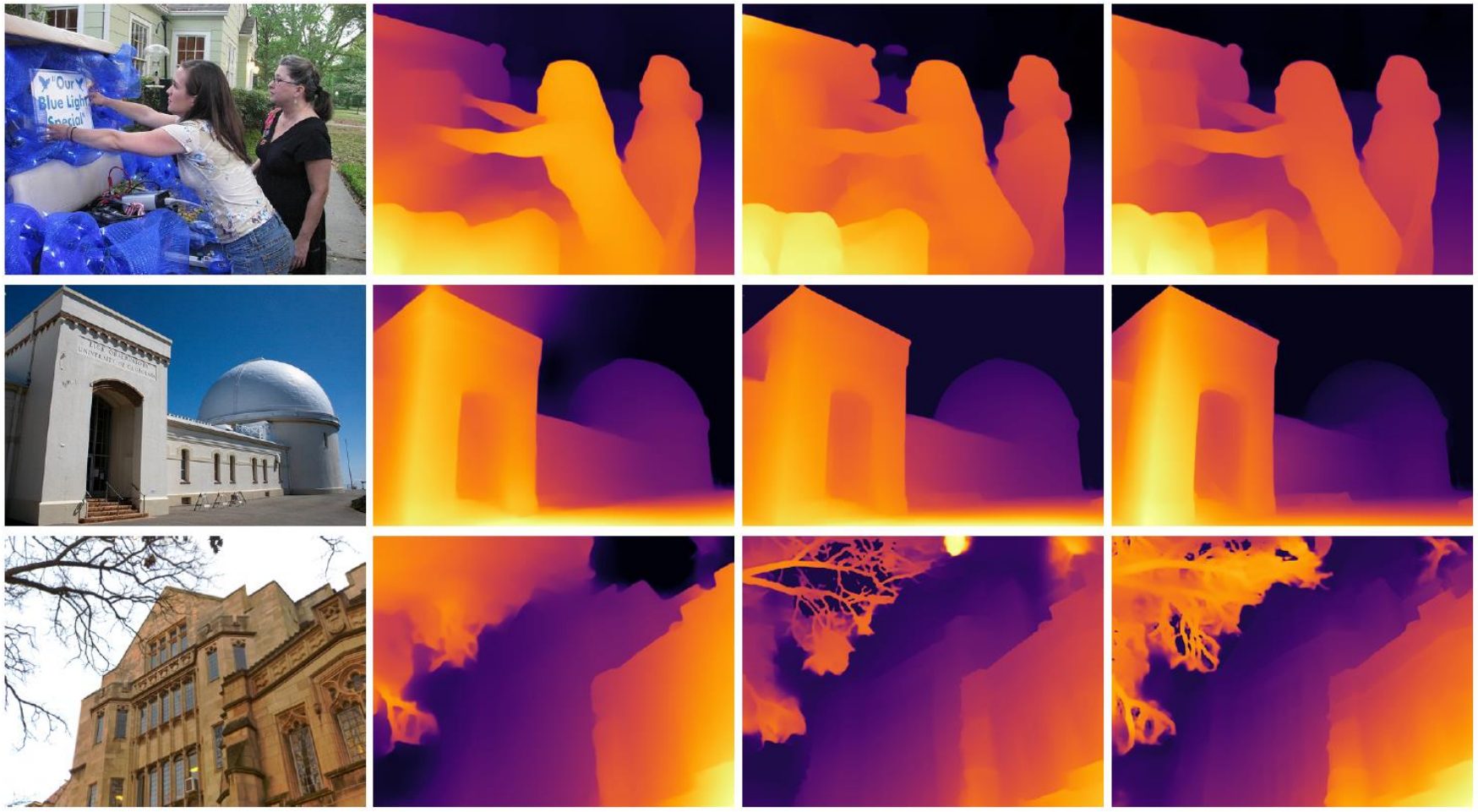


Figure taken from DPT [72] showing input images, MiDaS [71], DPT-Hybrid and DPT-Large [72] results

These are zero-shot cross-dataset transfers, i.e. inferences in **datasets** the model has not seen

Models were trained on *MIX 6* a meta-dataset composed of 1.4 million images for monocular depth estimation [72]

Questions?

Suggestions?

Complaints?

