POLITECNICO
MILANO 1863

Advanced Deep Learning Models and Methods for Spatial Data

Giacomo Boracchi, Luca Magri, Simone Melzi, Matteo Matteucci

Depth estimation

Luca Magri
https://magrilu.github.io
name.surname@polimi.it

POLITECNICO
MILANO 1863

Credits & acknowledgments

Some of the material presented in these slides is based on

Matteo Poggi*'s tutorial: Learning and understanding single image depth estimation in the wild, CVPR 2020*

which is an excellent resource for approaching the problem of depth estimation.

The introductory material is inspired by Justin Johnson: *Deep learning for computer vision* (lecture 17)

Monodepth and material on self-supervision is based on the oral presentation done by Clément Godard at CVPR 2017.

Some slides are "stolen" from the AN2DL course of Prof. Giacomo Boracchi, who is kindly acknowledged here for his suggestions and support. Typically, the images are taken from the papers cited at the bottom of each slide.

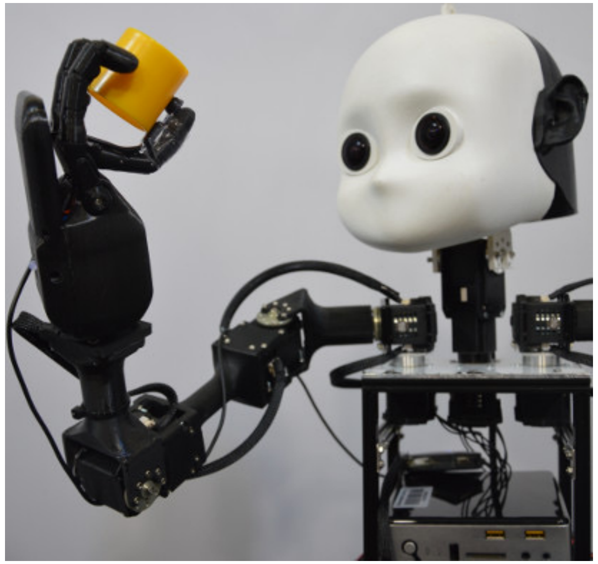
The codes accompanying this lecture were provided by Andrea Porfiri dal Cin, whom I thank for his help.

Errors are my own! You are encouraged to report any of them to luca.magri@polimi.it

Motivations

Navigation & mapping

Robot grasping



Augmented reality

Applications

- Robotics
- Autonomous Driving Assistive System
- Medical applications

Perceiving depth

Active technologies

- Structured Light
- Time of Flight (TOF)
- Laser Image Detection Ranging (LiDAR)

✓ very accurate

LiDAR:

✗ Sparse measurements
✗ Expensive

Structured Light:

✗ Can’t work outdoor
✗ Limited range

Passive technologies

- Binocular and multi-view stereo
- Structure from Motion

✓ cheap

Stereo:

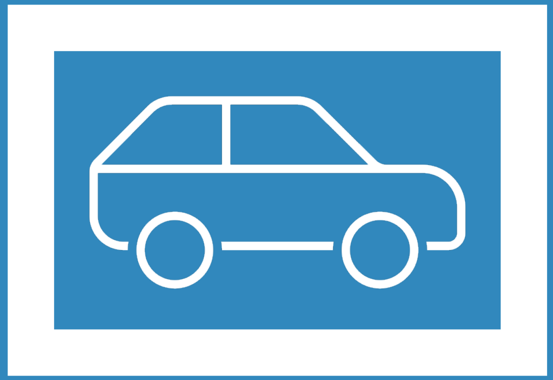
✗ Occlusions

SfM:

✗ Moving objects

By estimating depth from a **single** image, we can bypass all these limitations!

