Deep Scene Understanding from Images

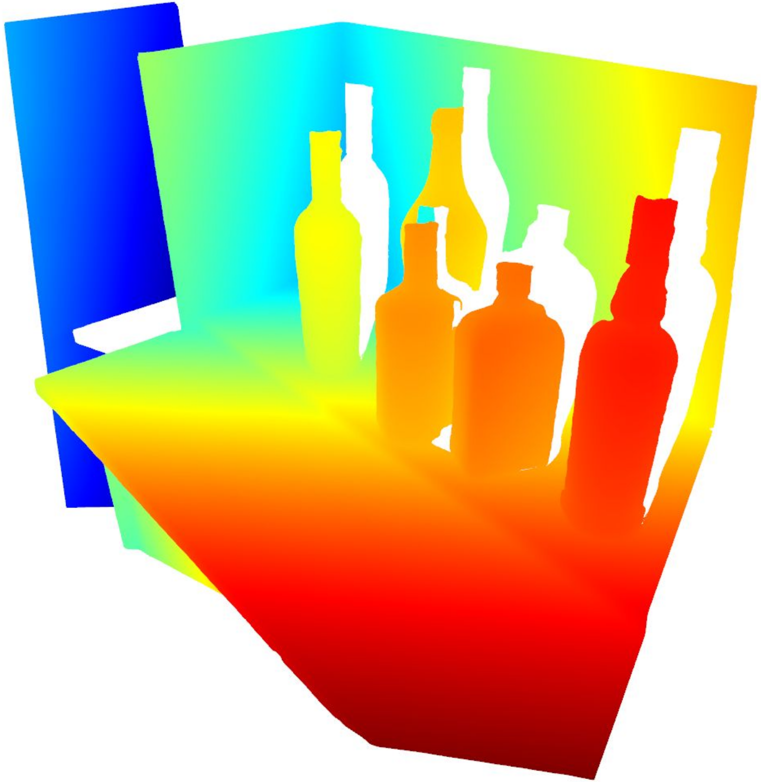
Matteo Poggi, Fabio Tosi, Pierluigi Zama Ramirez
Computer Vision Lab (CVLab), University of Bologna

Sensing 3D Geometry

“The goal of image-based 3D reconstruction is to infer the 3D geometry and structure of objects and scenes from one or multiple 2D images ”

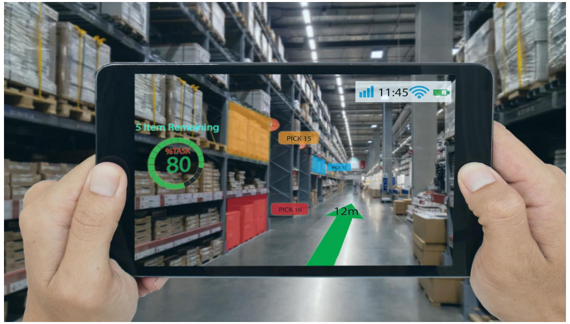
Sensing 3D Geometry

Input Images

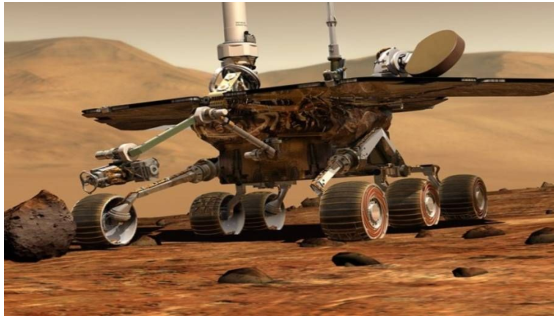

3D Reconstruction

Sensing 3D Geometry

Autonomous driving



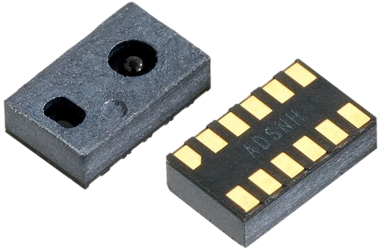
Augmented reality



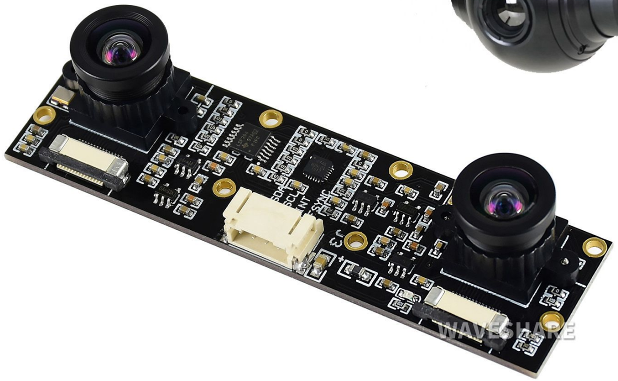
Robotics

Medical applications

Sensing 3D Geometry

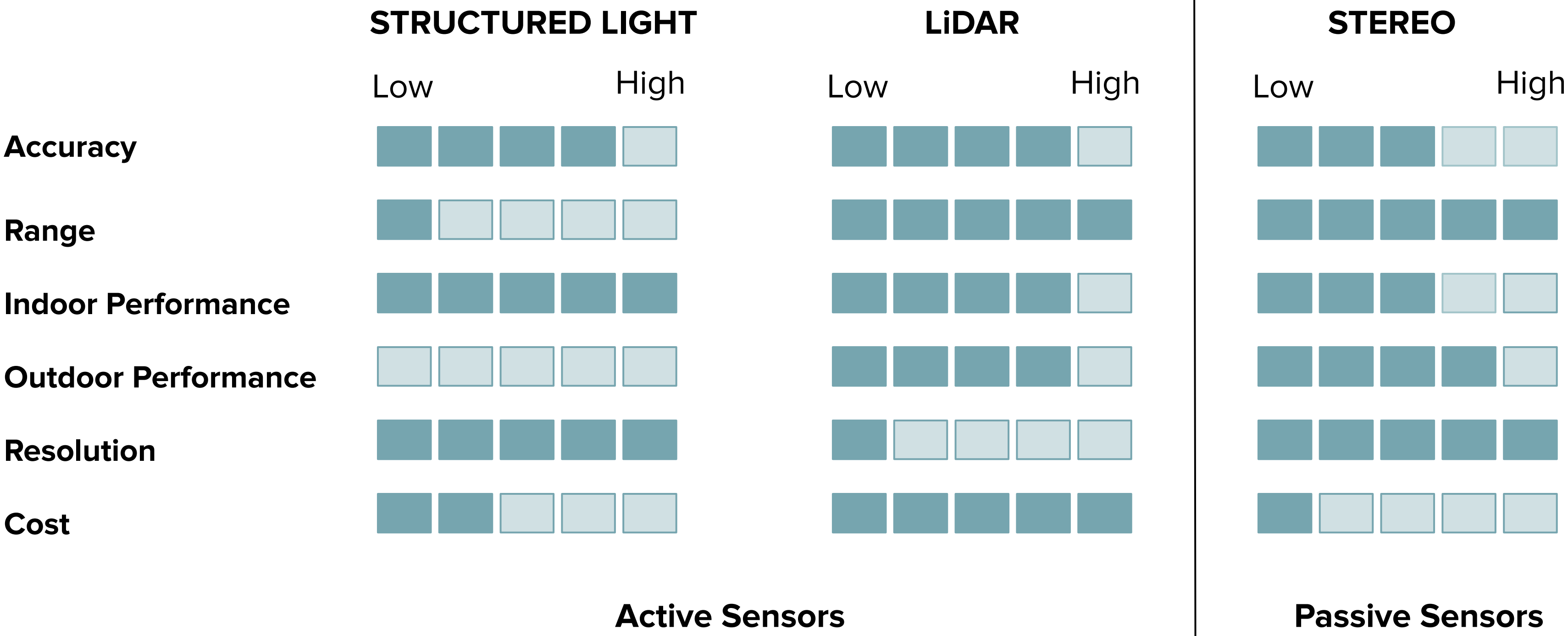


Active Sensors

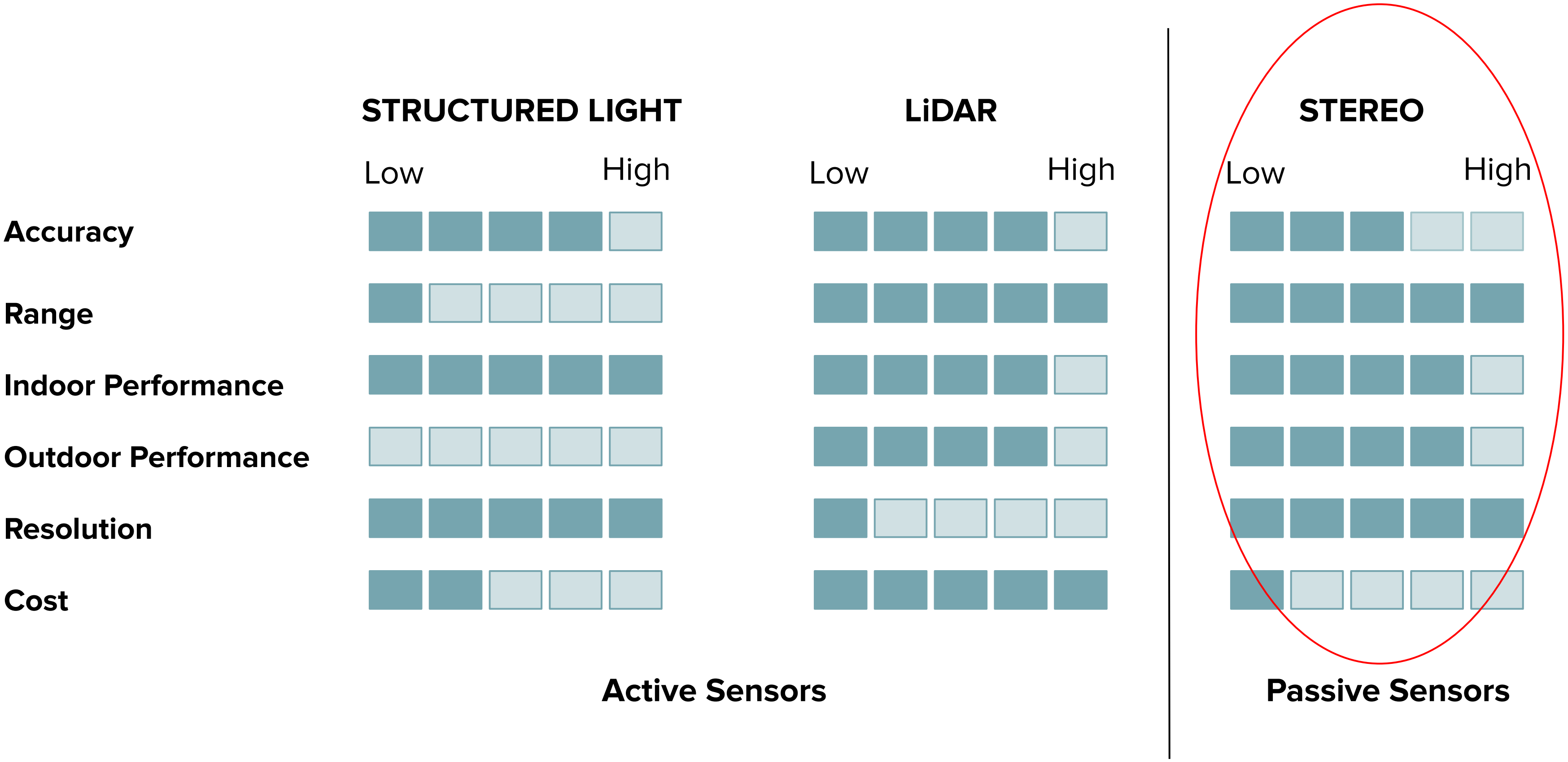


Passive Sensors

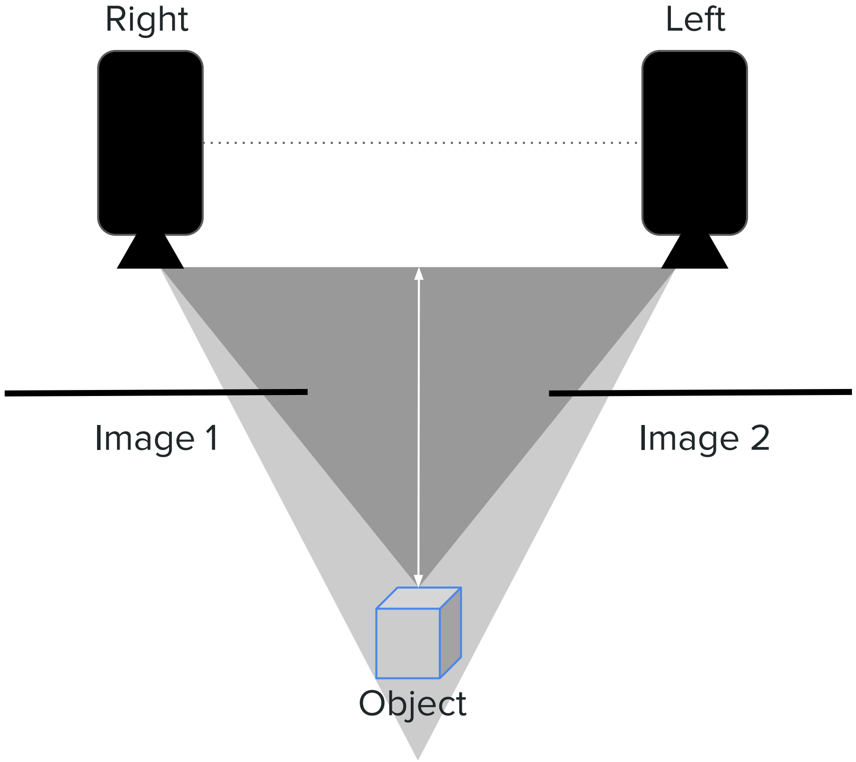
Depth Sensors - Overview



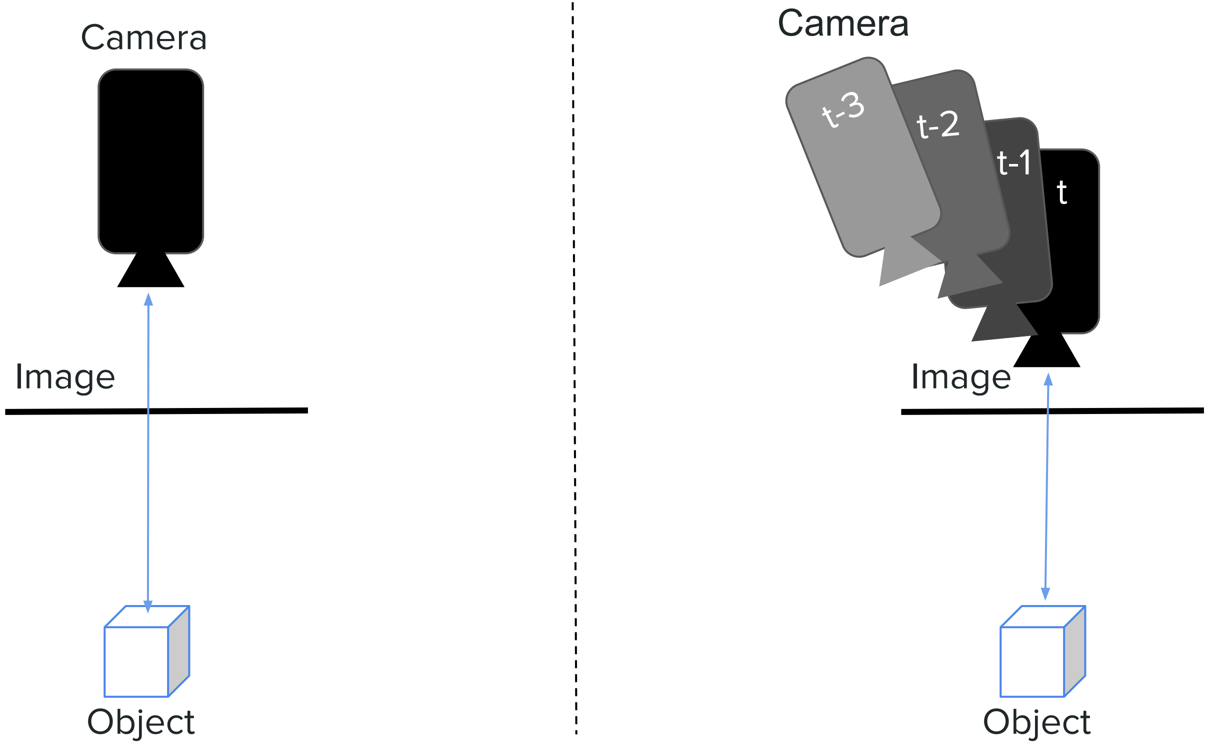
Depth Sensors - Overview



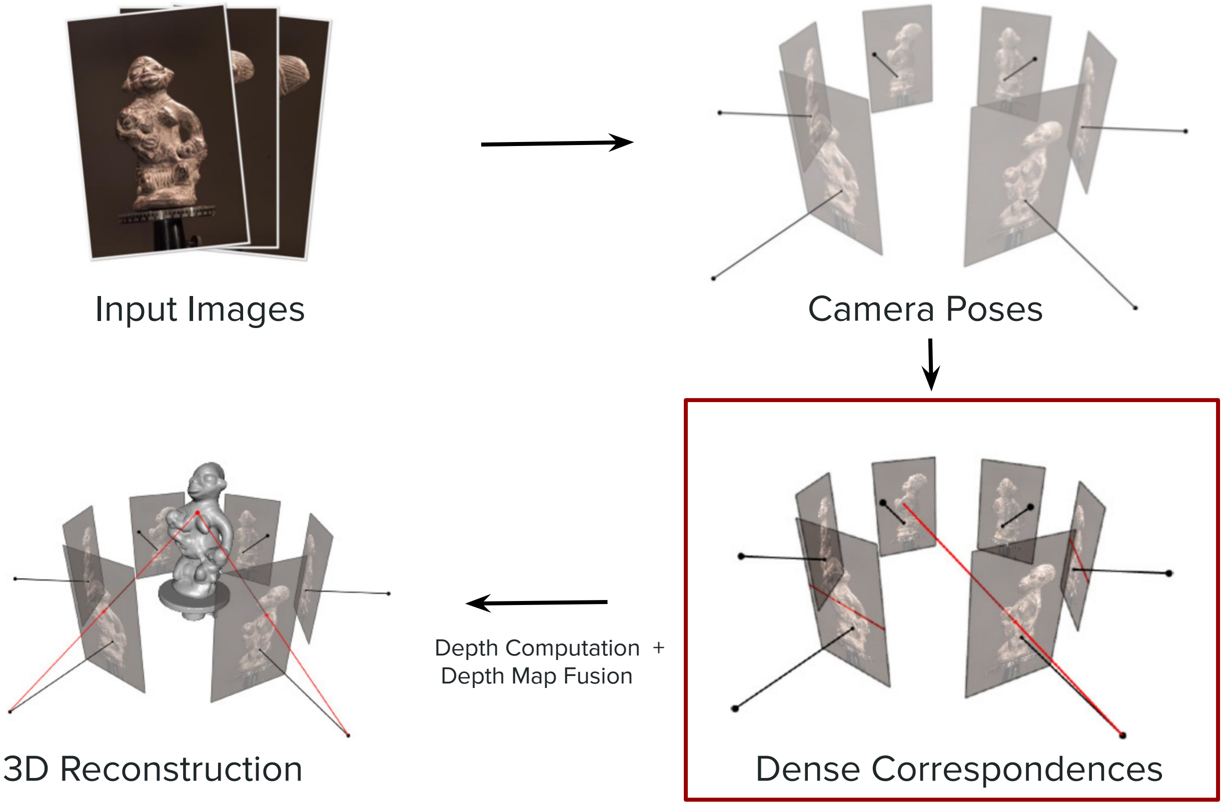
Stereo Setup



Monocular Setup - Single/Multi-view



3D Reconstruction Pipeline



Two-View Stereo Matching

Task:

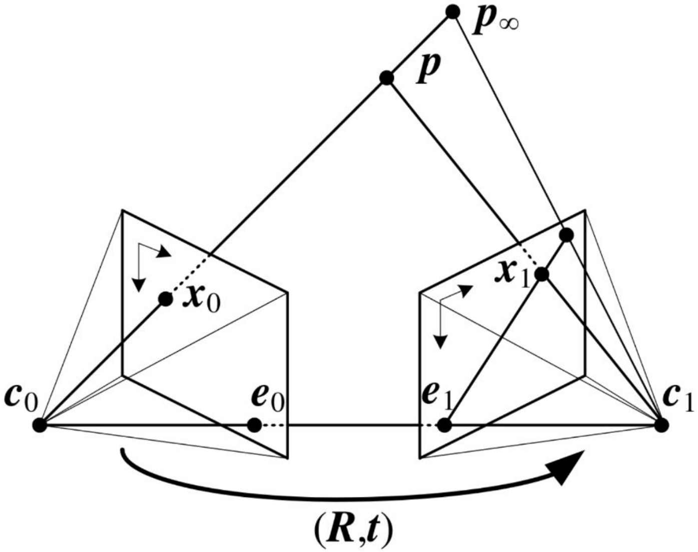
- Construct a **dense** 3D model from 2D images of a static scene (syncrhonized cameras)

Pipeline:

1. **Calibrate cameras** intrinsically and extrinsically
2. **Rectify images** given the calibration
3. **Compute disparity map** for reference image (e.g. left image)
4. **Remove outliers** using consistency/occlusion test
5. **Obtain depth** from disparity using camera calibration
6. **Construct 3D model**, e.g, via volumetric fusion and meshing