Thank you for your attention

References

[1] C.M. Bishop, *Pattern Recognition and Machine Learning,* Singapore: Springer, 2006. (PDF)

[2] J. Schmidhuber, “Deep learning in neural networks: An overview”, *Neural networks,* 2015, 61, pp.85-117, 2015. (PDF)

[3] I. Goodfellow, Y. Bengio and A. Courville, *Deep learning*, Cambridge: MIT press, 2016. (e-Book)

[4] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach,* Fourth edition, New Jersey: Pearson, 2021. (accompanying website)

[5] M. Bronstein, J. Bruna, T. Cohen and P. Velickovic, “Geometric Deep Learning”, GDL course part of AMMI 2021, 2021. [Online]. Available at: https://geometricdeeplearning.com/lectures/ (Accessed: 22.10.2021).

[6] T. Gärtner, M. Thiessen and A. Sepliarskaia, “194.100 Theoretical Foundations and Research Topics in Machine Learning”. [Online]. Available at: https://preview.tinyurl.com/yf73xbu7 (Accessed: 11.05.2021).

[7] M. Charikar and C. Ré, “CS229: Machine Learning”. [Online]. Available at: http://cs229.stanford.edu (Accessed: 11.05.2021).

[8] L. FeiFei, K. Ranjay, X. Danfei, „CS231n: Convolutional Neural Networks for Visual Recognition“. [Online]. Available at: http://cs231n.stanford.edu (Accessed: 11.05.2021).

[9] L. Fridman, „MIT Deep Learning and Artificial Intelligence Lectures“. [Online]. Available at: https://deeplearning.mit.edu (Accessed: 11.05.2021).

[10] G. Sanderson, „Deep Learning Series“. [Online]. Available at: https://preview.tinyurl.com/m92b8r9u (1st part) (Accessed: 11.05.2021).

[11] U. Von Luxburg, „Mathematics for Machine Learning“. [Online]. Available at: https://preview.tinyurl.com/jk9p3m9x (Accessed: 11.05.2021).

[12] L. Von Rueden, S. Mayer, R. Sifa, C. Bauckhage and J. Garcke, „Combining Machine Learning and Simulation to a Hybrid Modelling Approach: Current and Future Directions“, *International Symposium on Intelligent Data Analysis (IDA)*, 2020, pp.548-560, 2020. (PDF)

[13] H. Dawar, *Stochastic Gradient Descent.* [Online]. Available at: https://preview.tinyurl.com/erz8k26h (Accessed: 26.04.2021).

[14] J. Duchi, „Introduction to Convex Optimization for Machine Learning“, *Practical Machine Learning, Fall 2009*, UC Berkeley, 2009. (PDF)

[15] A. Ivakhnenko and V.G. Lapa, *Cybernetic predicting devices,* New York: CCM Information Corp., 1965. (accompanying website)

[16] J. Schmidhuber, “Learning Complex, Extended Sequences using the Principle of History Compression”, *Neural Computation*, 1992, 4(2), pp.234-242, 1992. (PDF)

[17] S. Hochreiter, *Untersuchungen zu Dynamischen Neuronalen Netzen*. München: Technische Universität München (master‘s thesis, german), 1991. (PDF)

[18] A. Krizhevsky, I. Sutskever and G.E. Hinton, “Imagenet Classification with Deep Convolutional Neural Networks”, *Advances in Neural Information Processing Systems (NIPS)*, 2015, 25, pp.1097-1105, 2015. (PDF)

[19] W. Fedus, B. Zoph and N. Shazeer, *Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity*. [Online]. Available at: https://arxiv.org/abs/2101.03961 (Accessed: 26.04.2021).

References

[20] Original source unknown. Taken from: L. Langer, „Algorithm Journey from PC to IoT – A Signal Processing Pipeline with TensorFlow 2.X on a Smartwatch“. [Online]. Available at: https://preview.tinyurl.com/5xfkchpb (Accessed: 11.05.2021).

[21] Ž. Ivezić, A.J. Connolly, J.T. Vanderplas and A. Gray, *Statistics, Data Mining and Machine Learning in Astronomy,* New Jersey: Princeton University Press, 2014. (accompanying website)

[22] S. Linnainmaa, The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors.* Helsinki: University of Helsinki (master's thesis, finnish)*,* 1970. (following Springer paper)

*[23] L. Euler, *Methodus inveniendi lineas curvas maximi minimive proprietate gaudentes,* Lausanne & Geneva: Marcum-Michaelem Bousquet, 1744. (euler archive)

*[24] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition", *Biological Cybernetics,* 1980, 36, pp. 193-202, 1980. (PDF)

[25] S. Saha, *A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way.* [Online]. Available at: https://preview.tinyurl.com/6d8njrv6 (Accessed: 26.04.2021).

[26] Aphex34, *typical CNN architecture.* [Online]. Available at: https://preview. tinyurl.com/33xvmpfm (Accessed: 26.04.2021).