Depth estimation from a single image

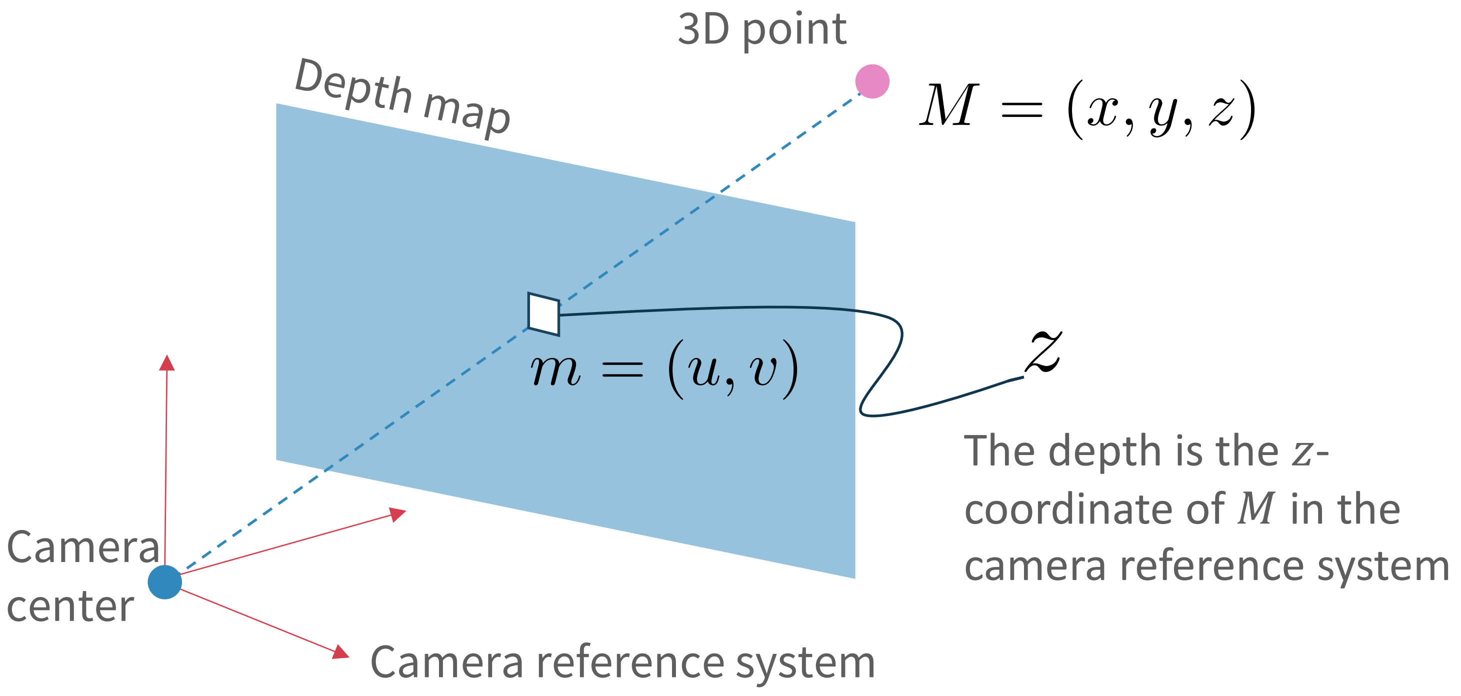


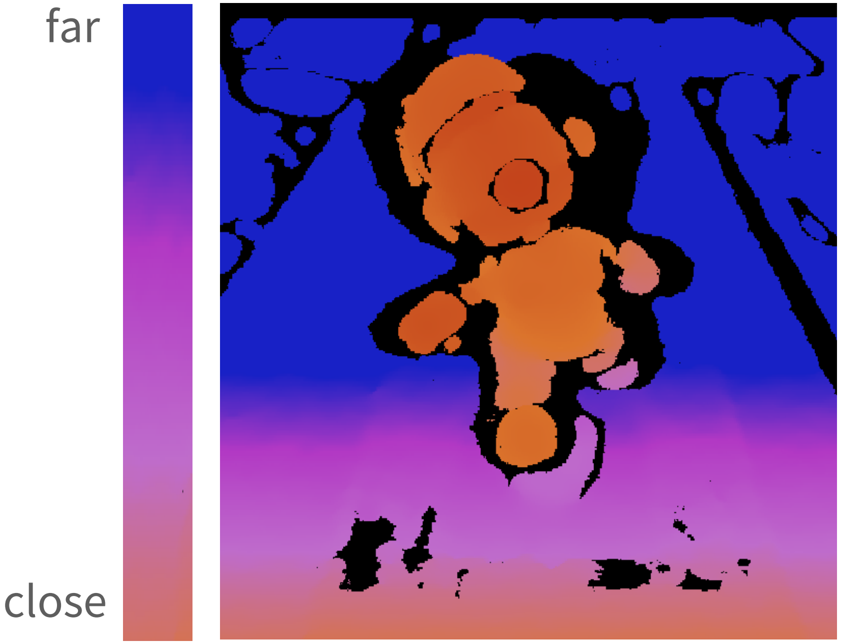
Depth map

Given a RGB image of size $H \times W \times 3$, we want to estimate

a **depth map**: an image of size $H \times W$