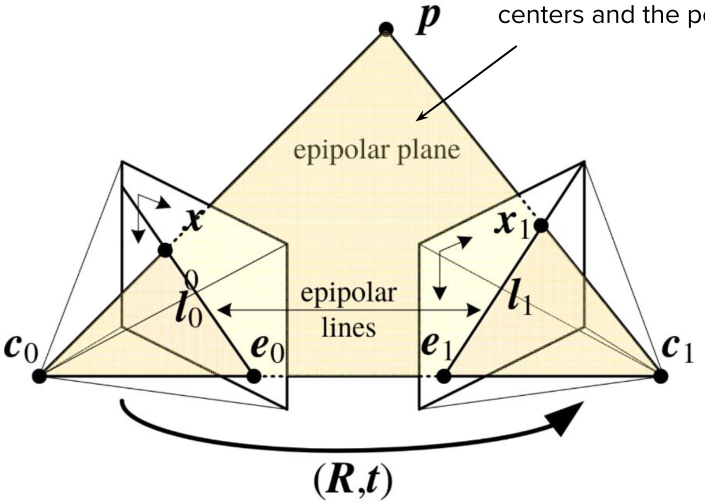
Epipolar Geometry

- Epipolar geometry is used to describe geometric relations in image pairs



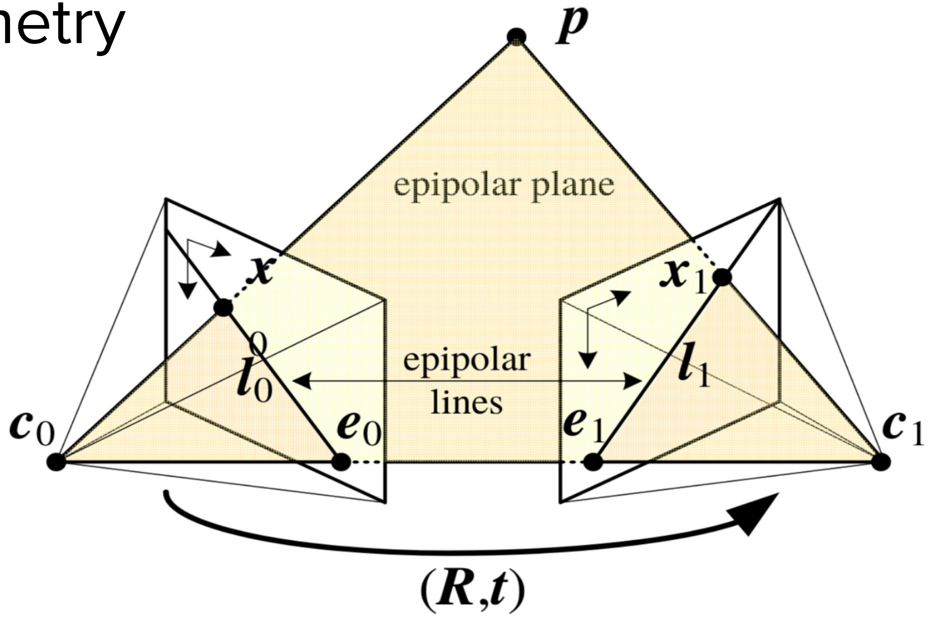
Epipolar line segment
corresponding to one ray

the epipolar plane
depends on the projection
centers and the point



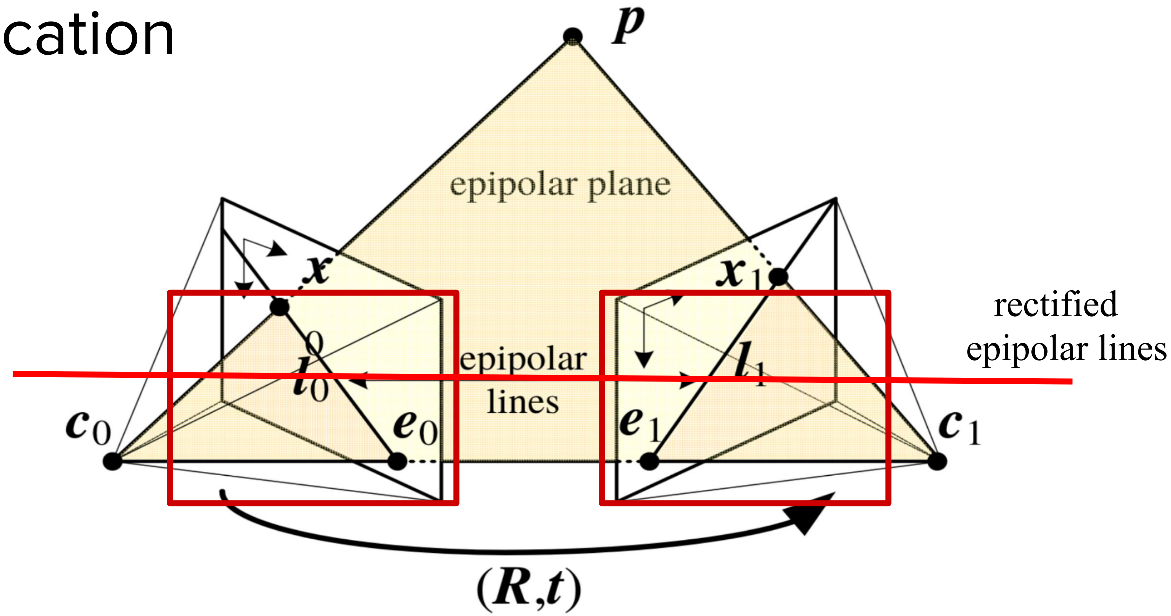
Corresponding set of epipolar
lines and their epipolar plane

Epipolar Geometry



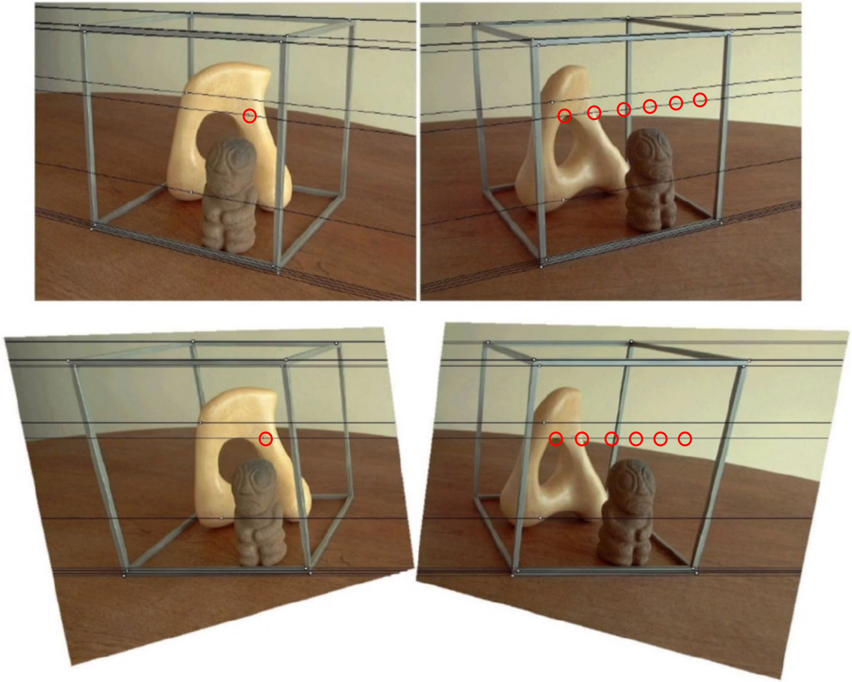
- A point in the first image must be located on the epipolar line in the right image
- This reduces correspondence search to a much simpler **1D problem**
- For VGA images: $\sim$640 instead of $\sim$300k hypotheses (factor 480 less)

Image Rectification



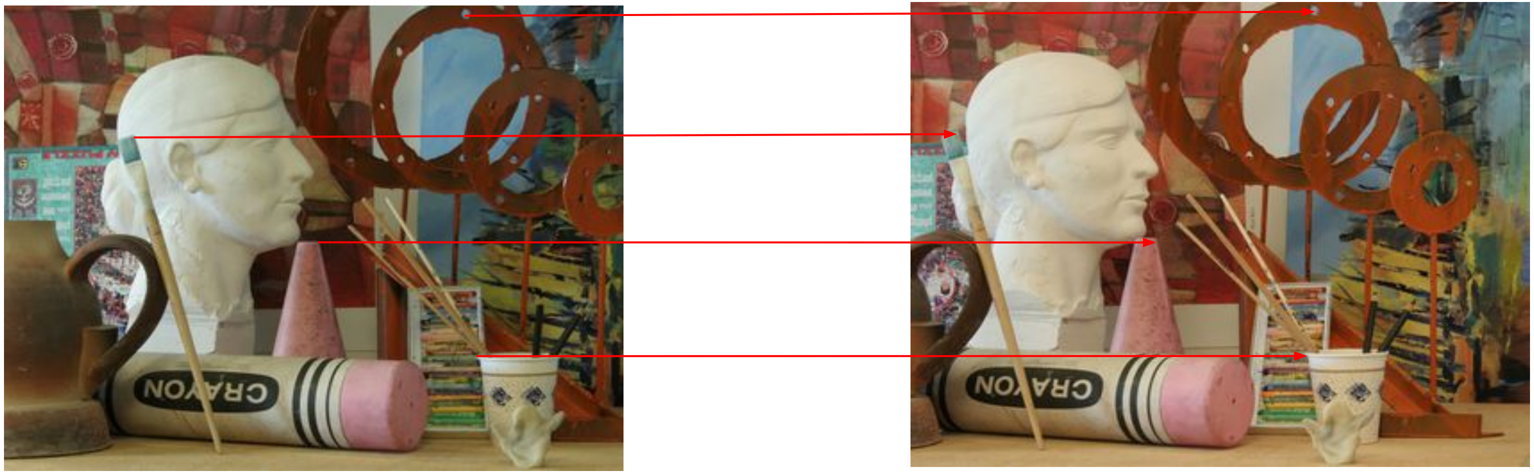
- Reproject image planes onto a common plane parallel to the line between camera centers
- The transformation can be expressed by a rotation around the optical center and an update of the focal length (the 3D structure must not be known).
- Image planes are coplanar $\Rightarrow$ Epipoles at infinity, epipolar lines parallel
- Correspondence search along **horizontal scanlines**

Rectification Example



- Correspondences are located on the **same image row** as the query point

Disparity Estimation Example

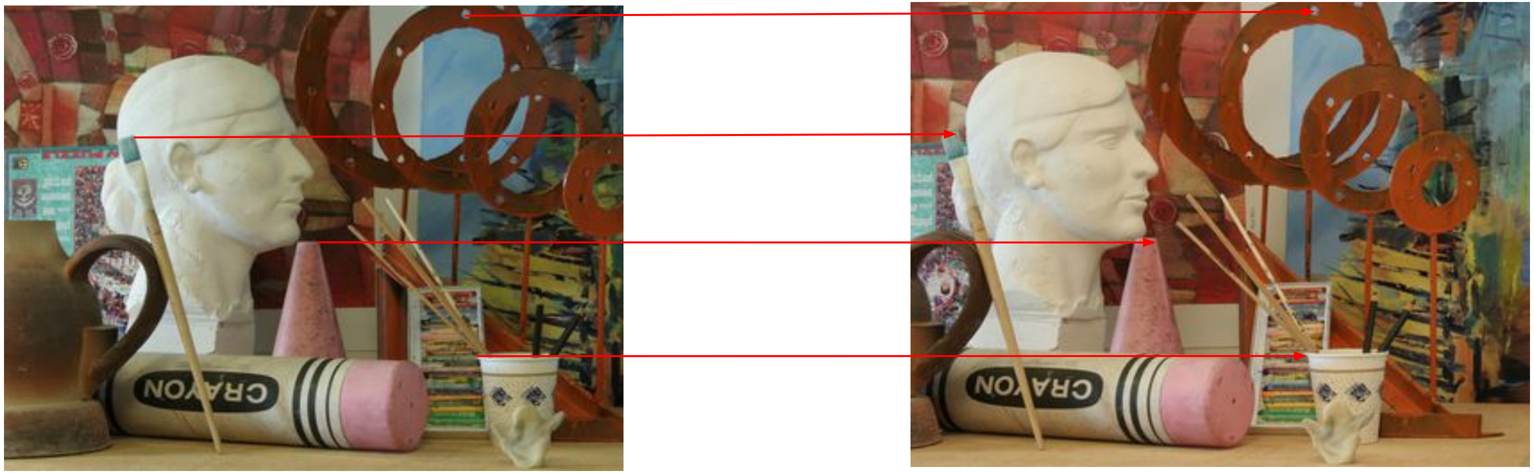


Left Image

Right Image

- **Disparity** refers to the difference in horizontal location of an object in the left and right image - an object at position (x,y) in the left image appears at position (x-d,y) in the right image

Disparity Estimation Example



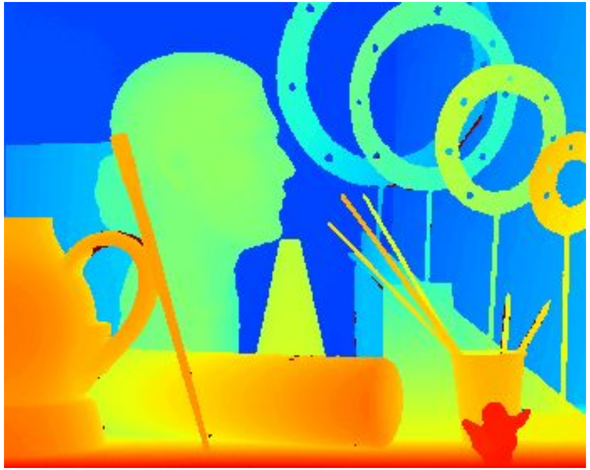
Left Image Right Image

- If we know the disparity of an object we can compute its **depth** using the relation:

$$z = \frac{fB}{d}$$

Disparity Estimation Example

Left Image



Disparity Map

- Warmer colors represent larger values of disparity (and smaller values of depth)

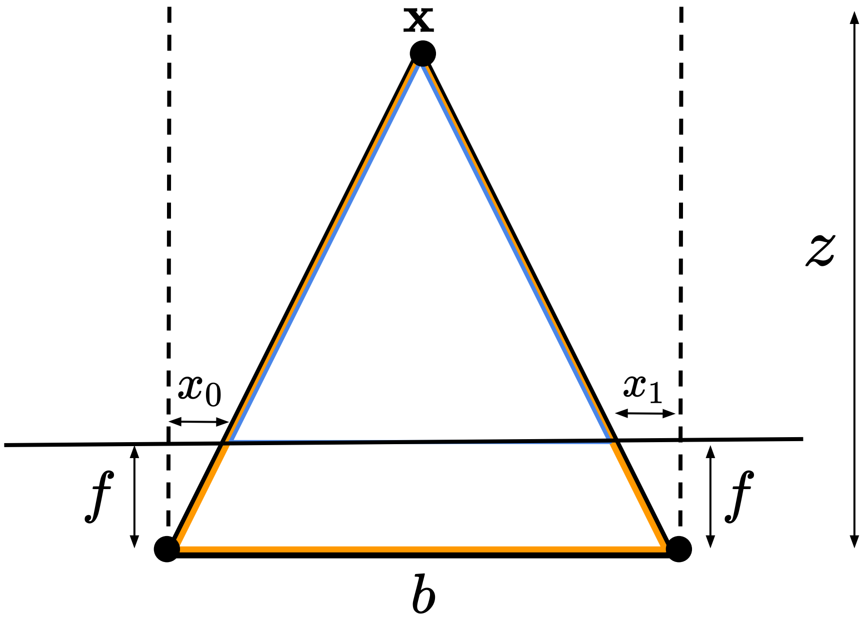
Disparity to Depth

- The left and right ray must intersect as both lie in the epipolar line
- Assuming disparity $d = x0 - x1$ with $x_0 > 0$ and $x_1 < 0$, we have

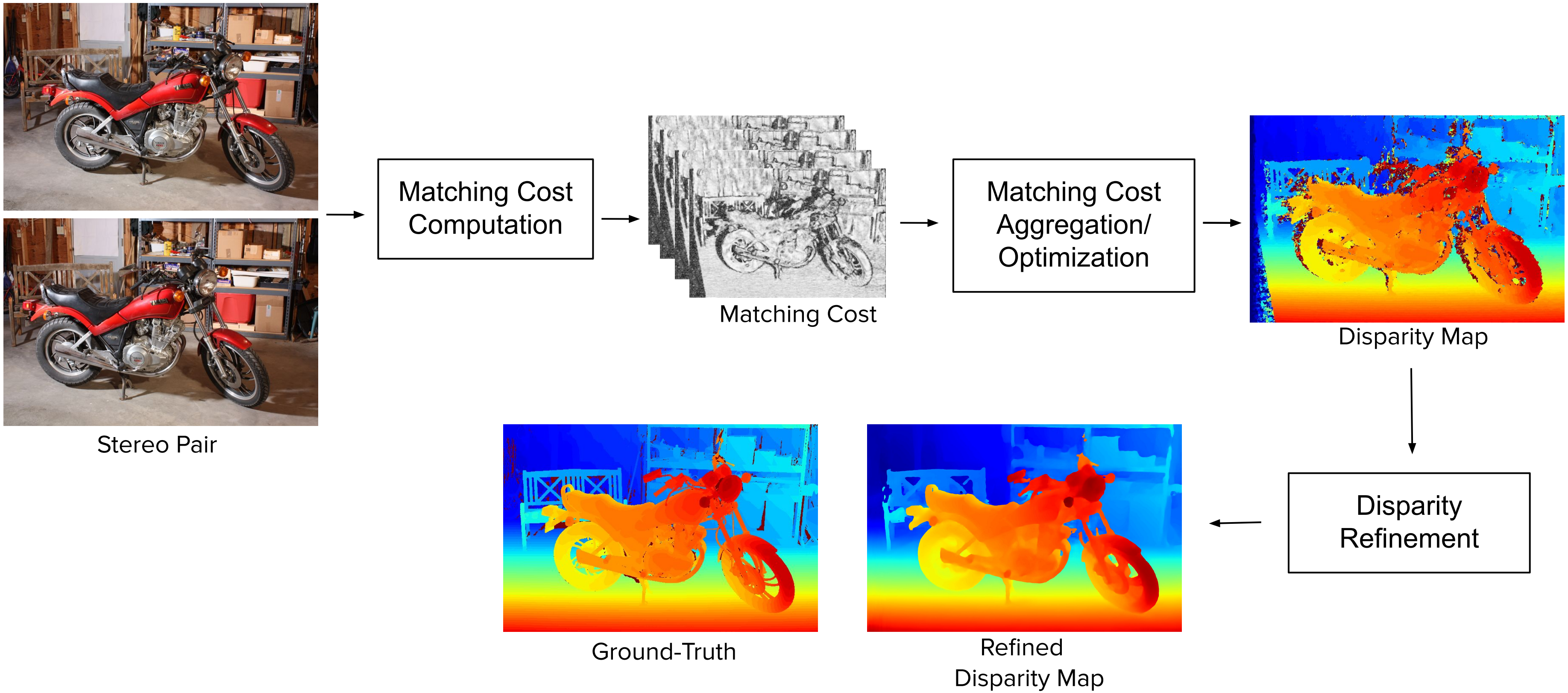
$$\frac{z-f}{b-d} = \frac{z}{b}$$

$$zb - fb = zb - zd$$

$$z = \frac{fb}{d}$$



Traditional Stereo Matching Pipeline



Similarity Metrics

Left Image

Right Image

- Radiometric **differences** often occur due to different imaging characteristics of the camera due to: different exposure time, non-lambertian reflection which is view-dependent etc.