[27] M.D. Zeiler and R. Fergus, „Visualizing and Understanding Convolutional Networks“, *ECCV 2014*, 2014, LNCS 8689, pp.818-833, 2014. (arXiv preprint)

[28] H. Bandyopadhyay, "An Introduction to Autoencoders: Everything You Need to Know", [Online]. Available at: https://preview.tinyurl.com/474f7n6k (Accessed: 09.08.2021).

[30] D. Scharstein and R. Szeliski, „A taxonomy and evaluation of dense two-frame stereo correspondence algorithms“, *International Journal of Computer Vision*, 2002, 47 (1/2/3), pp.7-42, 2002. (PDF)

[31] M.S. Hamid, N.A. Manap, R.A. Hamzah and A.F. Kadmin, „Stereo matching algorithm based on deep learning: A survey“, *Journal of King Saud University – Computer and Information Sciences*, 32, 2020. (ScienceDirect)

[32] K. Zhou, X. Meng and B. Cheng, „Review of stereo matching algorithms based on deep learning“, *Computational Intelligence and Neuroscience*, Volume 2020, 2020.

[33] J. Zbontar and Y. LeCun, „Computing the stereo matching cost with a convolutional neural network“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2015, pp.1592-1599, 2015. (PDF)

[34] Z.Y. Chen, X. Sun, L. Wang, Y.A. Yu and C. Huang, „A deep visual correspondence embedding model for stereo matching costs“, *Proceedings of the International Conference on Computer Vision (ICCV),* 2015, pp.972-980, 2015. (PDF)

References

[35] W. Luo, A. Schwing and R. Urtasun, „Efficient Deep Learning for Stereo Matching“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp.5695-5703, 2016. (PDF)

[36] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy and T. Brox, „A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2016, pp.4040-4048, 2016. (PDF)

[37] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy and A. Bachrach, „End-to-End learning of Geometry and Context for Deep Stereo Regression“, *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp.66-75, 2017. (PDF)

[38] J. Flynn, I. Neulander, J. Philbin and N. Snavely, „DeepStereo: learning to predict new views from the world‘s imagery“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recoginition (CVPR),* 2016, pp.5515-5524, 2016. (PDF)

[39] J.Y. Xie, R. Girshick and A. Farhadi, „Deep3D: Fully Automatic 2D-to-3D Video Conversion with Deep Convolutional Neural Networks“, *Proceedings of the European Conference on Computer Vision (ECCV),* 2016, pp.842-857, 2016. (PDF)

[40] A.T. Eitan, E. Smolyansky, I.K. Harpaz and S. Perets, *Connected Papers.* [Online]. Available at: https://www.connectedpapers.com (Accessed: 04.05.2021).

[41] X. Han, T. Leung, Y. Jia, R. Sukthankar and A.C. Berg, „MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2015, pp.3279-3286, 2015. (PDF)

[42] S. Zagoruyko and N. Komodakis, „Learning to Compare Image Patches via Convolutional Neural Networks“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2015, pp. 4353-4361, 2015. (PDF)

[43] J. Zbontar and Y. LeCun, „Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches“, *Journal of Machine Learning Research,* 17 (2016), pp.1-32, 2016. (PDF)

[44] A. Seki and M. Pollefeys, „SGM-Nets: Semi-Global Matching With Neural Networks“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp.231-240, 2017. (PDF)

[45] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers and T. Brox, „FlowNet: Learning Optical Flow With Convolutional Networks, *Proceedings of the IEEE International Conference on Computer Vision (ICCV),* 2015, pp.2758-2766, 2015. (PDF)

[46] Z. Liang, Y. Feng, Y. Guo, H. Liu, W. Chen, L. Qiao, L. Zhou and J. Zhang, „Learning for Disparity Estimation Through Feature Constancy“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2811-2820, 2018. (PDF)

[47] G. Yang, H. Zhao, J. Shi, Z. Deng and J. Jia, „SegStereo: Exploiting Semantic Information for Disparity Estimation“, *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 636-651, 2018. (PDF)

References

[48] M. Menze, C. Heipke and A. Geiger, „Object scene flow.“ *ISPRS Journal of Photogrammetry and Remote Sensing*, 2018, 140, pp.60-76. (PDF)

[49] R. Garg, V. Kumar, G. Carneiro, I. Reid, „Unsupervsied CNN for Single View Depth Estimation: Geometry to the Rescue“, *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 740-756, 2016. (PDF)

[50] C. Godard, O. MacAodha and G.J. Brostow, „Unsupervised monocular depth estimation with left-right consistency“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp.270-279, 2017. (PDF)

[51] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, „The Cityscapes Dataset for Semantic Urban Scene Undestanding“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp.3213-3223, 2016. (PDF)