Similarity Metrics

Left Image

Similarity Metrics

Right Image

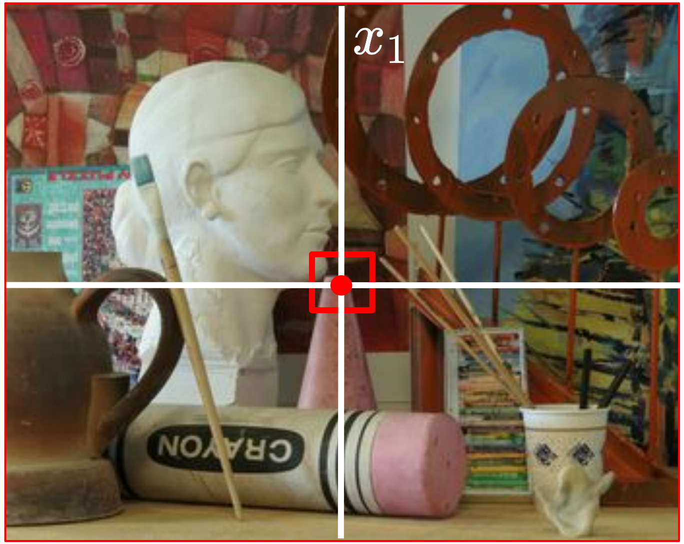
Similarity Metrics

Left Image

Right Image

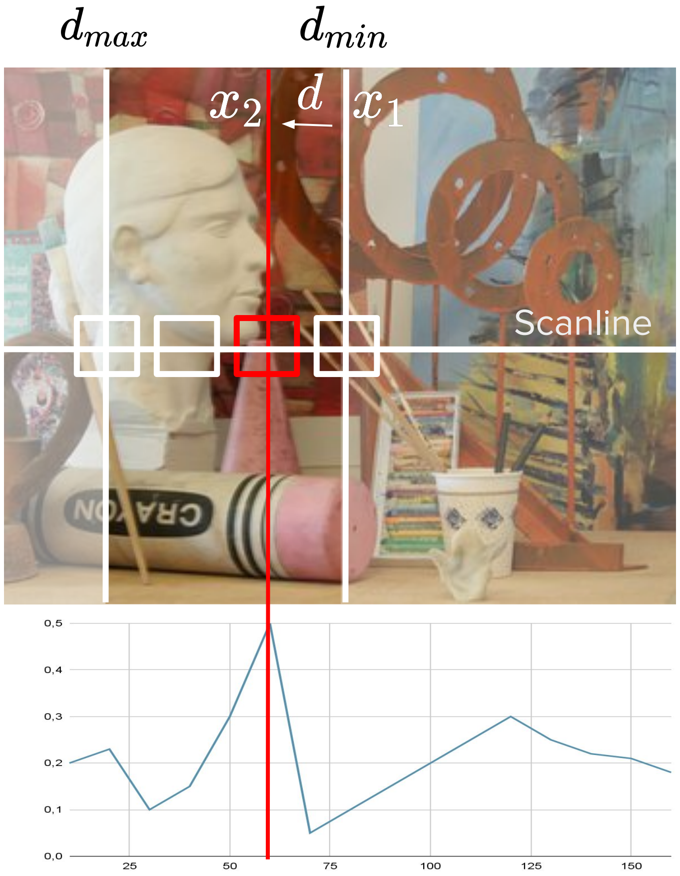
- How to determine if two image points correspond?
- A **single pixel** does not reveal the local structure (too many ambiguities)
- Therefore, we should compare at least a **small region/patch**

Similarity Metrics



Left Image

- Center a small window on a pixel and **match** the whole window in the right image

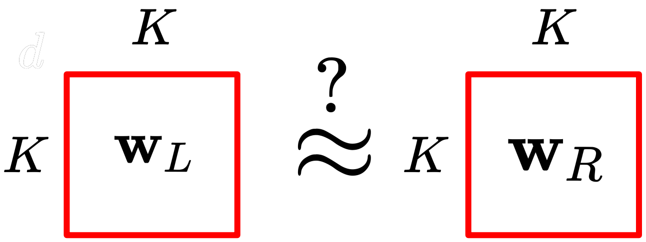


Matching Score

Similarity Metrics

Left Image

Right Image



- Consider two $K \times K$ windows of pixels flattened to vectors $\mathbf{w}_L, \mathbf{w}_R \in \mathbb{R}^{K^2}$
- Sum of squared difference (SSD):

$$SSD(x, y, d) = ||\mathbf{w}_L(x, y) - \mathbf{w}_R(x - d, y)||_2^2$$

Numerous other similarity metrics exist (NCC, Census+Hamming Distance)

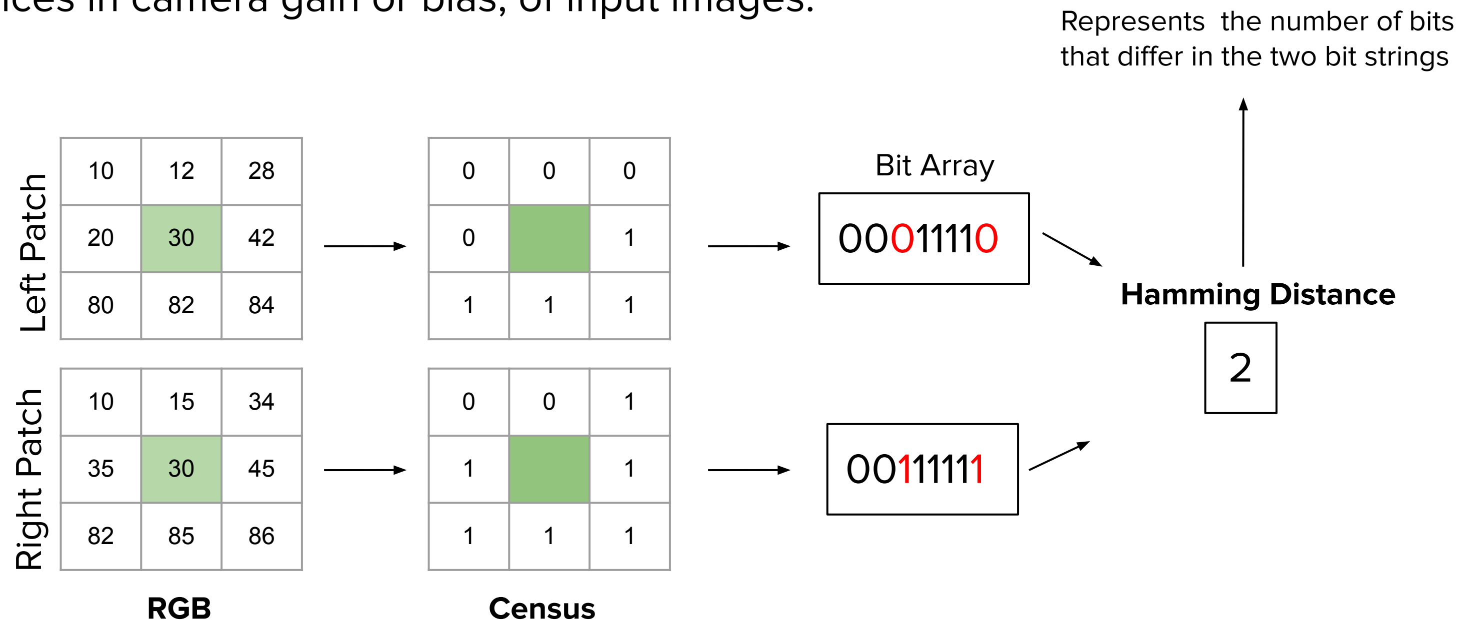
Similarity Metrics - Census Transform

- The **census transform (CT)** is an image operator that associates to each pixel of a grayscale image a binary string (that depends on a window around the pixel)
- Since the census transform uses the **relative intensity** of input images, it is **insensitive** to differences in camera gain or bias, of input images.

$$\xi(p, p') = \begin{cases} 0 & \text{if } p > p' \\ 1 & \text{if } p \le p' \end{cases}$$

Similarity Metrics - Census Transform

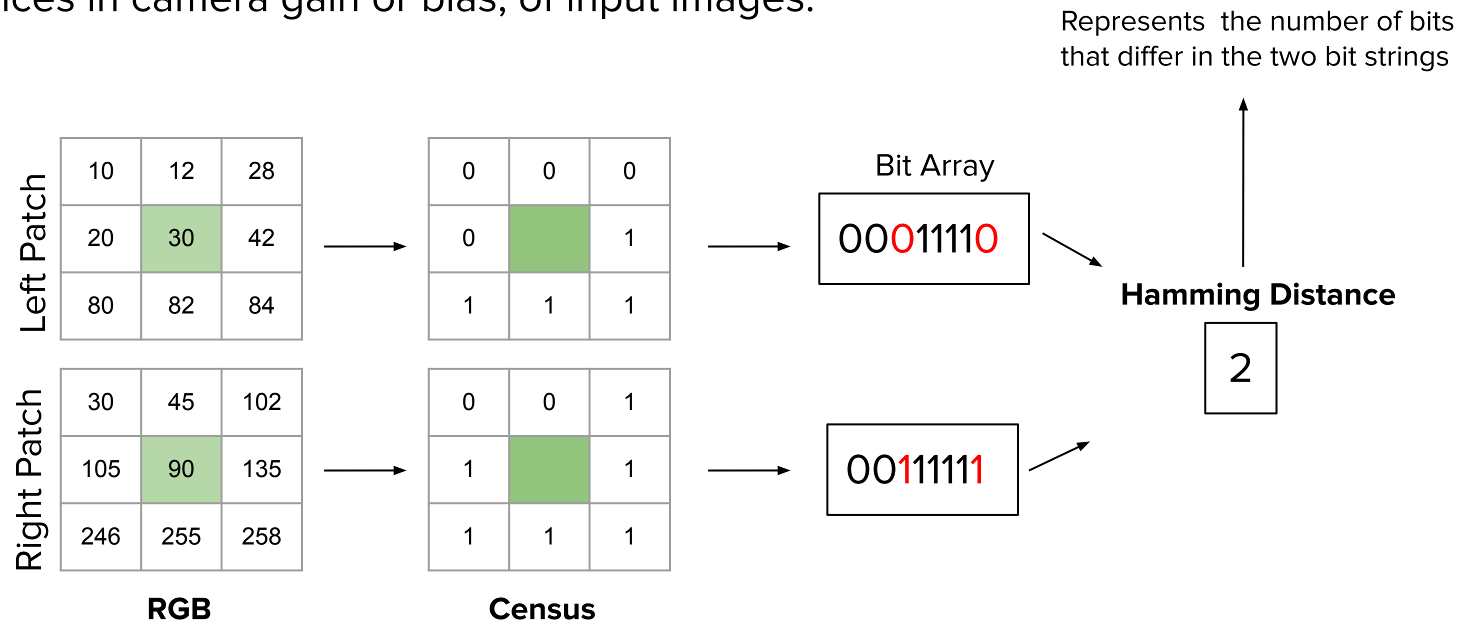
- The **census transform (CT)** is an image operator that associates to each pixel of a grayscale image a binary string (that depends on a window around the pixel)
- Since the census transform uses the **relative intensity** of input images, **insensitive** to differences in camera gain or bias, of input images.



Similarity Metrics - Census Transform

- The **census transform (CT)** is an image operator that associates to each pixel of a grayscale image a binary string (that depends on a window around the pixel)
- Since the census transform uses the **relative intensity** of input images, **insensitive** to differences in camera gain or bias, of input images.

What if we multiply an image by 3?



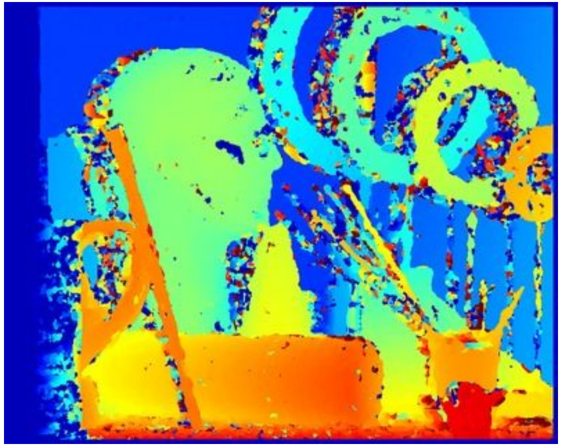
Similarity Metrics - Census Transform

Left Image

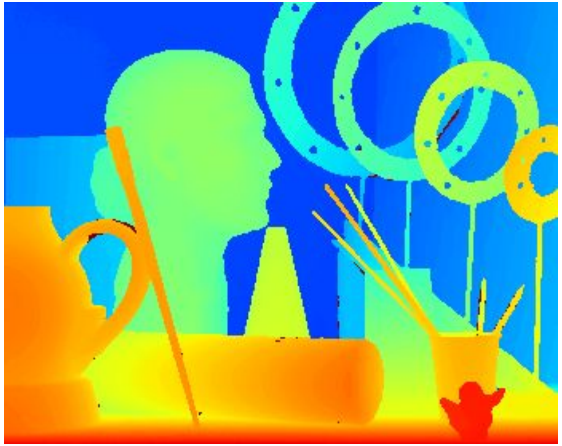
Left Image - Census (3x3)

Block Matching

Left Image



Disparity Map



Ground Truth

- Choose disparity range $[0, D]$
- For all pixels $\mathbf{x} = (x, y)$ compute the best disparity $\Rightarrow$ **Winner-Takes-All** (WTA)
- (optionally) Do this for both images and apply left-right consistency check to remove outliers

When will local matching fail?

The Underlying Assumption

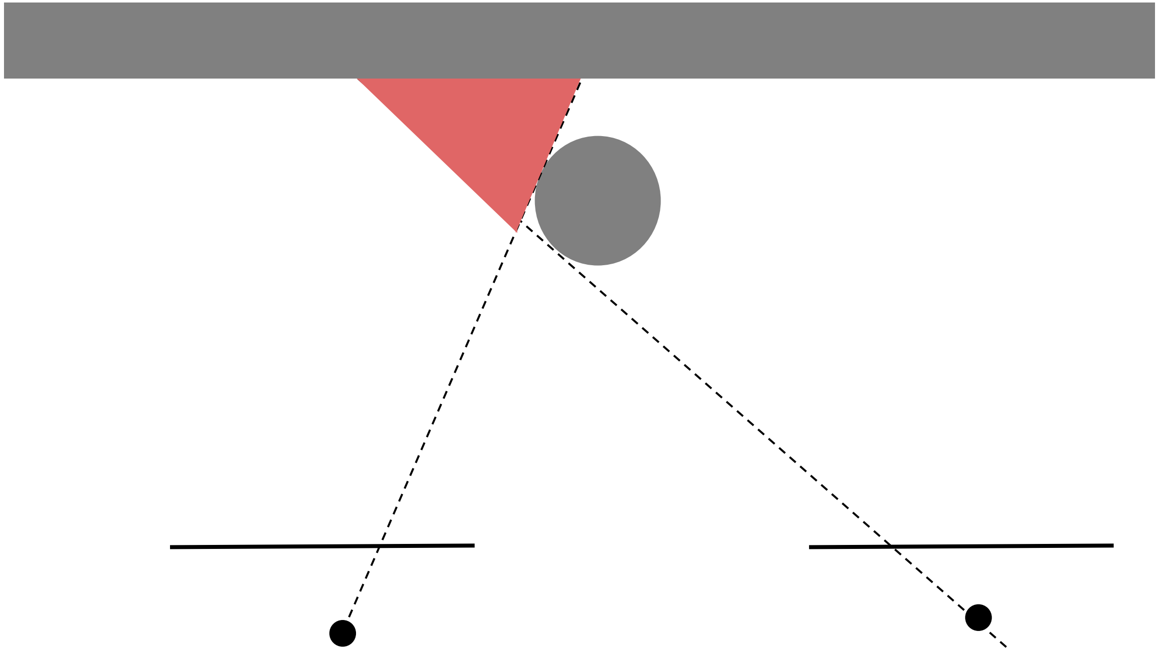
- Corresponding regions in both images should look similar
- Non-corresponding regions should look different
- When will this similarity constraint fail?

The Underlying Assumption

- Corresponding regions in both images should look similar
- Non-corresponding regions should look different
- When will this similarity constraint fail?

Similarity Constraint: Failure Cases

Block Matching: Occluded Regions



- The red area is visible in the left image, but **not** in the right image
- For occluded pixels there exists **no correspondence** (we cannot estimate disparity)

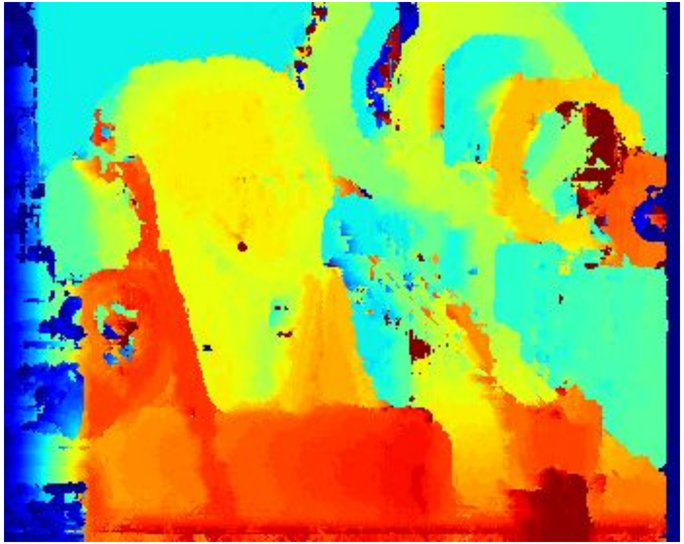
Left-Right Consistency Check

Left Image

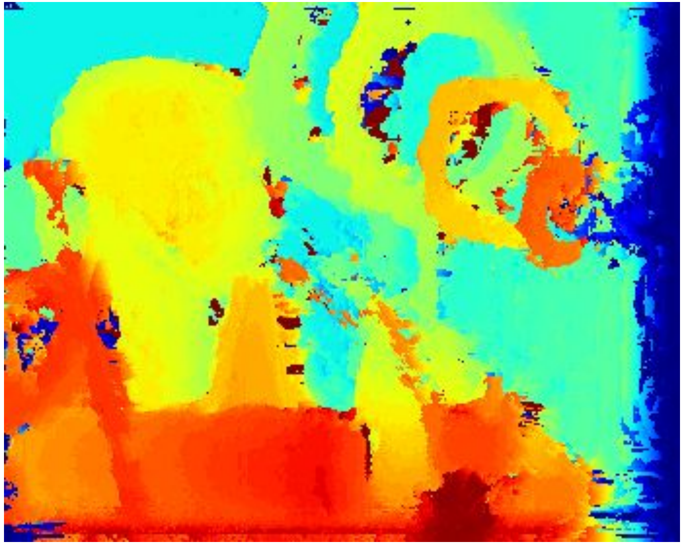
Right Image

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for **both** images, verify if they map to each other

Left-Right Consistency Check



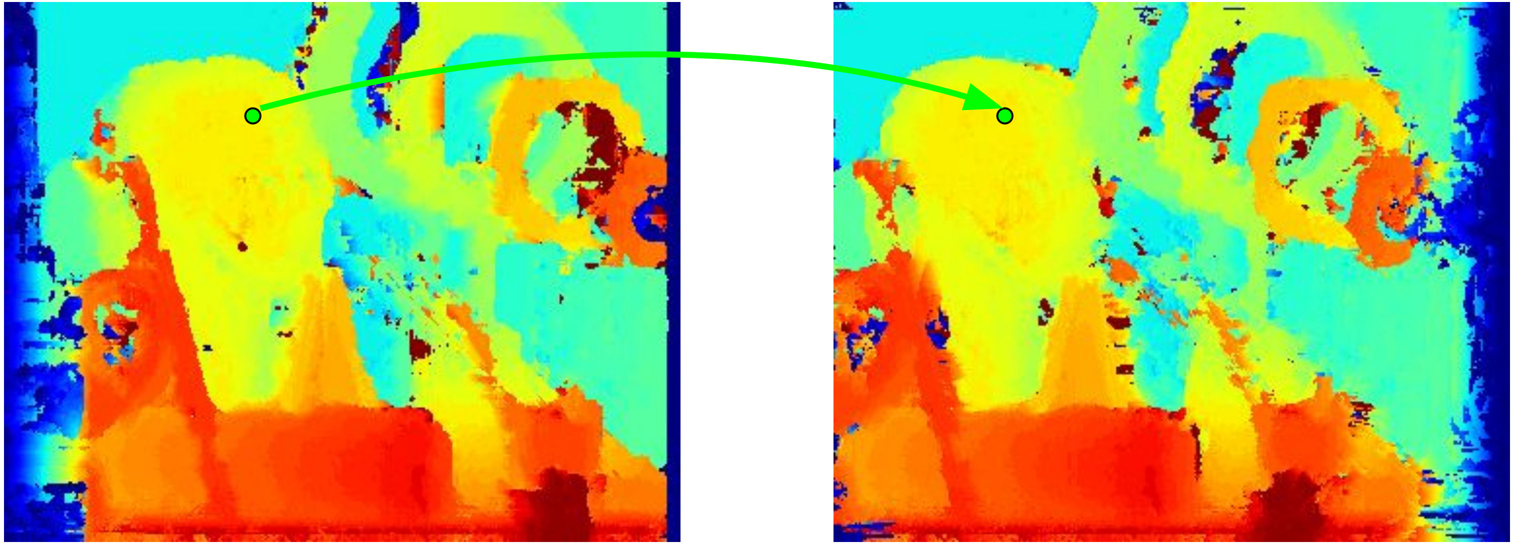
Left Disparity



Right Disparity

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Left-Right Consistency Check

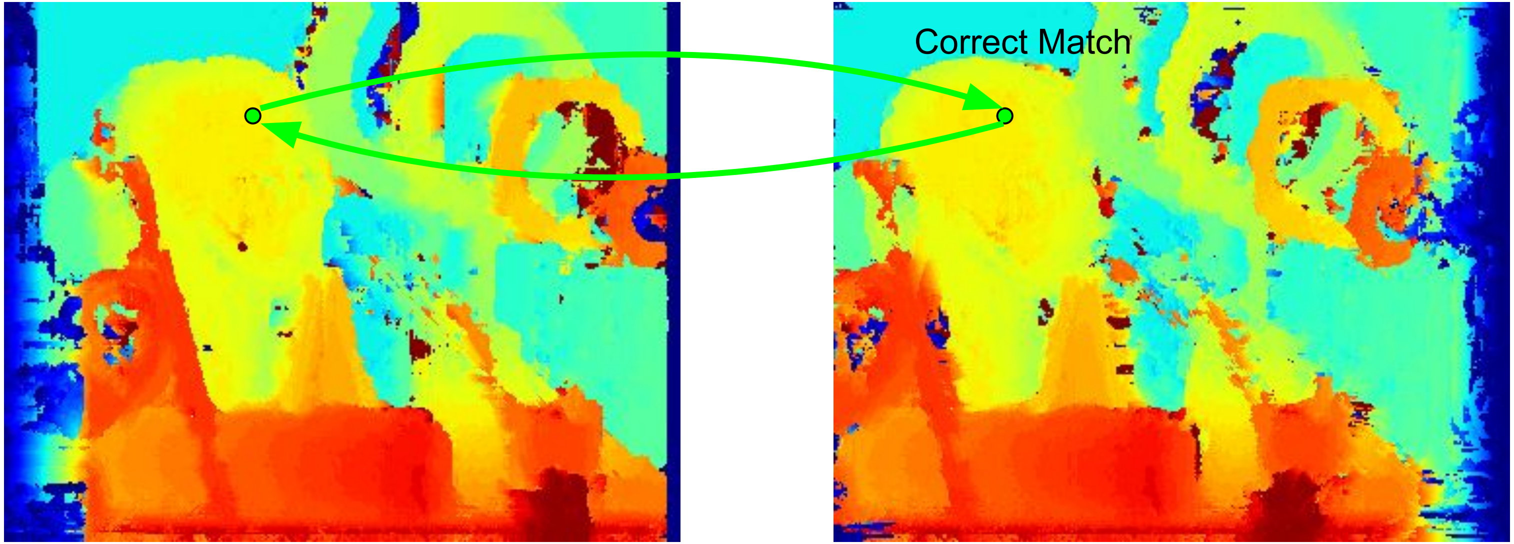


Left Disparity

Right Disparity

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Left-Right Consistency Check

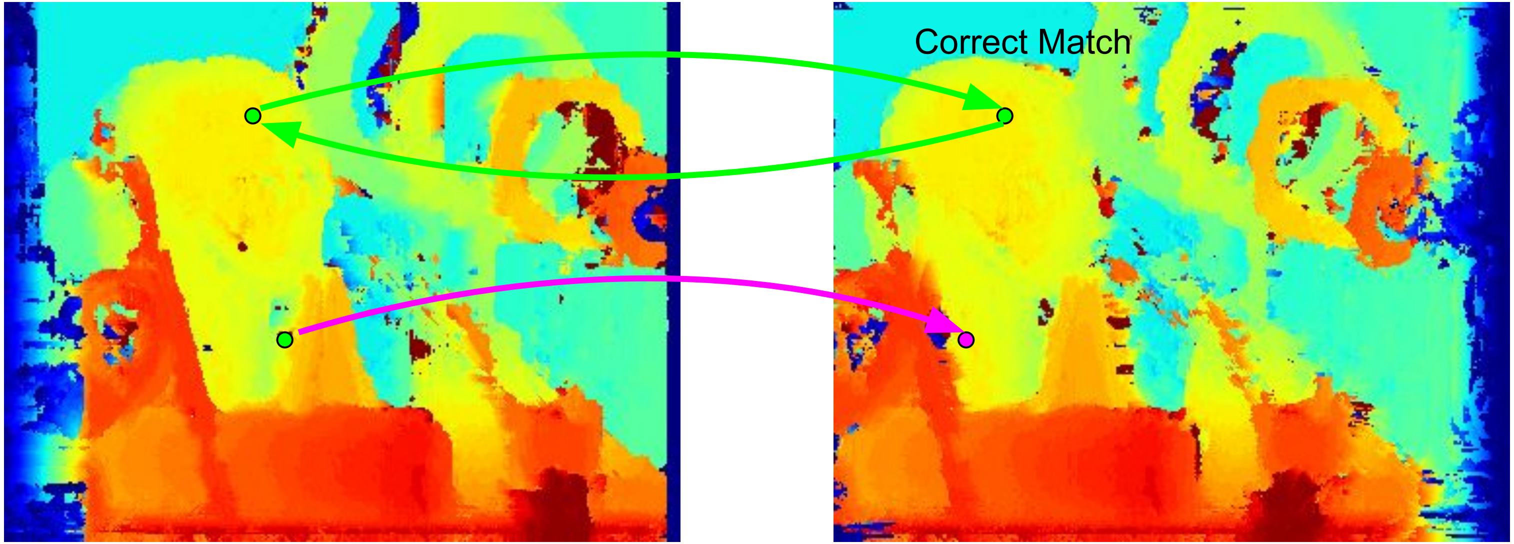


Left Disparity

Right Disparity

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Left-Right Consistency Check

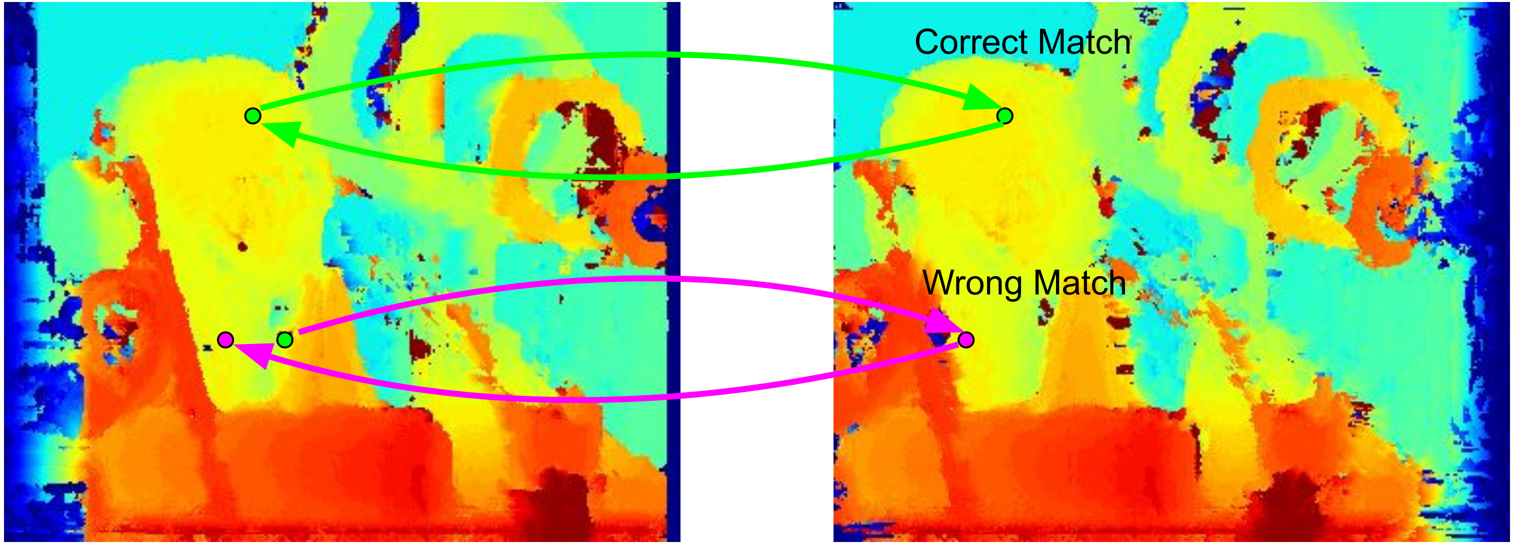


Left Disparity

Right Disparity

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Left-Right Consistency Check

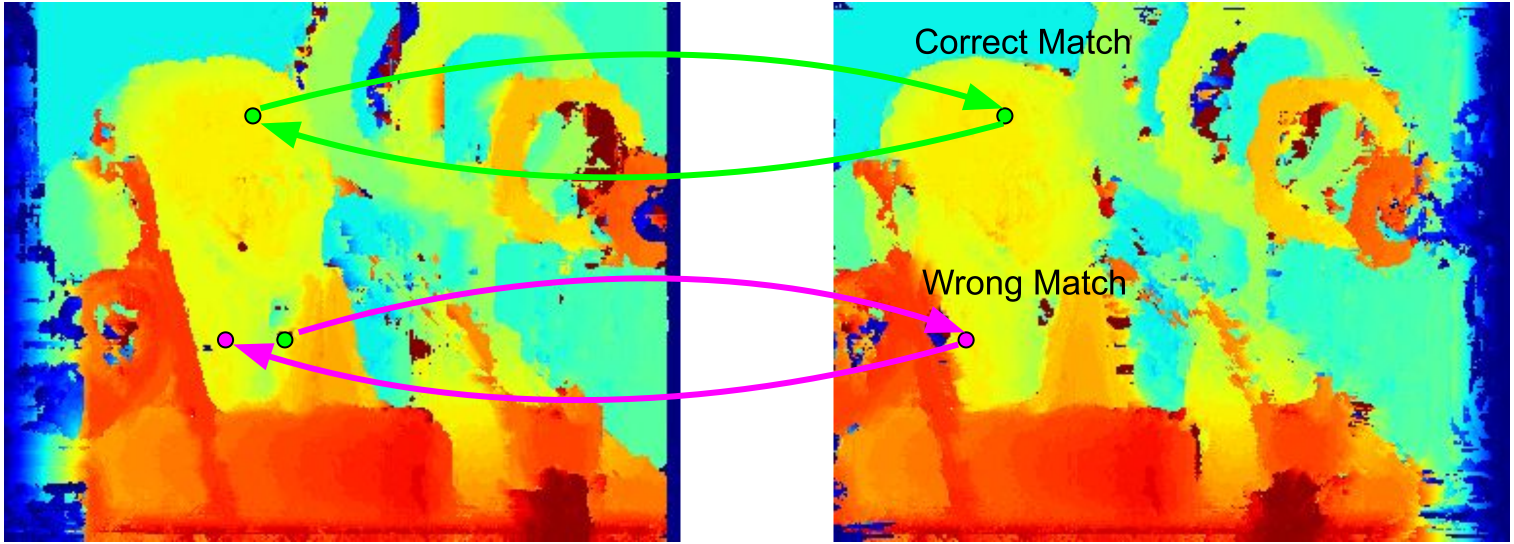


Left Disparity

Right Disparity

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Left-Right Consistency Check

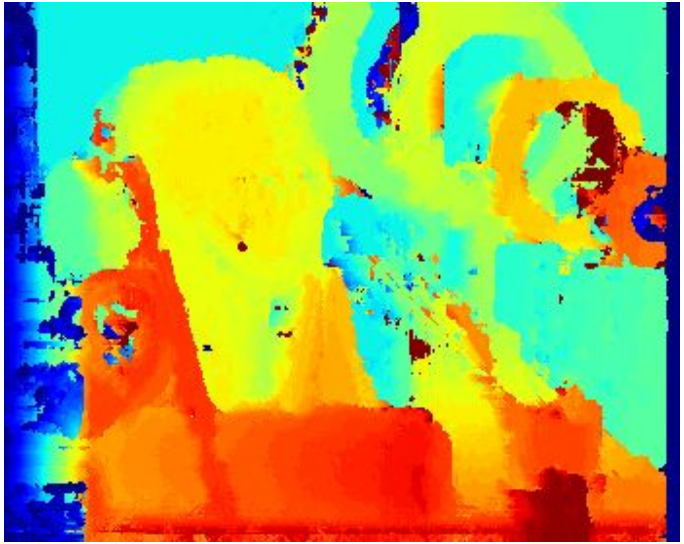


Left Disparity

Right Disparity

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Left-Right Consistency Check



Disparity w/o LRC

Disparity with LRC

- Outliers and occlusions can be detected via a **left-right consistency check**
- Compute disparity map for both images, verify if they map to each other

Block Matching: Assumption Violation

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** which is often invalid (valid only for 3D planes that are parallel to the image plane)
- **Slanted surfaces** deform perspectively when the viewpoint changes

Block Matching: Assumption Violation

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** which is often invalid (valid only for 3D planes that are parallel to the image plane)
- **Slanted surfaces** deform perspectively when the viewpoint changes

Block Matching: Assumption Violation

Left Image

Left Image Patch

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** which is often invalid (valid only for 3D planes that are parallel to the image plane)
- The window content changes differently at **disparity discontinuities**

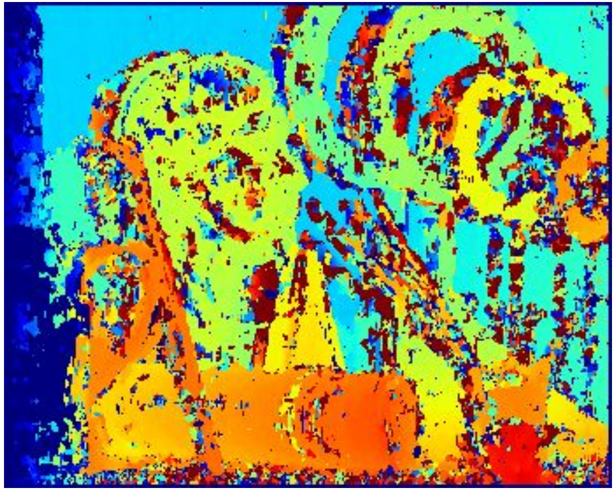
Block Matching: Assumption Violation

Right Image

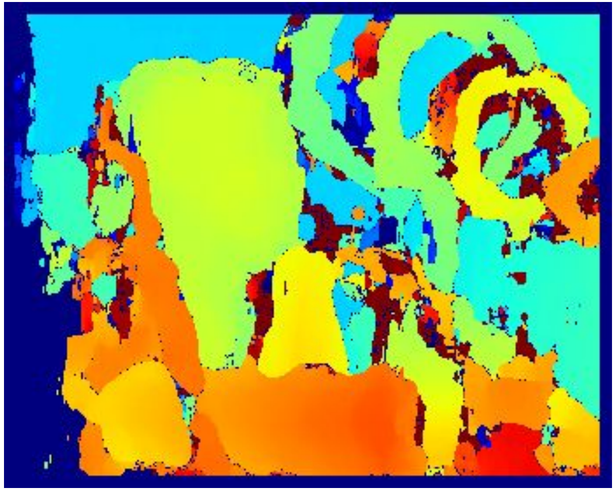
Right Image Patch

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** which is often invalid (valid only for 3D planes that are parallel to the image plane)
- The window content changes differently at **disparity discontinuities**

Effect of Window Size



Window Size: 5×5

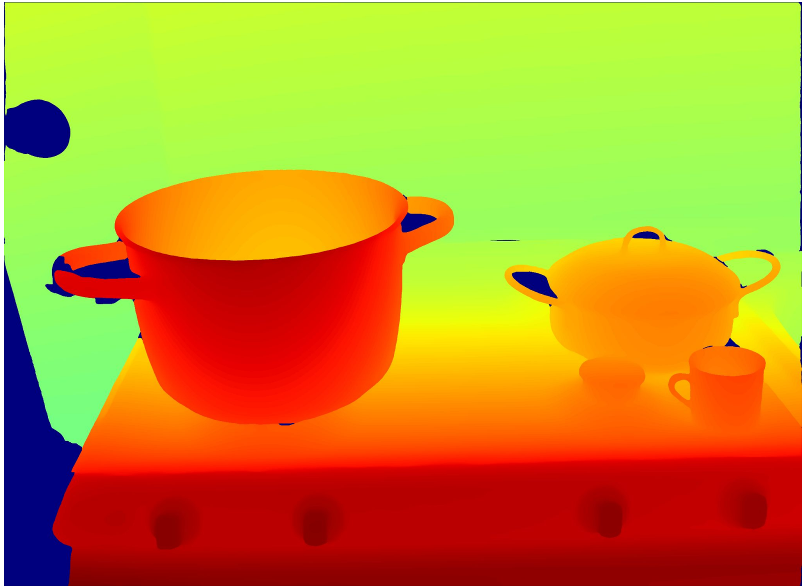


Window Size: 15×15

Tradeoff (there is no optimal window size that can handle all these problems at once)

- **Small windows** lead to matching ambiguities and noise in the disparity maps
- **Larger windows** lead to smoother results, but loss of details (better for untextured regions and repetitive patterns)

How does the real world look like?



- Depth varies **slowly** except at object discontinuities

Stereo Processing by Semiglobal Matching and Mutual Information (Hirschmuller, 2005)

- Find the best disparity map that minimizes the following **global 2D energy function**

$$
\begin{aligned}
\mathbb{E}(D) = \sum_{\mathbf{x}} \Big(C(\mathbf{x}, d^{\mathbf{x}}) &+ \sum_{\mathbf{y} \in \mathbf{N}_{\mathbf{x}}} P_1 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| = 1] \\
&+ \sum_{\mathbf{y} \in \mathbf{N}_{\mathbf{x}}} P_2 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| > 1] \Big)
\end{aligned}
$$

Stereo Processing by Semiglobal Matching and Mutual Information (Hirschmuller, 2005)

- Find the best disparity map that minimizes the following **global 2D energy function**

$$\mathbb{E}(D) = \sum_{\mathbf{x}} \Big(\underbrace{C(\mathbf{x}, d^{\mathbf{x}})}_{\text{Data Term}} + \sum_{\mathbf{y} \in \mathbf{N_x}} \underbrace{P_1 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| = 1]}_{\text{Smoothness Term (Slanted Surfaces)}} + \sum_{\mathbf{y} \in \mathbf{N_x}} \underbrace{P_2 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| > 1]}_{\text{Smoothness Term (Depth Discontinuities)}} \Big)$$

Smoothness terms that penalizes
disparity differences between neighboring pixels

Stereo Processing by Semiglobal Matching and Mutual Information (Hirschmuller, 2005)

- Find the best disparity map that minimizes the following **global 2D energy function**

$$\mathbb{E}(D) = \sum_{\mathbf{x}} \Big(C(\mathbf{x}, d^{\mathbf{x}}) + \sum_{\mathbf{y}\in \mathbf{N_x}} P_1 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| = 1] + \sum_{\mathbf{y}\in \mathbf{N_x}} P_2 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| > 1] \Big)$$

Data Term

Smoothness Term (Slanted Surfaces)

Smoothness Term (Depth Discontinuities)

- If parameters have not been properly tuned, the performance of the algorithm may not be as efficient as expected
- Minimizing 2D global minimization is a **NP-complete** problem
- Semi-Global Matching (SGM) idea: perform line optimisation along **multiple directions**

Stereo Processing by Semiglobal Matching and Mutual Information (Hirschmuller, 2005)

- Find the best disparity map that minimizes the following **global 2D energy function**

$$\mathbb{E}(D) = \sum_{\mathbf{x}} \Big(C(\mathbf{x}, d^{\mathbf{x}}) + \sum_{\mathbf{y} \in \mathbf{N}_{\mathbf{x}}} P_1 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| = 1] + \sum_{\mathbf{y} \in \mathbf{N}_{\mathbf{x}}} P_2 T[|d^{\mathbf{x}} - d^{\mathbf{y}}| > 1] \Big)$$

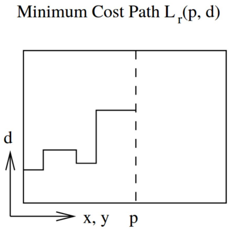
Data Term

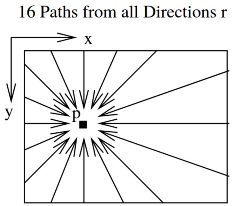
Smoothness Term (Slanted Surfaces)

Smoothness Term (Depth Discontinuities)

- **1D-cost approximation** in each of 8/16 directions (paths)

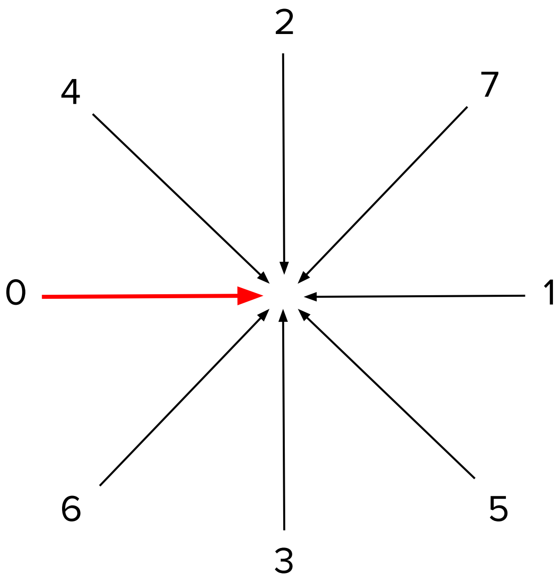
$$L'_{\mathbf{r}}(\mathbf{x}_0, d) = c(\mathbf{x}_0, d) + \min\Big(L'_{\mathbf{r}}(\mathbf{x}_1, d), L'_{\mathbf{r}}(\mathbf{x}_1, d-1) + P_1, L'_{\mathbf{r}}(\mathbf{x}_1, d+1) + P_1, \min_{i \neq d \pm 1} L'_{\mathbf{r}}(\mathbf{x}_1, i) + P_2 \Big).$$



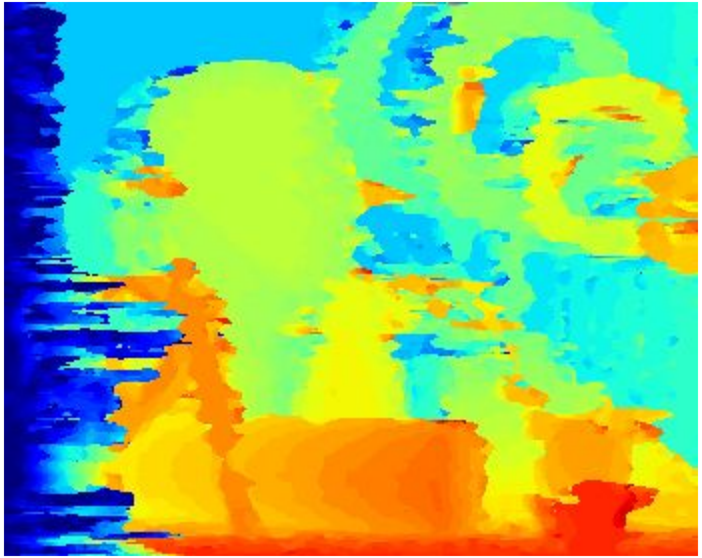


- Advantages: accuracy, computational efficiency, simplicity...but suffers **streaking artifacts**