[52] S. Khamis, S. Fanello, C. Rhemann, A.Kowdle, J. Valentin and S. Izadi, „StereoNet: Guided Hierarchical Refinement for Real-Time Edge-Aware Depth Prediction“, *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp.573-590. (PDF)

[53] J.R. Chang and Y.S. Chen, „Pyramid Stereo Matching Network“, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp.5410-5418, 2018. (PDF)

[54] A.G. Kendall, *Geometry and Uncertainty in Deep Learning for Computer Vision*, Cambridge: University of Cambridge (doctoral dissertation), 2018. (PDF)

[55] D. Eigen, C. Puhrisch and R. Fergus, „Depth map prediction from a single image using a multi-scale deep network“, *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp.2366-2374, 2014. (PDF)

[56] C. Godard, O.M. Aodha, M. Firman and G.J. Brostow, „Digging Into Self-Supervised Monocular Depth Estimation“, *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp.3828-3838, 2019. (PDF)

[57] H. Hirschmüller, „Accurate and efficient stereo processing by semi-global matching and mutual information“, *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05),* 2005, Vol.2, pp.807-814, 2005. (PDF)

[58] B.K. Horn and B.G. Schunck, „Determining optical flow“, *Artificial Intelligence*, 17, 1-3, pp.185-203, 1981. (PDF)

[59] Z. Wang, A.C. Bovik, H.R. Sheikh and E.P. Simoncelli, „Image Quality Assessment: From Error Visibility to Structural Similarity“, *Transactions on Image Processing*, 13(4), pp.600-612, 2004. (PDF)

[60] O. Ronneberger, P. Fischer and T. Brox, „U-net: Convolutional Networks for Biomedical Image Segmentation“, *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015, Springer, pp.234-241, 2015. (PDF)

[61] C. Luo, Z. Yang, P. Wang, Y. Wang, W. Xu, R. Nevatia and Y. Yuille, „Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10), pp.2624-2641, 2019. (PDF)

[62] V. Tankovich, C. Häne, Y. Zhang, A. Kowdle, S. Fanello and S. Bouaziz, *HITNet: Hierarchical Iterative Tile Refinement Network for Real-time Stereo Matching*. [Online] Available at: https://arxiv.org/abs/2007.12140 (Accessed: 10.05.2021).

References

[63] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, A. Geiger, "A Multi-View Stereo Benchmark with High-Resolution Images and Multi-Camera Videos", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2017, pp.3260-3269, 2017. (PDF)

[64] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang and P. Westling, „High-Resolution Stereo Datasets with Subpixel-Accuracte Ground Truth“, *German Conference on Pattern Recognition (GCPR)*, 2014, pp.31-42, 2014. (PDF)

[65] S. Geman, E. Bienenstock and Rene Doursat, „Neural Networks and the Bias/Variance Dilemma“, *Neural Computation*, 4(1), 1-58, 1992. (PDF)

[66] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan and M. Shah, *Transformers in Vision: A Survey*. [Online]. Available at: https://arxiv.org/abs/2101.01169 (Accessed: 10.05.2021)

[67] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly and J. Uszkoreit, *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. [Online]. Available at: https://arxiv.org/abs/2010.11929 (Accessed: 10.05.2021)

[68] A. Vaswani, N. Shazeer, N. Parmar, J. Uskoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, „Attention Is All you Need“, *Proceeding of the 31st International Conference on Neural Information Processing (NIPS)*, 2017, pp.6000-6010, 2017. (PDF)

[69] Z. Li, X. Liu, N. Drenkow, A. Ding, F.X. Creighton, R.H. Taylor, M. Unberath, *Revisiting Stereo Depth Estimation From a Sequence-to-Sequence Perspective with Transformers*. [Online]. Available at: https://arxiv.org/abs/2011.02910 (Accessed: 10.05.2021)

[70] D.J. Butler, J. Wulff, G.B. Stanley and M.J. Black, „A Naturalistic Open Source Movie for Optical Flow Evaluation“, *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012, pp.611-625, 2012. (PDF)

[71] R. Ranftl, K. Lasing, D. Hafner, K. Schindler and V. Koltun, „Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer“, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, doi: 10.1109/TPAMI.2020.3019967, 2020. (arXiv)

[72] R. Ranftl, A. Bochkovskiy and V. Koltun, *Vision Transformers for Dense Prediction*. [Online]. Available at: https://arxiv.org/abs/2103.13413 (Accessed: 10.05.2021)

[73] Saama, „Different Kinds of Convolutional Filters“. [Online]. Available at: https://preview.tinyurl.com/2p98v4ks (Accessed: 26.11.2021) (Original image source unknown)