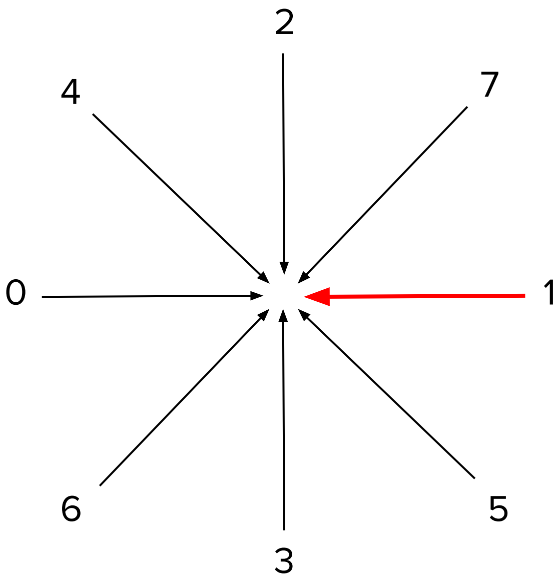
Streaking Artifacts



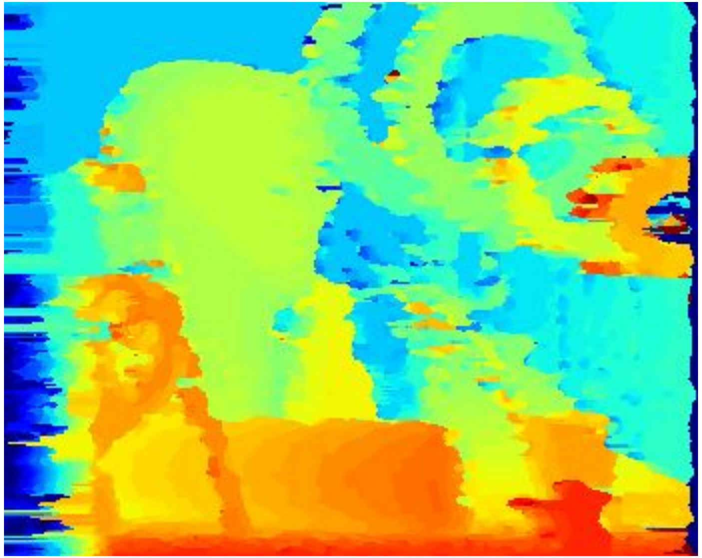
Scanline 0



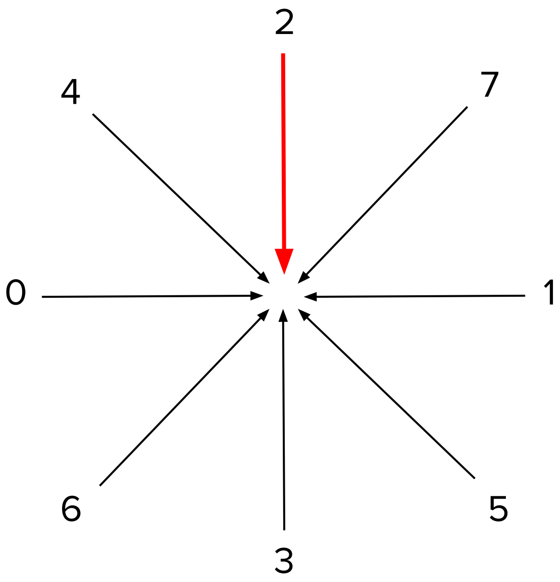
Streaking Artifacts



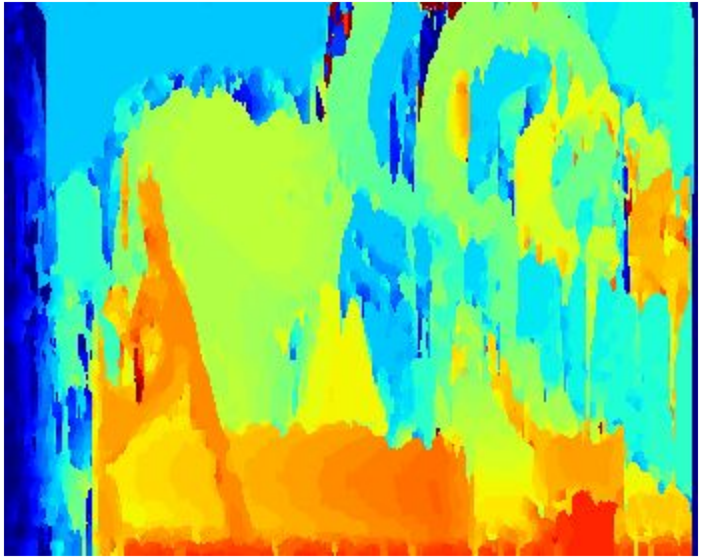
Scanline 1



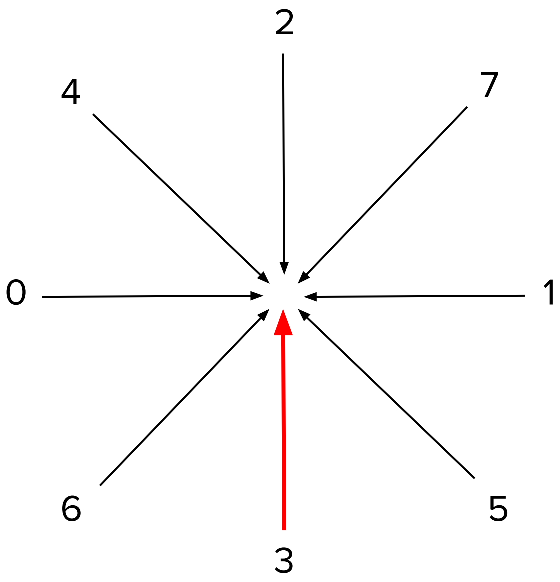
Streaking Artifacts



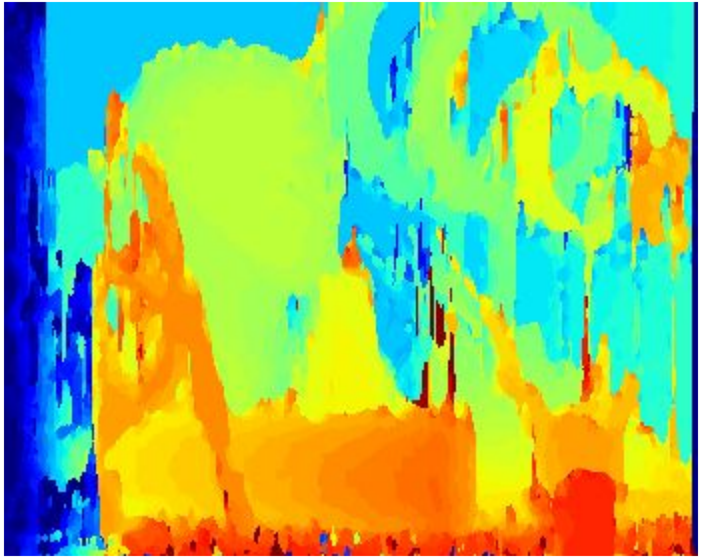
Scanline 2



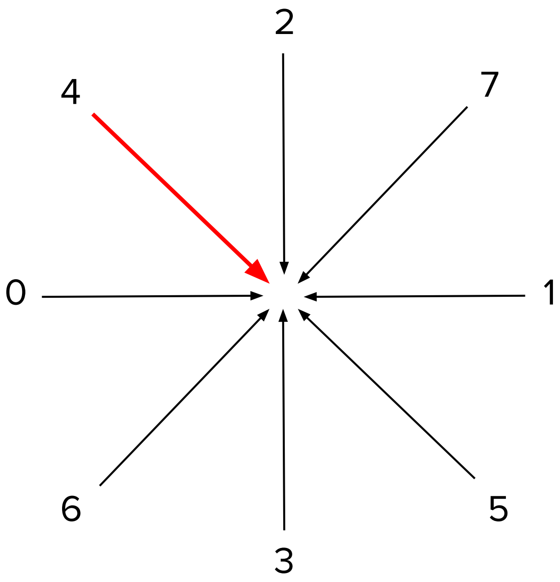
Streaking Artifacts



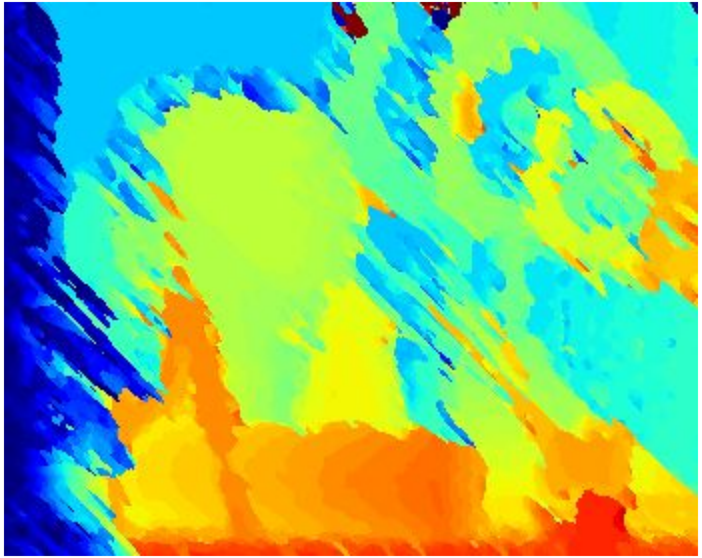
Scanline 3



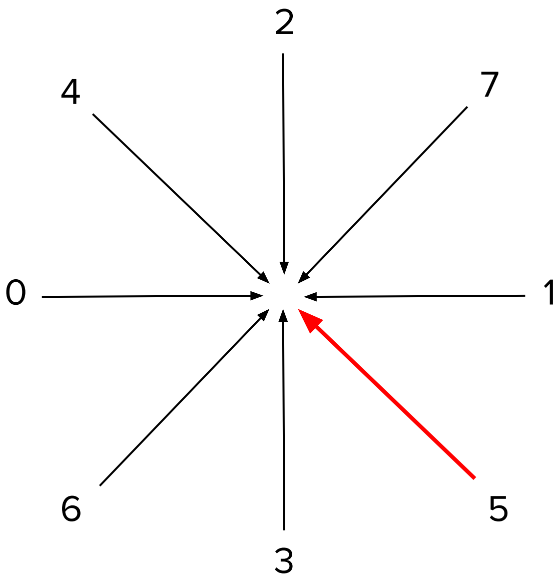
Streaking Artifacts



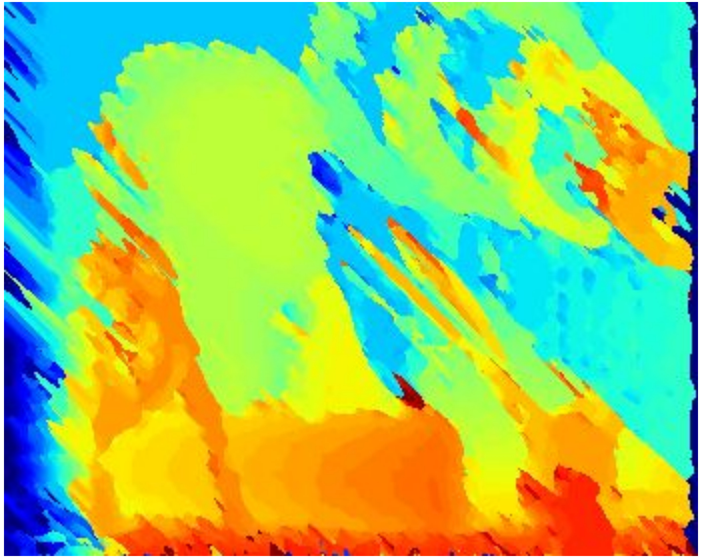
Scanline 4



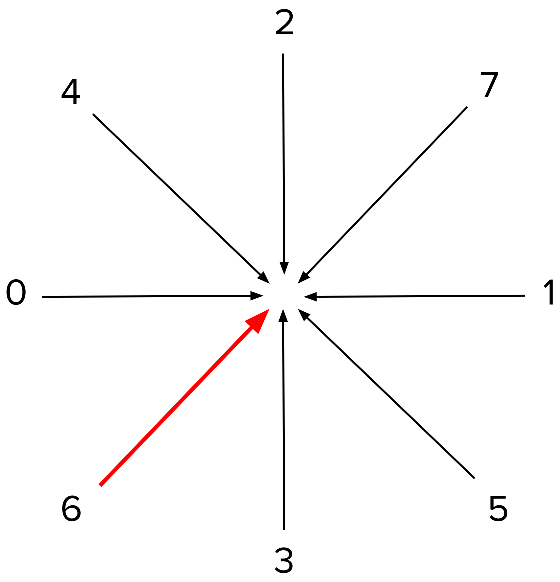
Streaking Artifacts



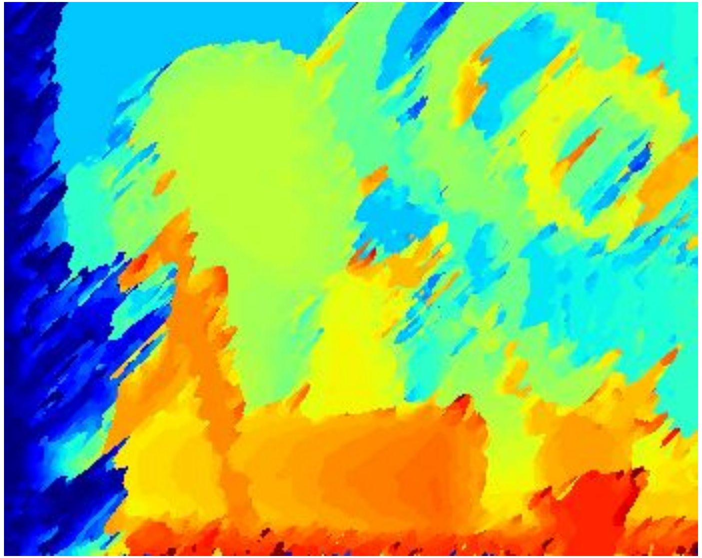
Scanline 5



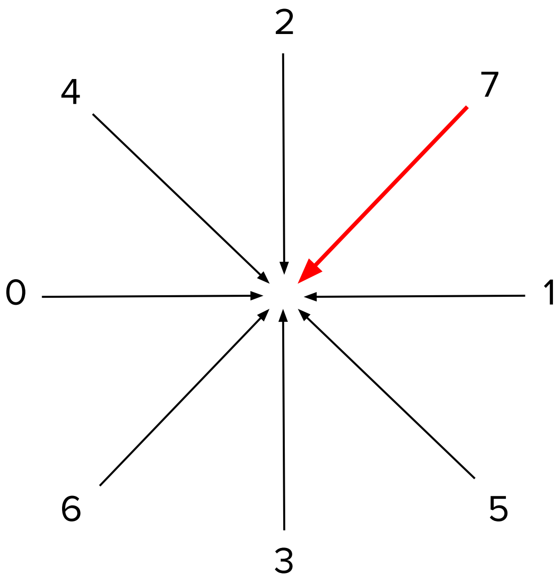
Streaking Artifacts



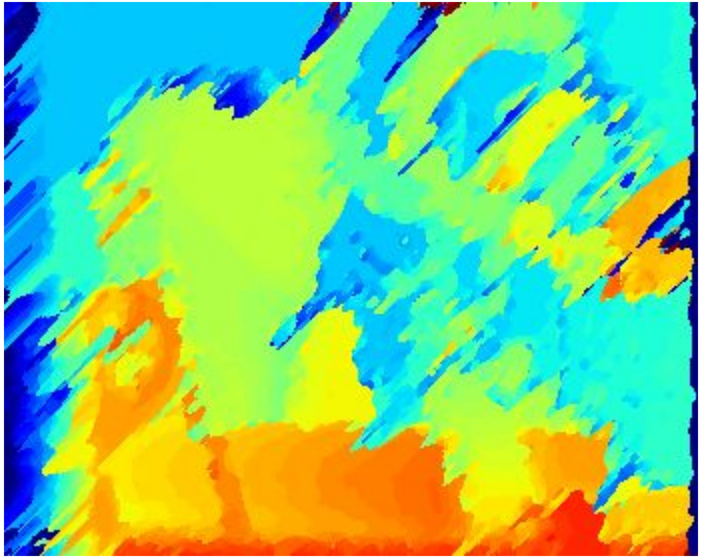
Scanline 6



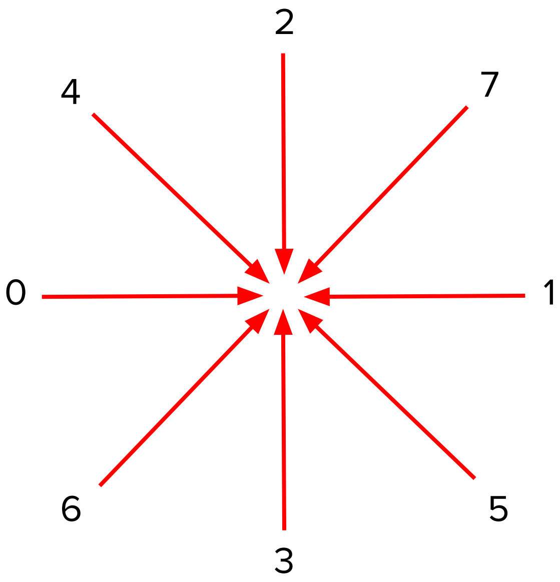
Streaking Artifacts



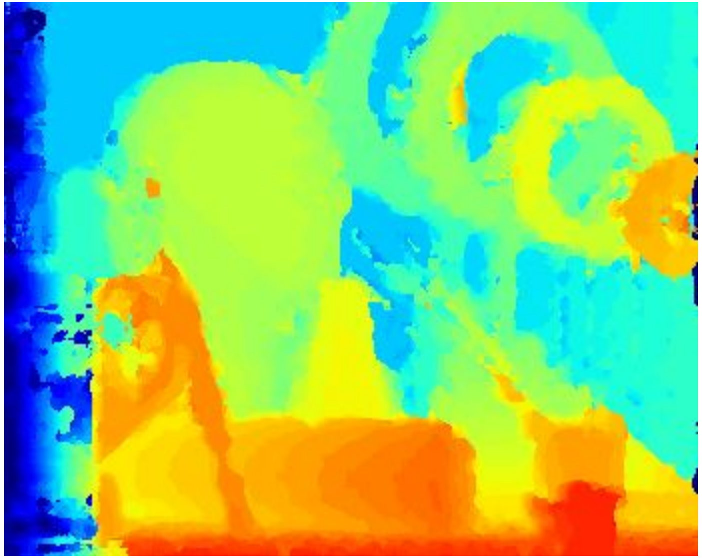
Scanline 7



Streaking Artifacts

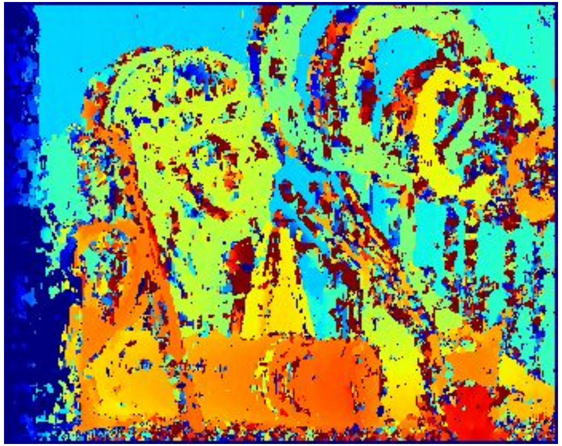


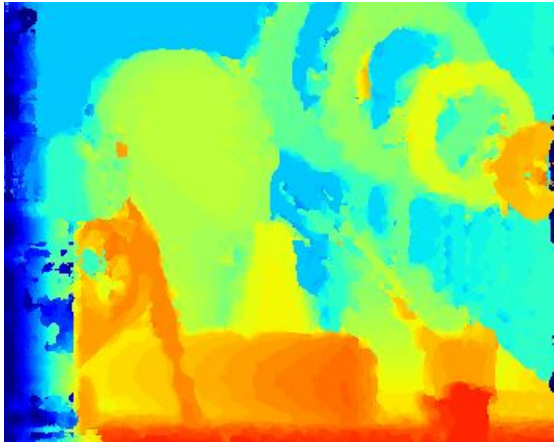
Scanline 0-7



Local vs (Semi-)Global


Reference Image

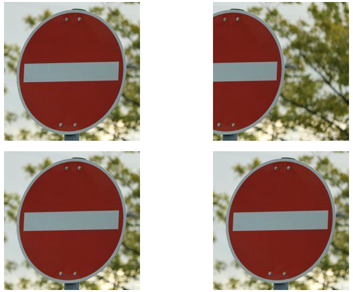

Local Approach


Semi-Global Approach

Siamese Networks for Stereo Matching

- **Hand crafted features** and similarity metrics do not take into consideration relevant geometric and radiometric invariances or occlusion patterns
- The world is too complex to specify this by hand
- Matching cost computation can be treated as **image classification problem**

Left Patch	Right Patch	Label



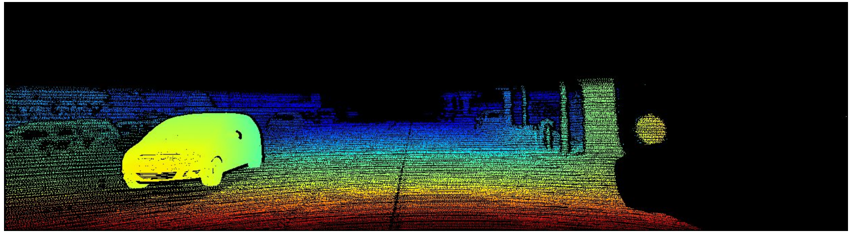
Wrong Match

Good Match → The two center pixels are the images of the same 3D position

Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches (Zbontar and LeCun, 2016)

- **Learning** a similarity measure on small image patches using a convolutional neural network (CNN)
- The output of the convolutional neural network is used to **initialize** the stereo matching cost
- Training is carried out in a **supervised** manner by constructing a binary classification data set with example of **similar** and **dissimilar** pairs of patches

Image



Ground truth Disparity (LiDAR)

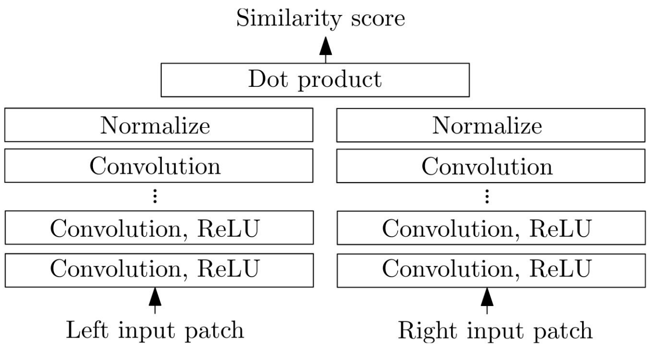
Network Architectures

Cosine Similarity:

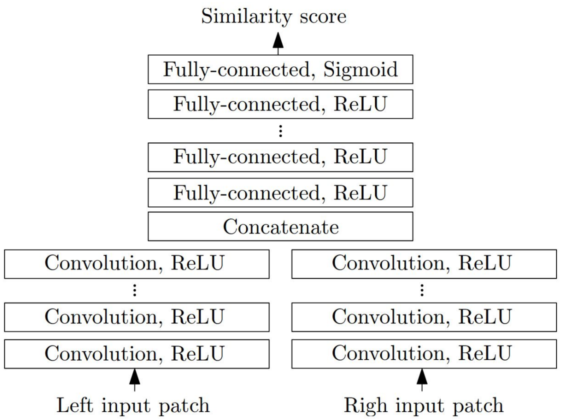
- Learn features and, then, dot-product
- Features must do the heavy lifting
- Fast matching (no network eval.)

Learned Similarity:

- Learn features and similarity metric
- Potentially more expensive
- Slow (WxHxD MLP evaluation)



MC-CNN-fast



MC-CNN-acrt

MC-CNN-acrt vs MC-CNN-fst

- In both architectures the Siamese network is responsible for describing the given patches by extracting learned features
- The fast architecture computes a similarity score using the **dot product** of the extracted features
- The accurate architecture **learns** a similarity function based on the extracted feature vectors
- As the names imply the accurate architecture (MC-CNN-acrt) is **more accurate** but much **slower**. This is because features must be concatenated and forward propagated through the fully connected layers for each candidate disparity *d*

Training Process

- The training set is composed of patch triplets

$$(\mathbf{w}_L(\mathbf{x}_L^{ref}), \mathbf{w}_R(\mathbf{x}_R^{neg}), \mathbf{w}_R(\mathbf{x}_R^{pos}))$$

- $\mathbf{w}_L(\mathbf{x}_L)$ is an image patch from the left image centered at $\mathbf{x}_L = (x_L, y_L)$
- $\mathbf{w}_R(\mathbf{x}_R)$ is an image patch from the right image centered at $\mathbf{x}_R = (x_R, y_R)$

How to choose both the positive and negative examples?

Training Process

- The training set is composed of patch triplets

$$(\mathbf{w}_L(\mathbf{x}_L^{ref}), \mathbf{w}_R(\mathbf{x}_R^{neg}), \mathbf{w}_R(\mathbf{x}_R^{pos}))$$

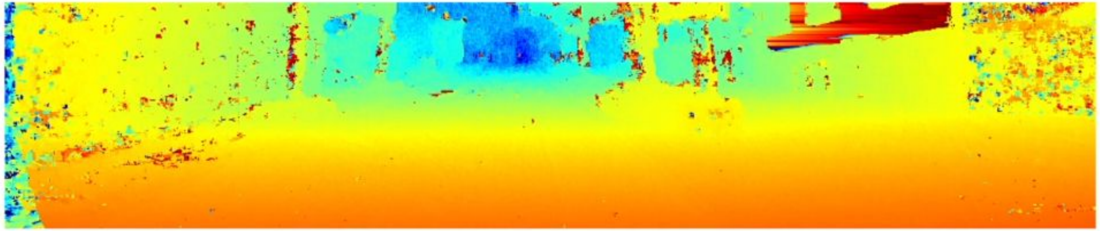
- $\mathbf{w}_L(\mathbf{x}_L)$ is an image patch from the left image centered at $\mathbf{x}_L = (x_L, y_L)$
- $\mathbf{w}_R(\mathbf{x}_R)$ is an image patch from the right image centered at $\mathbf{x}_R = (x_R, y_R)$
- Negative example: $\mathbf{x}_R^{neg} = (x_L^{ref} - d + o_{neg}, y_L^{ref})$
  - Offset o_{neg} drawn from $\mathcal{U}(\{-N_{hi}, \ldots, -N_{lo}, N_{lo}, \ldots, N_{hi}\})$
- Positive example: $\mathbf{x}_R^{pos} = (x_L^{ref} - d + o_{pos}, y_L^{pos})$
  - Offsets o_{pos} drawn from $\mathcal{U}(\{-P_{hi}, \ldots, -P_{hi}\})$
- Here, d denotes the true disparity for a pixel (provided as ground truth)
- Typically $P_{hi} = 1$, $N_{lo} = 3$ and $N_{hi} = 6$

Training Process

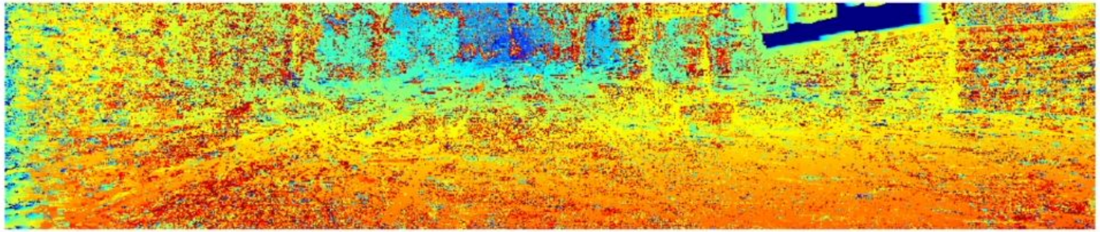
- **Ground truth disparities** from standard datasets (e.g. KITTI or Middlebury) to construct a **binary** classification dataset
- The fast architecture is trained using a **hinge loss** on pairs of positive and negative samples. The hinge loss for a pair is defined as $max(0, m + s_- - s_+)$. The loss is zero when the similarity of the positive example is greater than the similarity of negative example by at least the margin m
- The accurate architecture is trained using the **binary cross entropy** $tlog(s) + (1 - t)log(1 - s)$ where t is the ground label of the sample. 1 for positive and 0 for negative
- The decision to use two different loss functions, one for each architecture, was based on **empirical** evidence

Winner-takes All Results

Input Image



Siamese Network

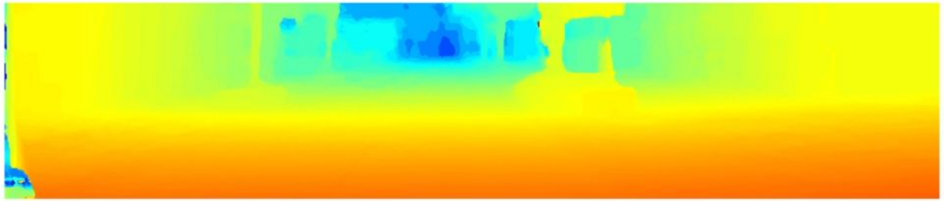


Standard Block-Matching

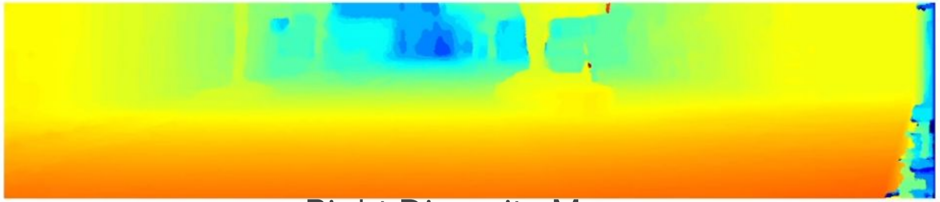
MC-CNN cost optimization and post processing

- Cross based cost aggregation (CBCA)
- Semi-Global Matching (SGM)
- Left-Right Consistency Check (LRC)
- Background Interpolation
- Subpixel enhancement
- Median Filter
- Bilateral Filter

MC-CNN cost optimization and post processing



Left Disparity Map



Right Disparity Map

Left-Right Consistency Check

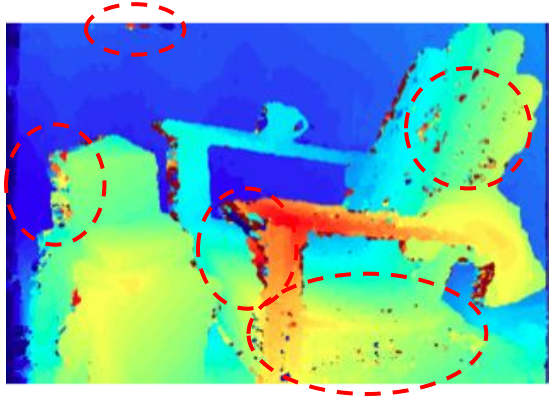
Runtime

- Original version implemented in CUDA and Lua/Torch7
- Run on Nvidia GTX Titan GPU
- **Training** takes 5 hours
  - 45 million training examples
  - 16 epochs
  - Stochastic gradient descent with batch size of 128
- **Inference** for a single pair of images takes 6 seconds/100 seconds
  - 1 second / 95 seconds for the neural network (depending on the architecture)
  - 3 seconds for the semi-global matching
  - 1 second for cost aggregation

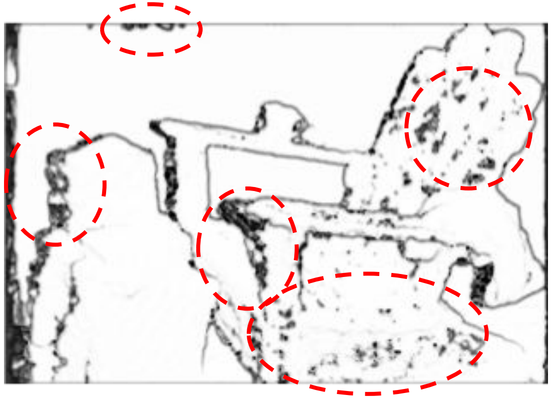
Confidence measures

- Regardless of the stereo algorithm, disparity maps contain **outliers**
- Confidence estimation aims at detecting such **unreliable** depth assignments

Reference image



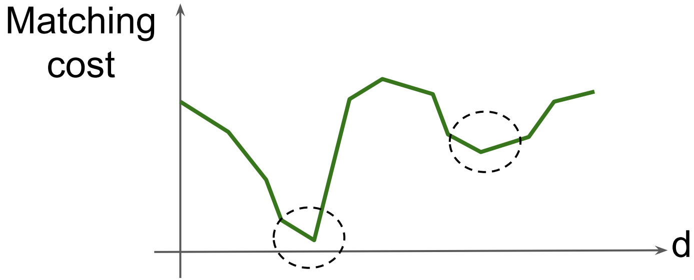
Disparity map (SGM)



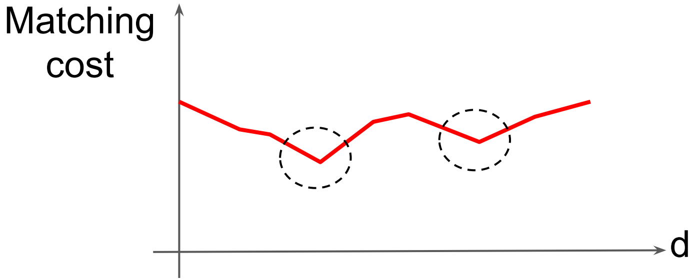
Confidence map
(the brighter, the more reliable)

Confidence measures - Basics

- Conventional methods, reviewed and evaluated in (Hu and Mordhoai, 2012), relies on assumptions mostly based on matching cost analysis
- For instance, the matching costs on the left are assumed to be more likely to yield a more reliable correspondence compared to the right ones
- Many other **heuristics** have been proposed in the literature

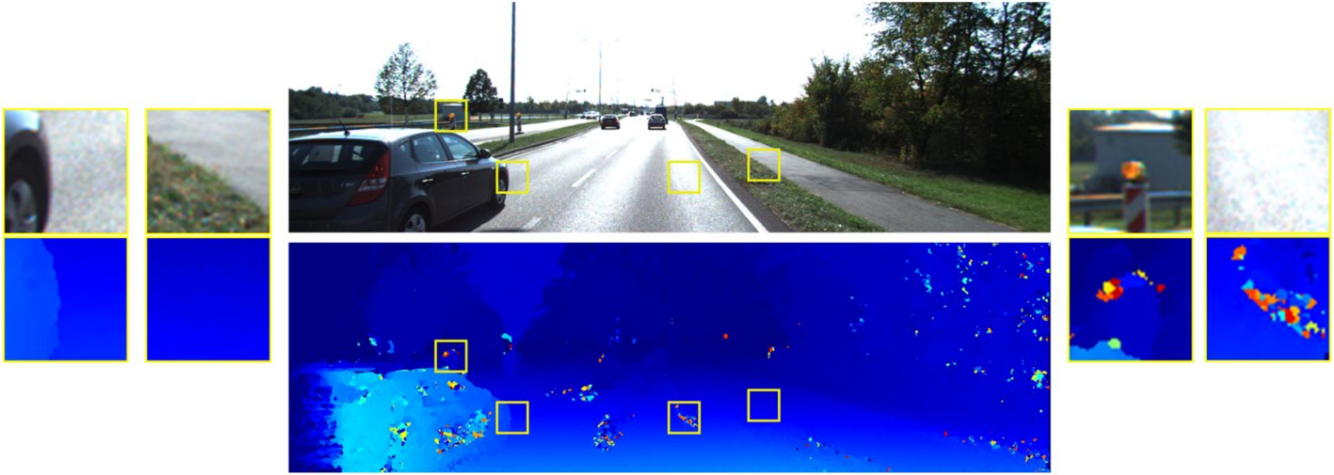


High confidence (?)

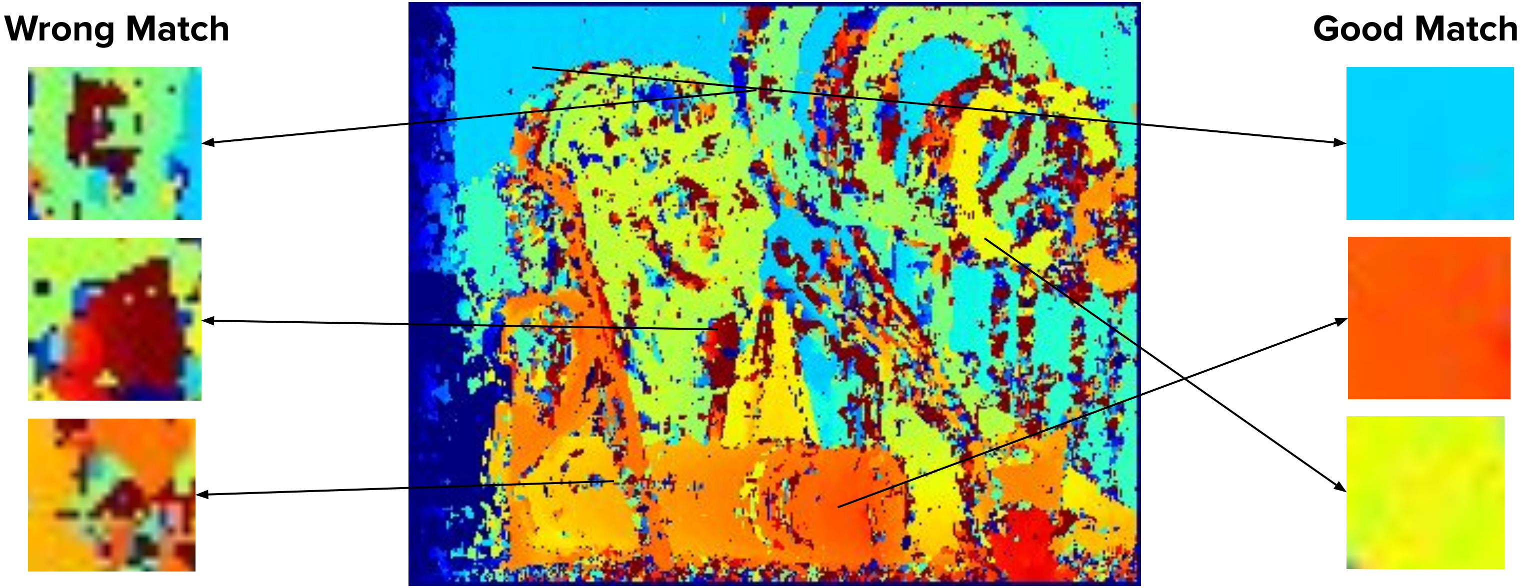


Low confidence (?)

Learning from scratch a confidence measure (Poggi and Mattoccia 2016)

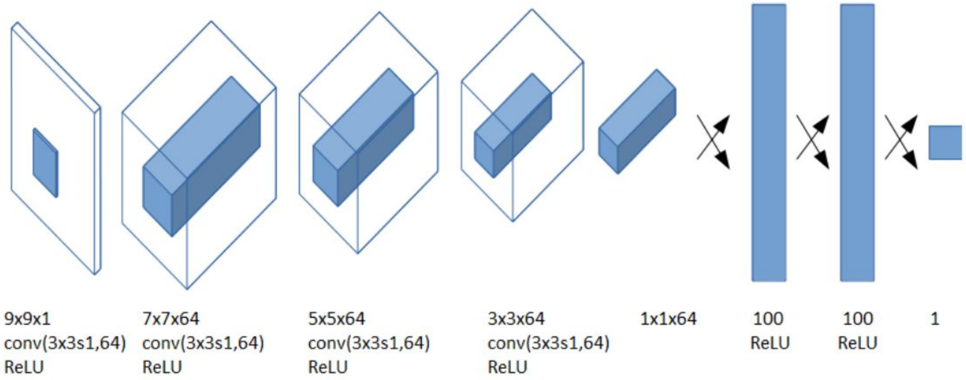


- Recurrent **local patterns** occurring in the disparity maps can tell a correct assignment from a wrong one
- Leveraging on **CNNs**, confidence formulation as a regression problem by analyzing the disparity map provided by a stereo vision system



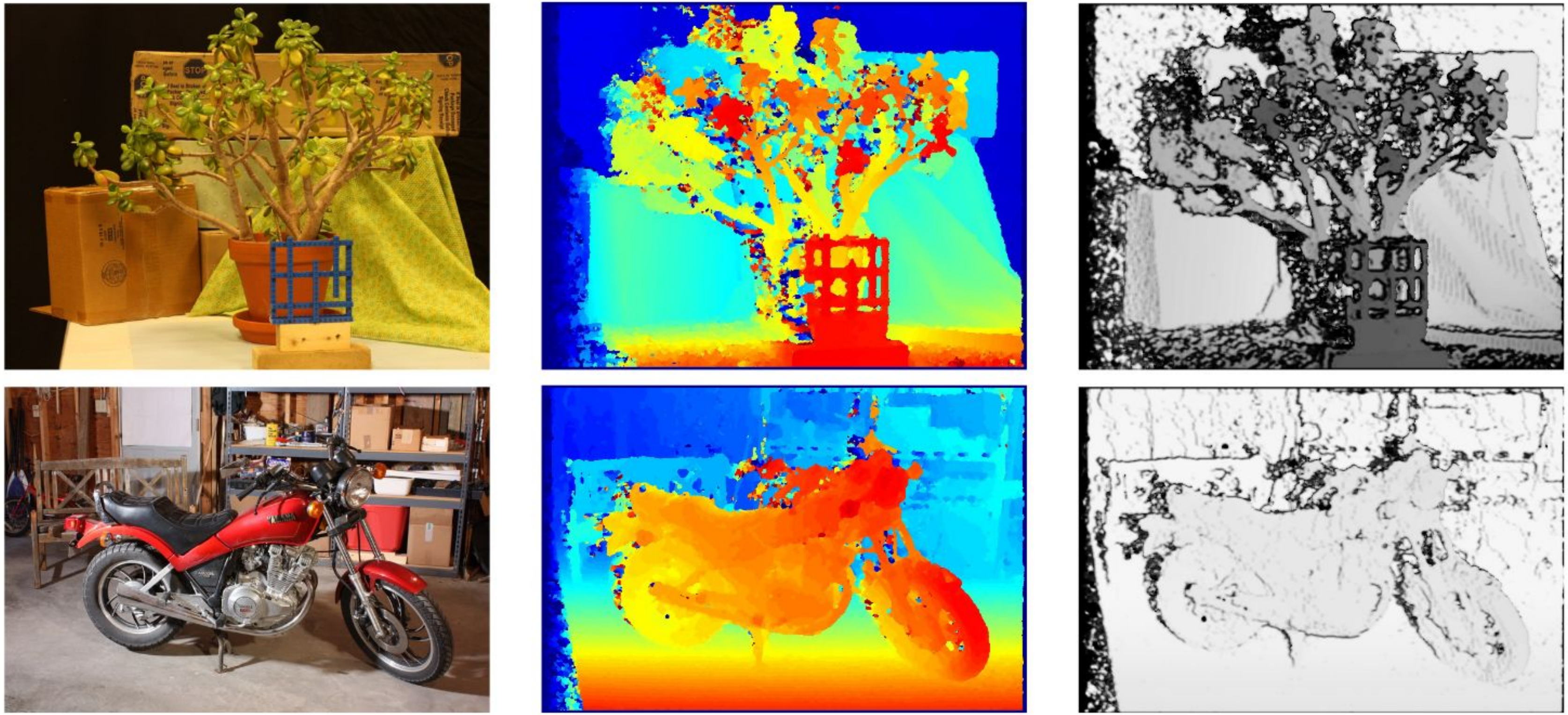
- By visual inspection, disparity maps contains meaningful patterns to tell correct assignments from wrong ones

Network Architecture



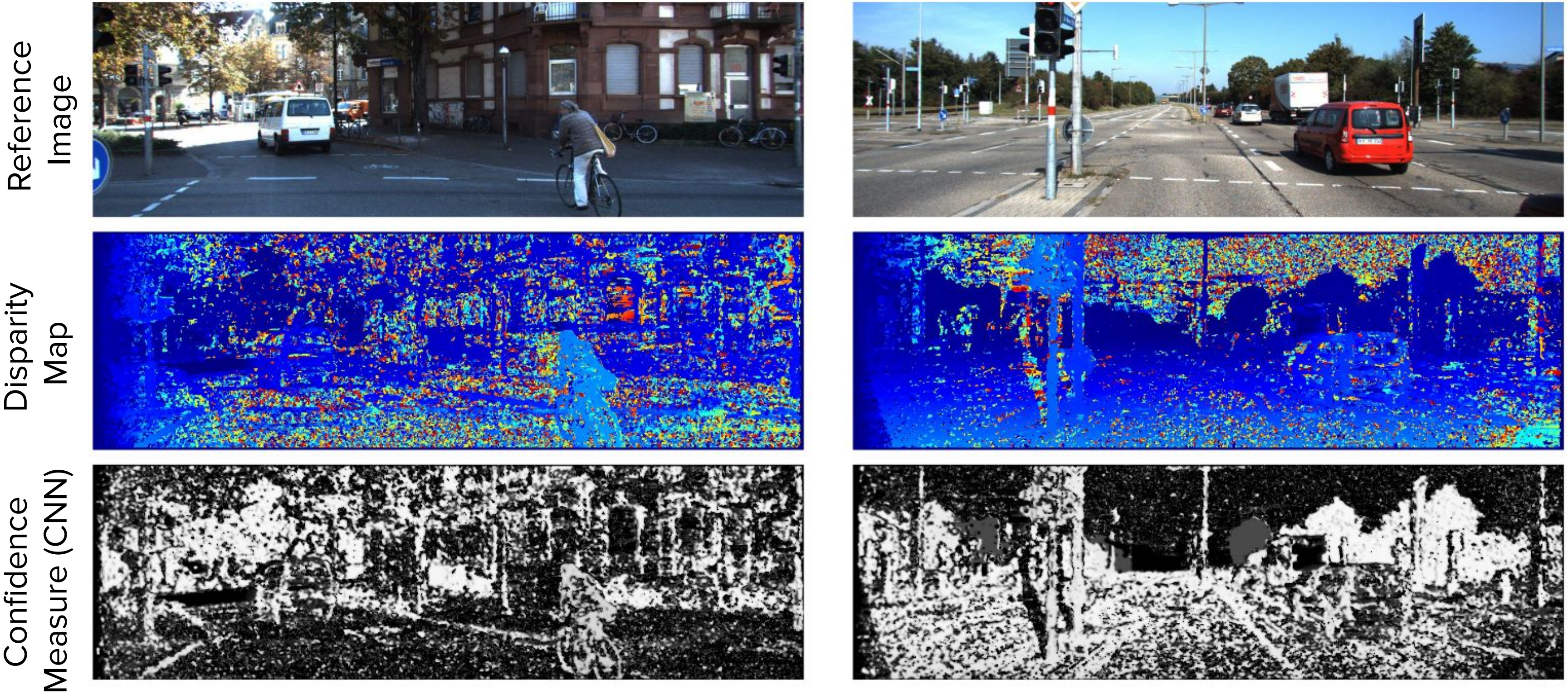
- A **single channel** network that takes small patches as input, each one containing **disparity** values normalized between zero and one.
- The **single** output value represents the degree of **uncertainty** from the disparity map
- **Binary cross-entropy** loss during training
- Trained in a **supervised** manner using disparities computed by a Block-Matching algorithm as well as SGM

Middlebury v3



Reference Image | Disparity Map | Confidence Measure (CNN)

KITTI 2015



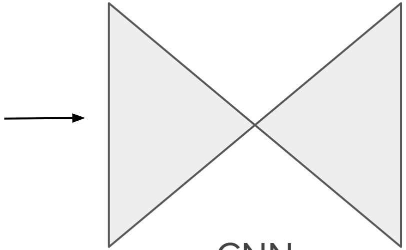
End-to-End Stereo Matching

- Convolutional Neural Networks proved good performance for single tasks of the stereo pipeline
  - Confidence Estimation
  - Matching Cost
  - Refinement

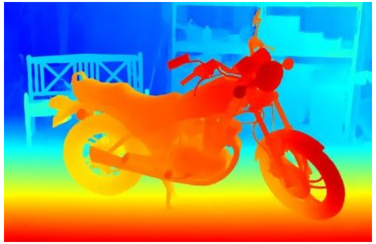
- However, separate trainings for each sub-step lead to **sub-optimal solutions**

End-to-End Stereo Matching

Stereo Pair



CNN



Disparity Map

- End-to-end models can reach unpaired **accuracy** if evaluated in the **same domain** as that on which they are trained

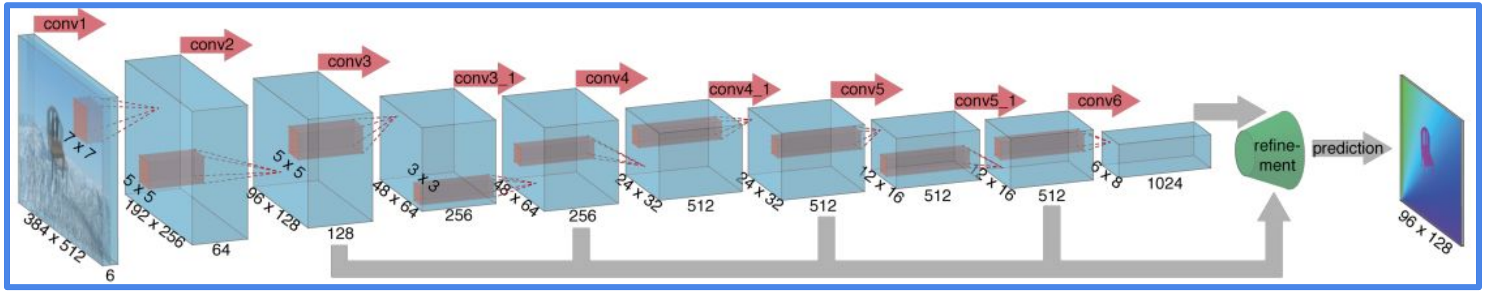
FlowNet and DispNet

- Dosovitskiy et al. proposed **FlowNet** (Dosovitskiy, 2015)
  - End-to-end architecture for optical flow estimation
  - Extremely fast (10+ FPS on GPU)
  - Promising results on synthetic datasets (MPI Sintel)

- Mayer et al. proposed **DispNet** (Mayer, 2016)
  - Competitive with state-of-the-art in 2016 (MC-CNN-acrt) on KITTI data
  - But 100x faster than MC-CNN-acrt!

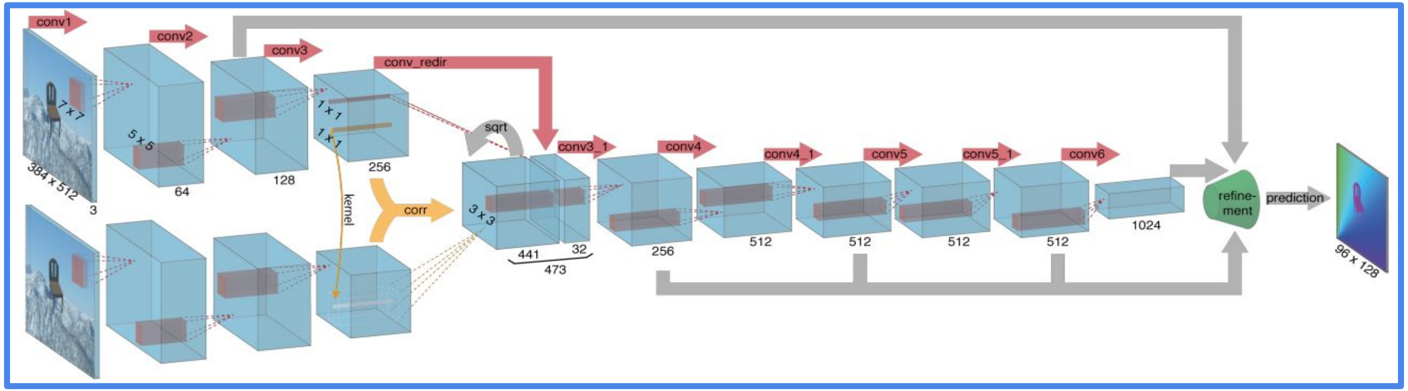
- Both requires a **huge** amount of data to be trained

FlowNet and DispNet



- **U-Net architectures**
  - <u>Encoding part</u>: decimates resolution while increasing receptive field
  - <u>Decoding part</u>: restores original resolution (actually, half resolution)
  - <u>Skip connections</u> between encoder and decoder to recover fine details
- **Dense regression task**
  - End-point-error between prediction and ground truth flow/disparity as loss function

FlowNet and DispNet



- **Correlation layer**
  - Features are extracted from input images
  - Shifted correlations (i.e. dot products) between features on the two images
    - Optical flow: 2D search window
    - Stereo: 1D (horizontal) search
  - Concatenation on the feature channel

Shared parameters to more effectively learn corresponding features

Data for Training

- Given a single stereo pair, we have
  - **thousand of samples** for patch-based CNNs (small receptive field)
  - **one sample** for end-to-end architectures (large receptive field)
- KITTI provides only 200 pairs for training -> not enough for DispNet!
- Use of **synthetic datasets**

Synthetic data: SceneFlow Dataset

- A large **synthetic dataset** has been released (Mayer, 2016)
- 39K synthetic stereo pairs in 3 splits
  - FlyingThings3D
    - Train split - 22K
    - Test split - 4K
  - Driving - 4.5K
  - Monkaa - 8.5K

- **Ground truth** disparity, optical flow, disparity change (for scene flow) and object segmentation are provided
- **Fine-tuning** on little real data (expensive annotations)

Towards state-of-the-art

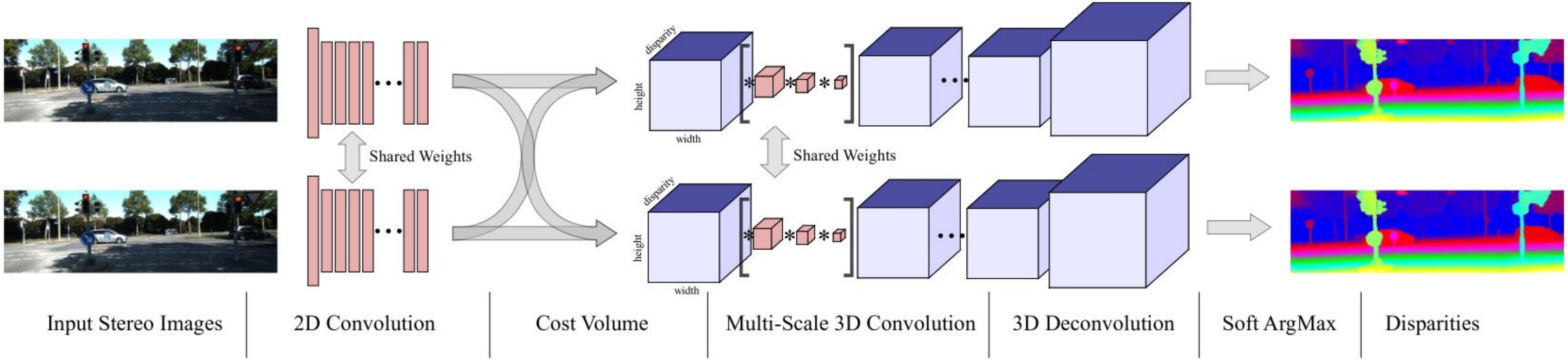
- DispNetC proved that end-to-end CNNs can be **extremely fast** and **competitive**, but still less accurate than hand-designed pipelines

- Re-thinking the architecture of the network considering **explicit knowledge** about the problem, e.g. geometry (Kendall, 2017), will bring these approaches to dominate the most popular benchmarks

End-to-End Learning of Geometry and Context for Deep Stereo Regression (Kendall, 2017)

- Avoid designing each step of the stereo algorithm by hand
- Use of the insights from many decades of multi-view geometry research to guide architectural design (no black-box model)
- **Differentiable layers** representing each major component in traditional stereo pipelines
- Goal: learn the entire model end-to-end while leveraging our **geometric knowledge** of the stereo problem

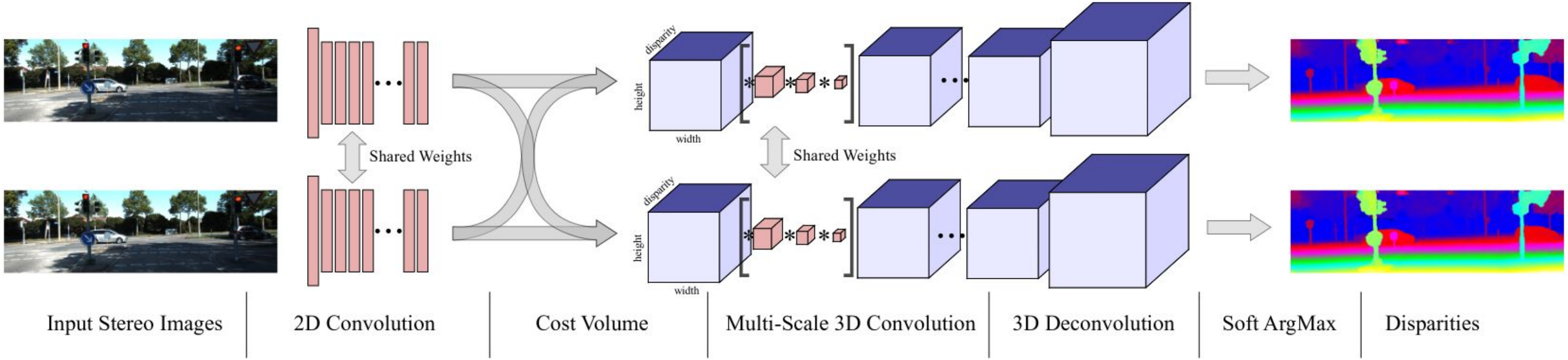
GC-Net (Geometry and Context Network)



- 2D feature extraction
  - resnet-18 feature extractor (shared weights)
- Cost volume building
  - **features concatenation**: D x H x W x 2F (4D cost volume)
- 3D feature optimization
  - U-Net encoder-decoder with 3D convolutions and skip connections
- **Differentiable** WTA (soft-argmax) -> standard WTA is not differentiable and it is discrete

Descriptor which is more robust to the ambiguities in photometric appearance and can incorporate local context

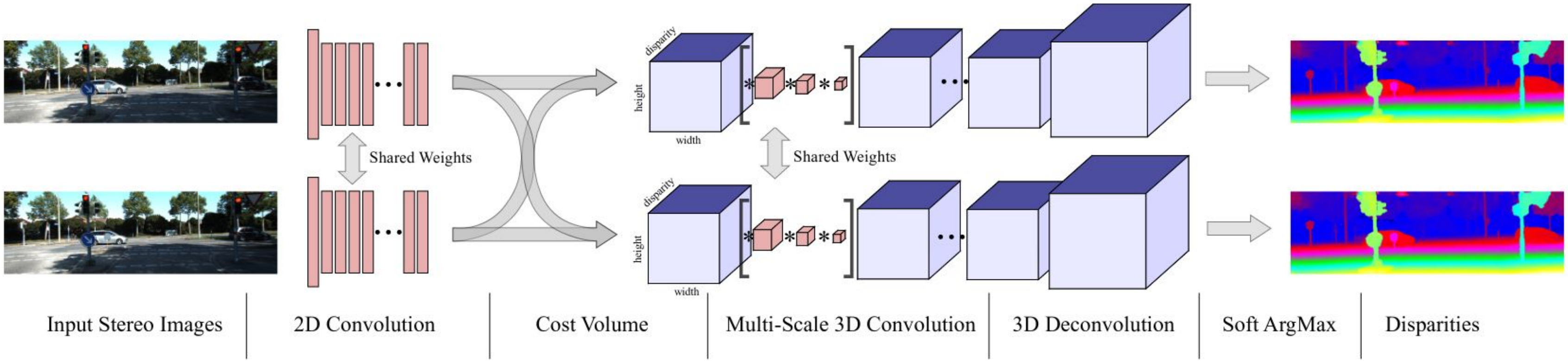
GC-Net (Geometry and Context Network)



- 2D feature extraction
  - resnet-18 feature extractor (shared weights)
- Cost volume building
  - **features concatenation**: D x H x W x 2F (4D cost volume)
- 3D feature optimization
  - U-Net encoder-decoder with 3D convolutions and skip connections
- **Differentiable** WTA (soft-argmax) -> standard WTA is not differentiable and it is discrete

Forming a cost volume allows to constrain the model in a way which preserves the knowledge of the geometry of stereo vision

GC-Net (Geometry and Context Network)

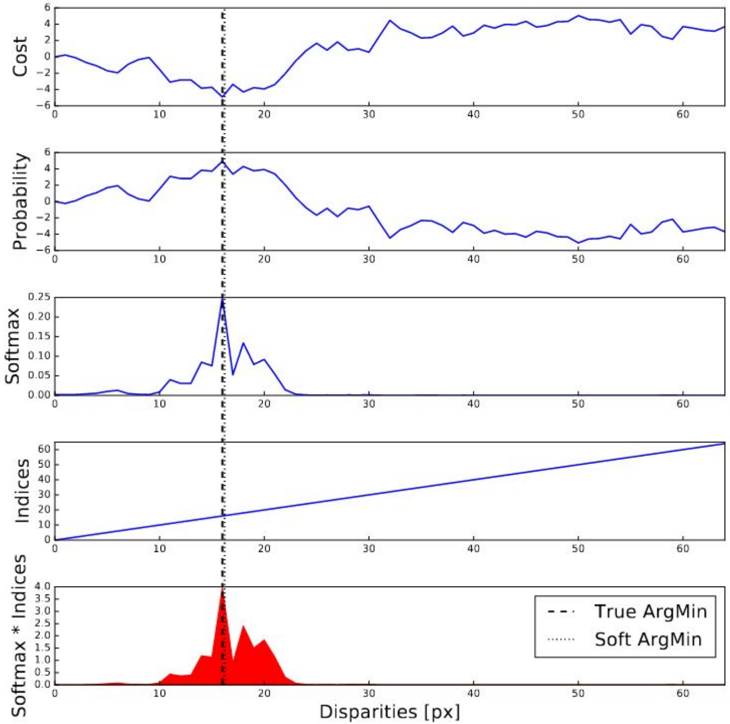


- 2D feature extraction
  - resnet-18 feature extractor (shared weights)
- Cost volume building
  - **features concatenation**: D x H x W x 2F (4D cost volume)
- 3D feature optimization
  - U-Net encoder-decoder with 3D convolutions and skip connections
- **Differentiable** WTA (soft-argmax) -> standard WTA is not differentiable and it is discrete

Matching costs between unary features can never be perfect. The goal is to learn to regularize and improve this volume

$$SoftArgmax := \sum_{d=0}^{D_{max}} d \times \sigma(-c_d)$$

Fully Differentiable and allows **sub-pixel** disparity estimates



Soft ArgMax

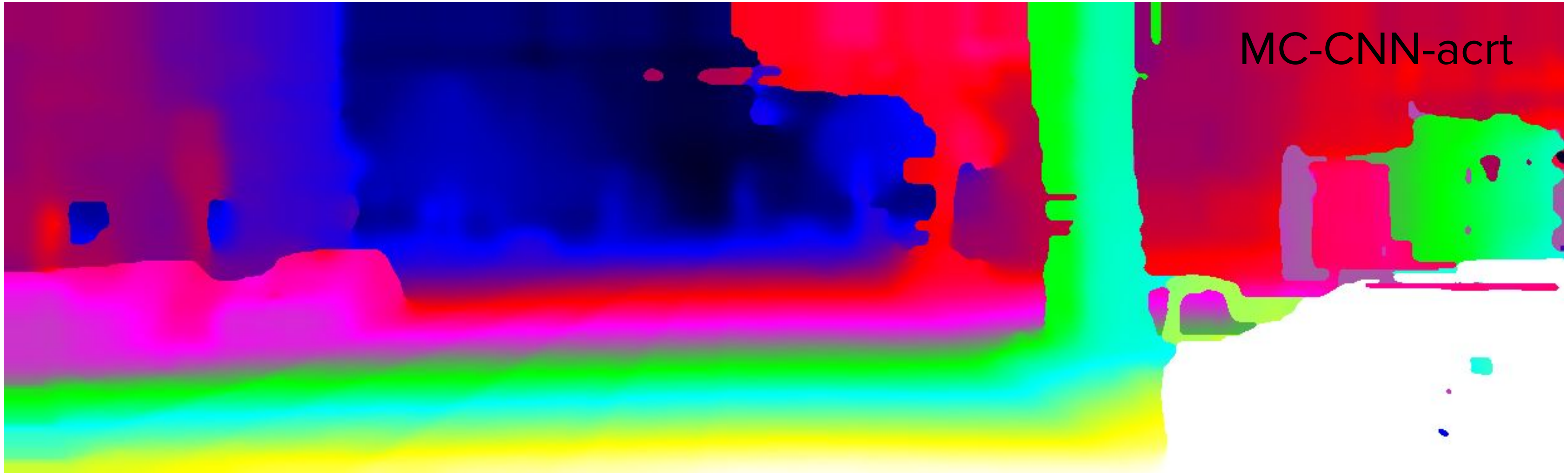
Correlation vs 4D volume

- To resume, the network reported so far are built upon one of these principles:
  - **Feature Correlation**
    - Encodes similarity into features channel
    - Faster runtime, but the real geometric context is lost
  - **4D Cost Volume**
    - Similarity costs as third dimension
    - Slower runtime, but real geometric context is maintained
    - High amount of memory usage

Qualitative results on the KITTI benchmark

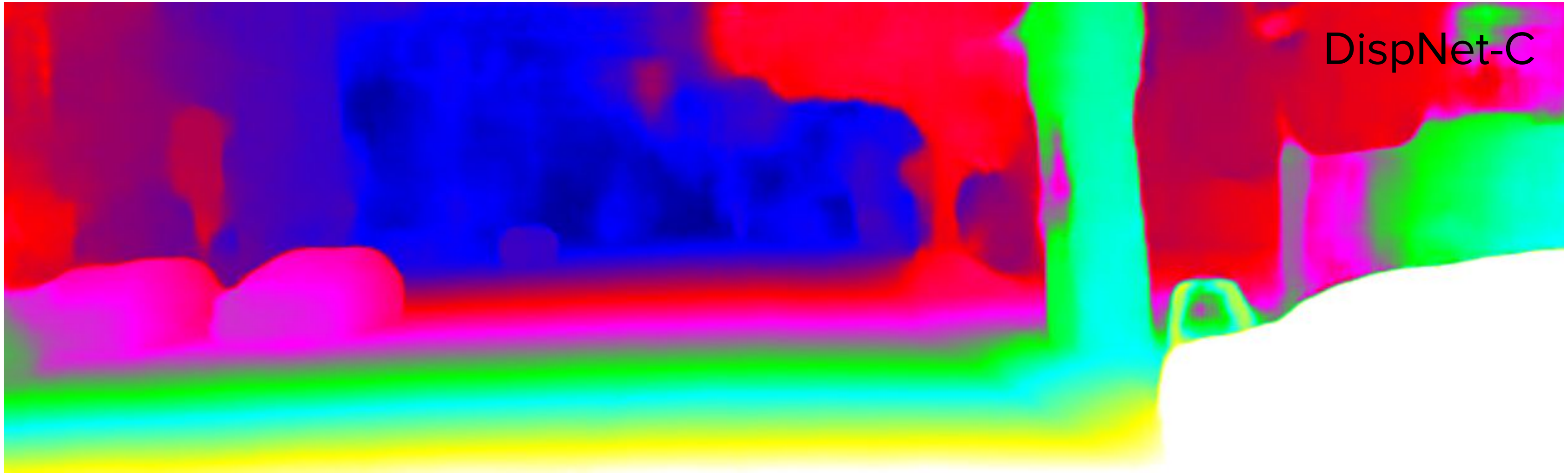
Left Image

Qualitative results on the KITTI benchmark



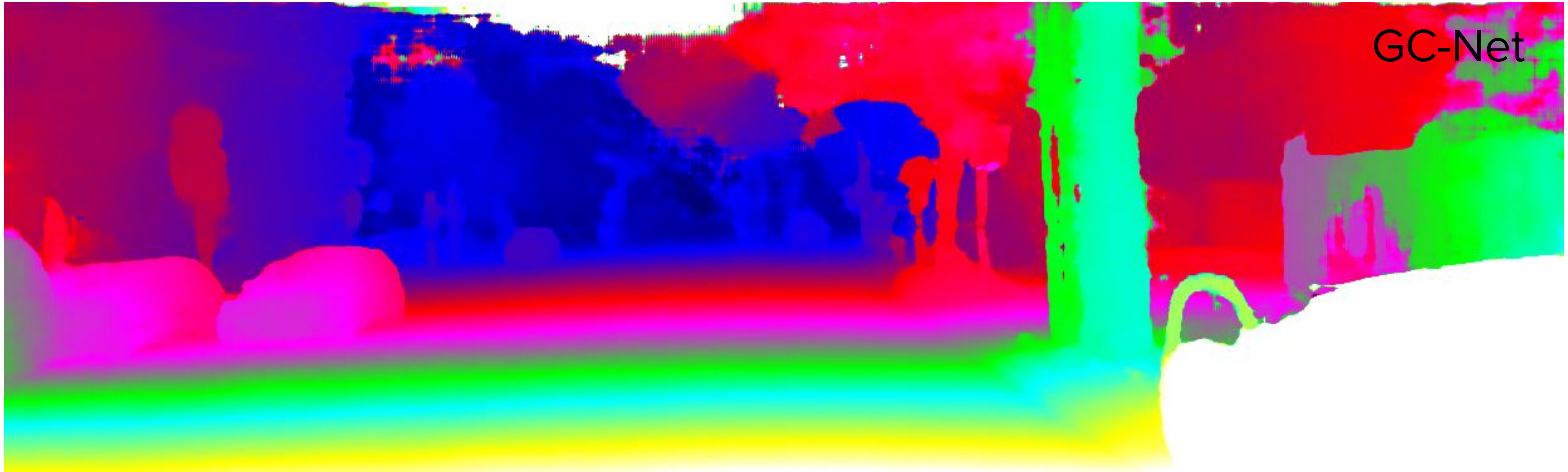
Disparity Map (D1-All: **4.99**)

Qualitative results on the KITTI benchmark



Disparity Map (D1-All: **4.53**)

Qualitative results on the KITTI benchmark



Disparity Map (D1-All: **1.92**)

Domain Shift

- Deep stereo networks works extremely well when enough data is available
- Most of them use large synthetic datasets. However..

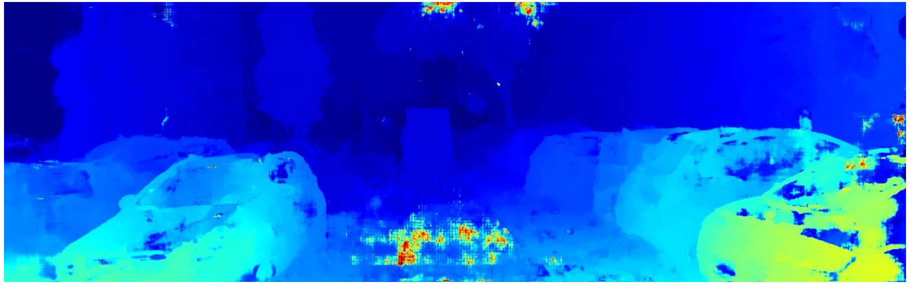
Synthetic Image

Real-World Image

- Domain shift caused by the very different conditions between **real** and **fake** imagery results in lower accuracy on real environments

Domain Shift

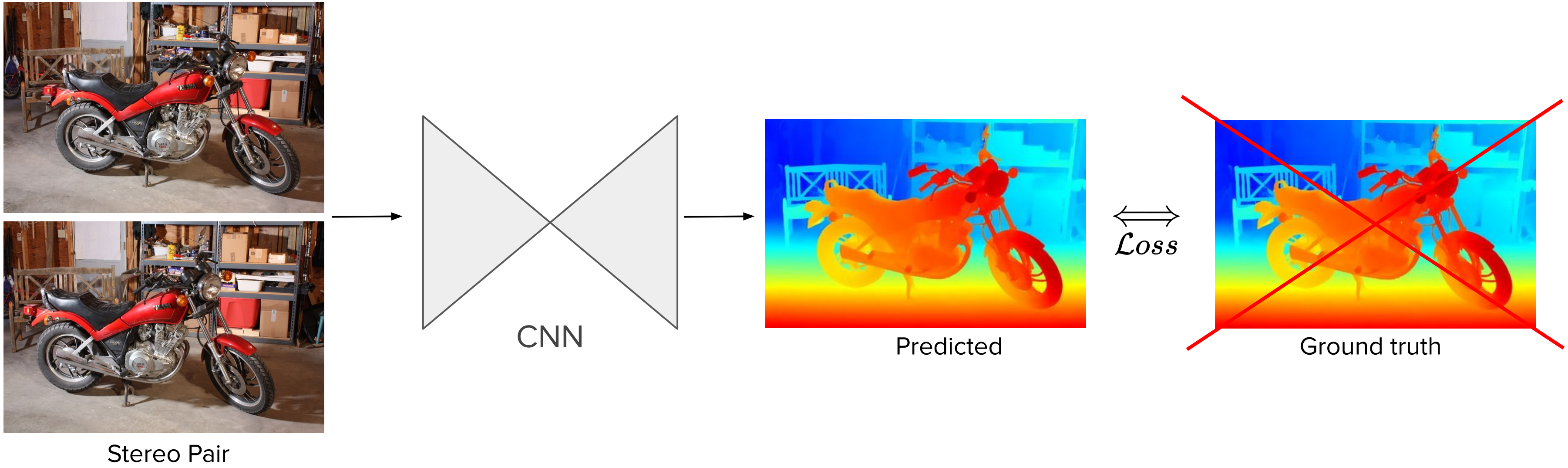
- Train data is **hard** and **expensive** to collect
- Further **fine-tune** on few annotated samples of the target domain is performed to address the domain shift.



Disparity Map - without fine-tuning

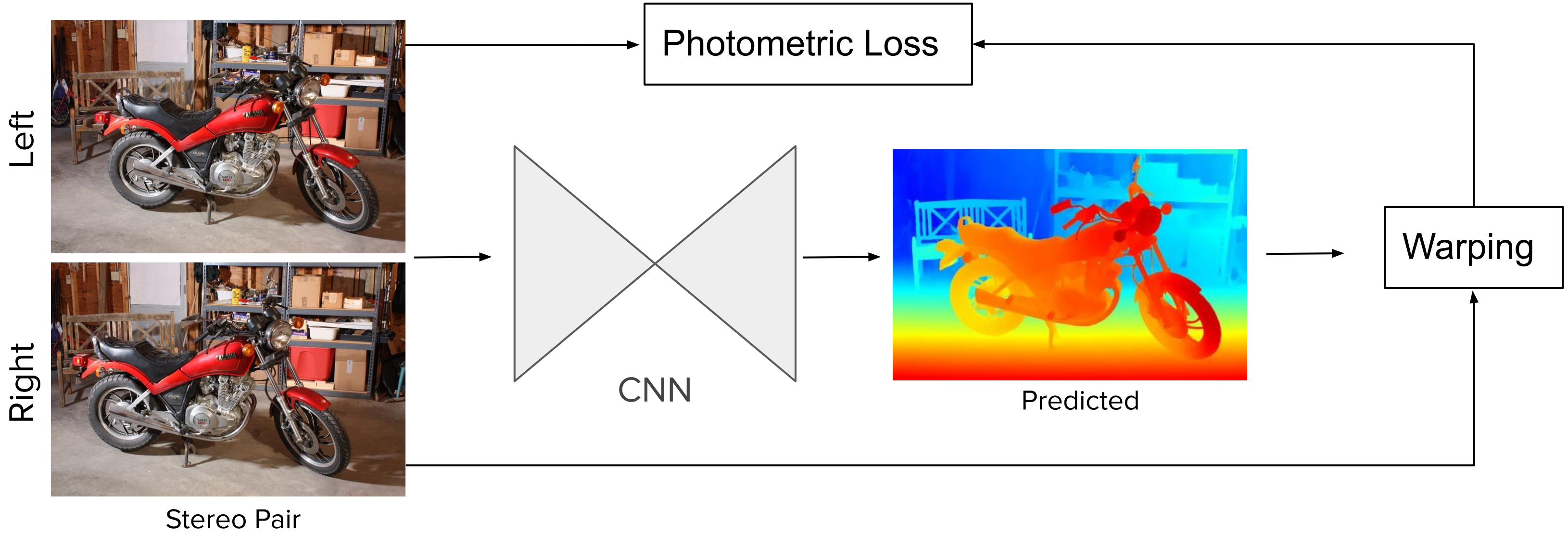
Disparity Map - with fine-tuning

Self-Supervised Stereo



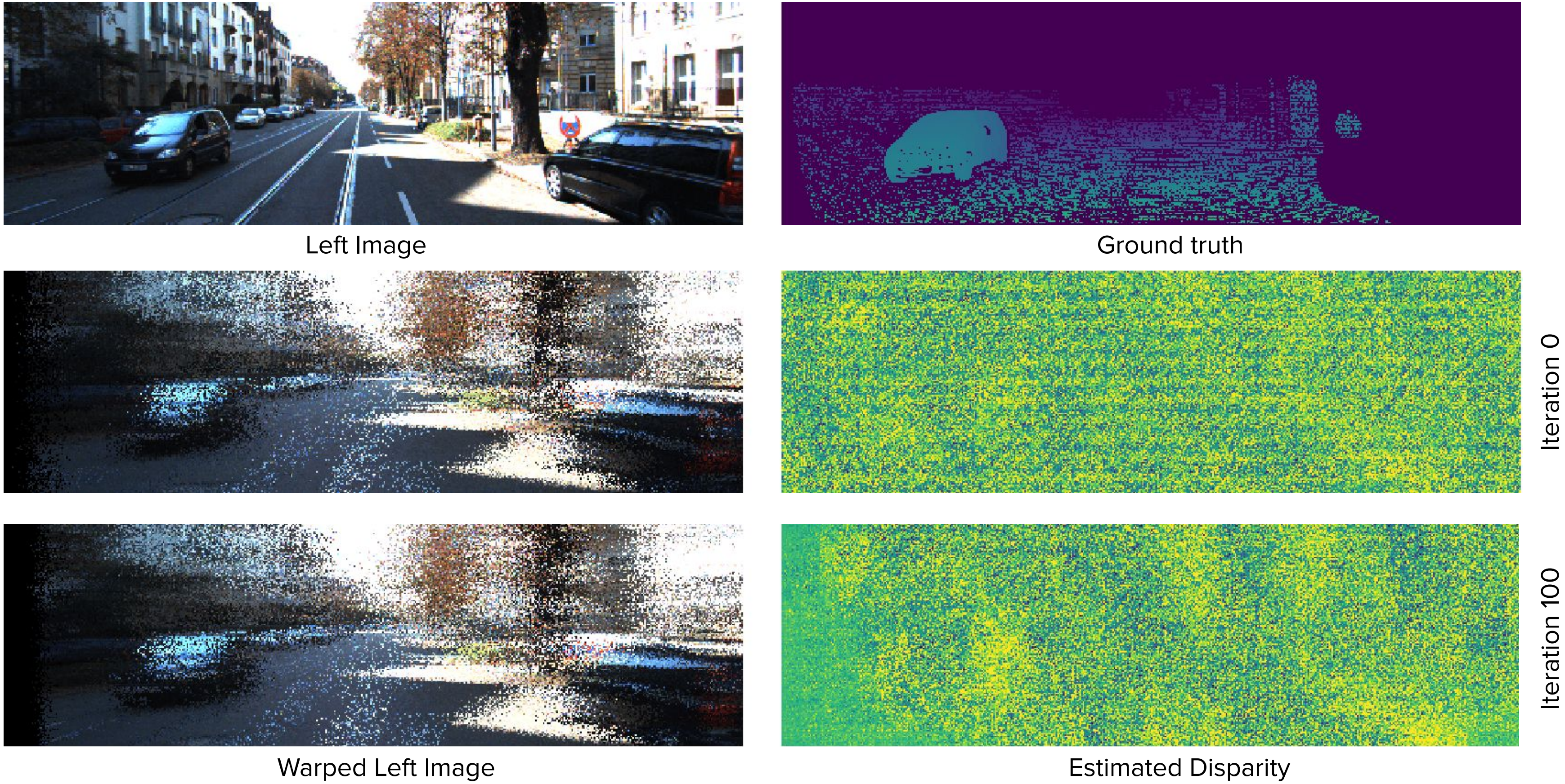
- Obtaining ground truth depth labels on real-world scenes is really expensive
- What if ground truth depth labels are not available on the target domain?
- **Self-supervision** as alternative solution

Self-Supervised Stereo

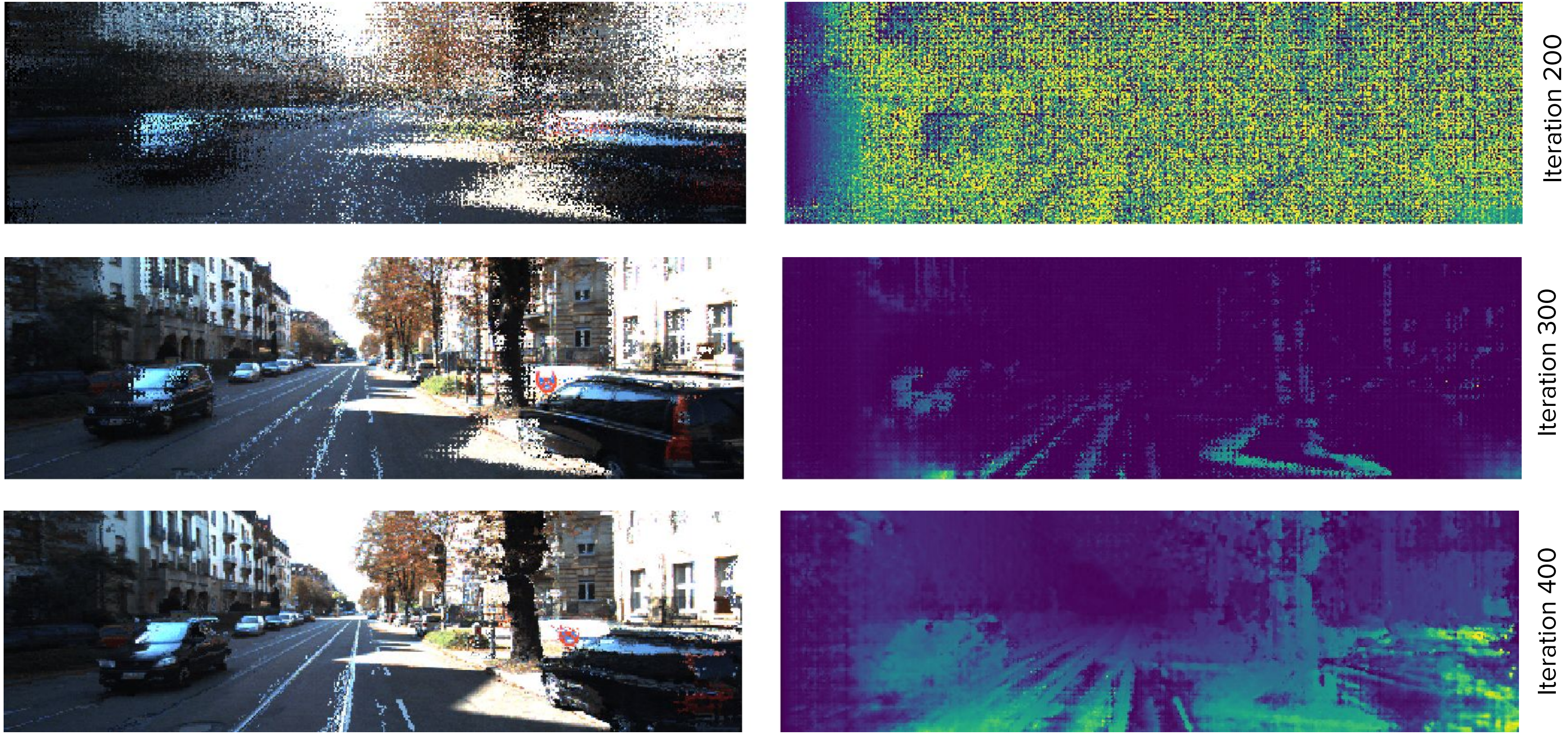


- The intuition is that if we can warp between the image pair properly, then we must have learned the dense disparity map
- Given the right image and the disparity map for the left image, the left image can be generated by warping the right image with the dense disparity map as $I'_L(x, y) = I_R(x - d(x, y), y)$

Self-Supervised Stereo



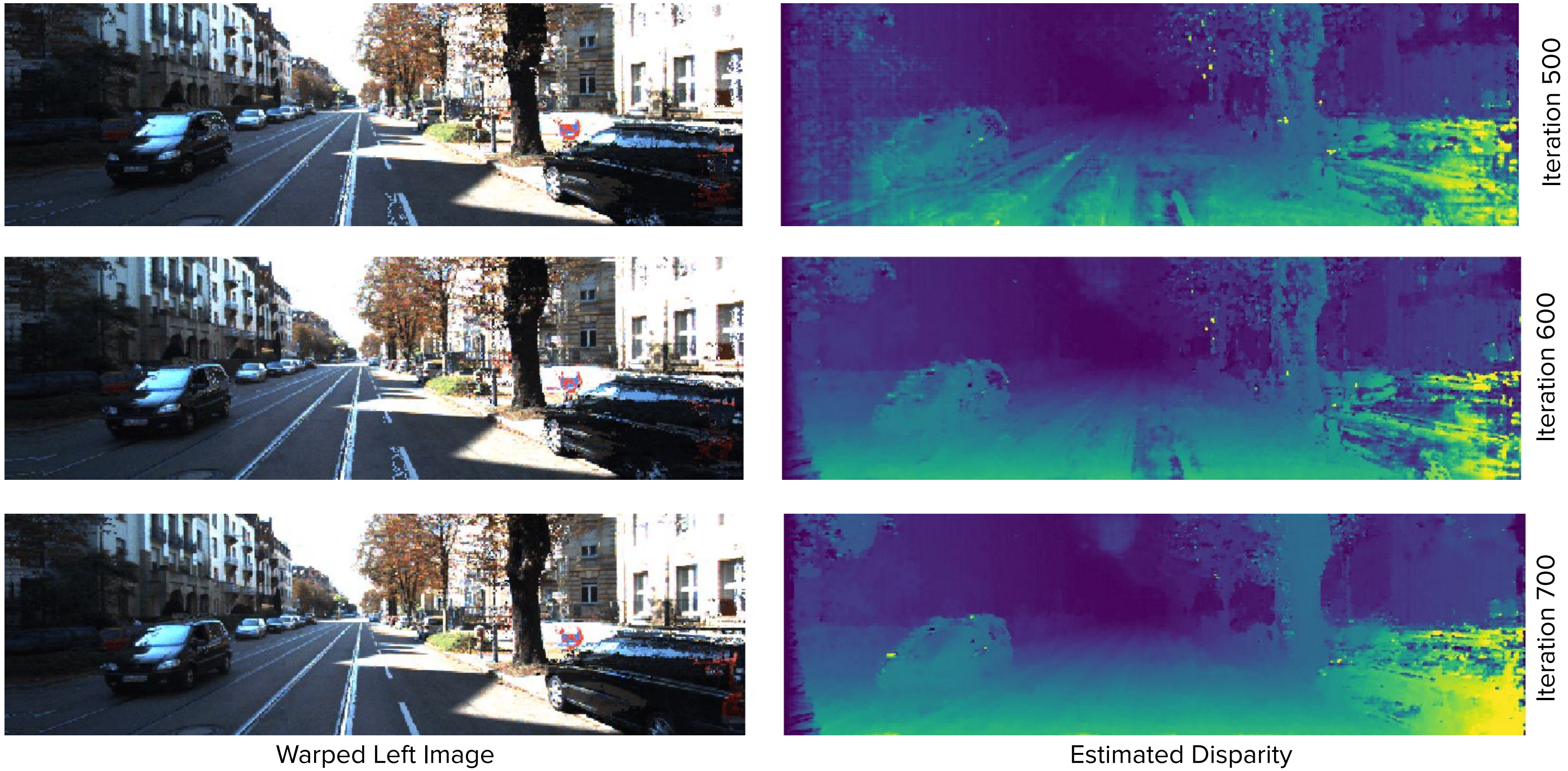
Self-Supervised Stereo



Warped Left Image

Estimated Disparity

Self-Supervised Stereo



Discussion

- The greatest turning-point was due to the change from hand-crafted pipelines to **end-to-end** networks
- Conventional knowledge about stereo **survived** this paradigm shift and has not gone extinct
- The main shortcomings introduced by end-to-end models concern the need for large amounts of ground truth annotated samples
- Two major challenges remain in this field:
  - **Generalization** across different domains
  - Applicability on **high-resolution** images