that, for each pixel, gives the distance from the camera to the object in the world at that pixel.

RGB image + Depth map = RGB-D image 2.5D



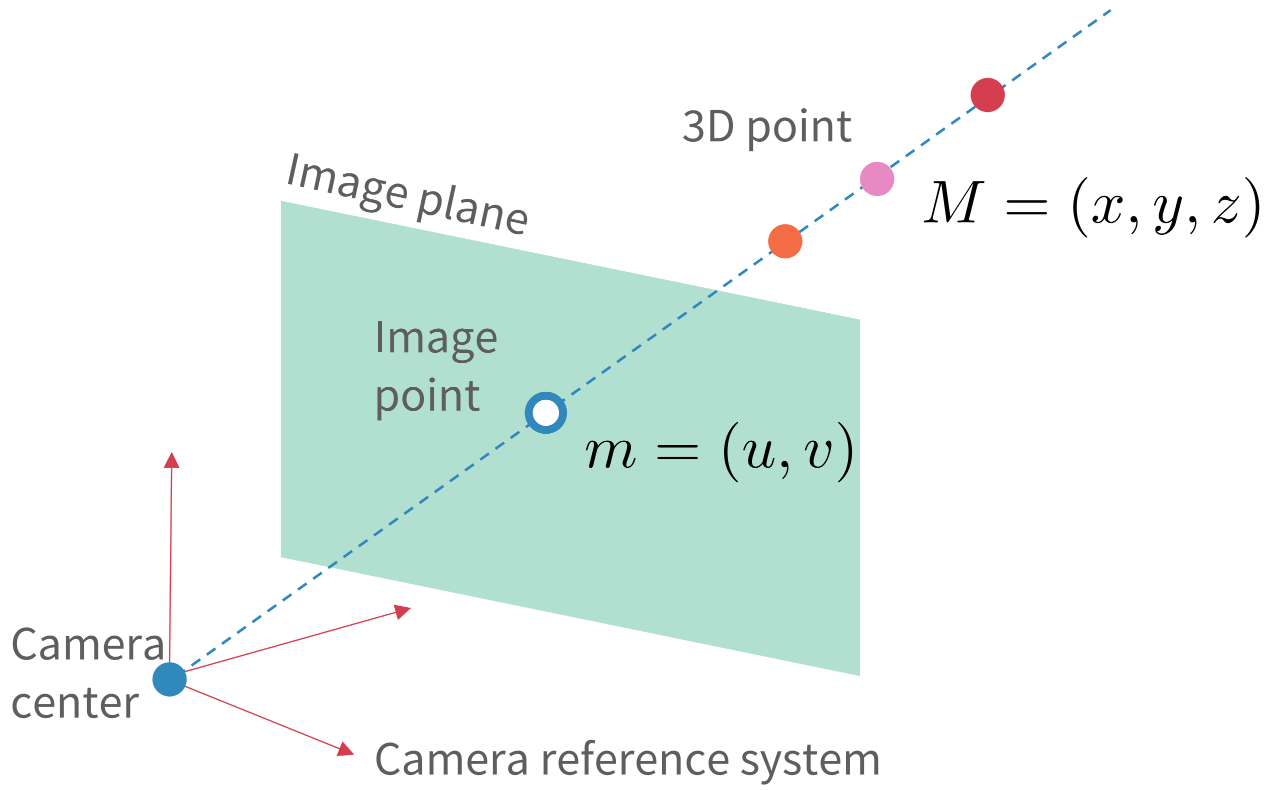


Depth from a single image is an ill posed problem

The capture of an image of a 3D scene is modelled as the **projection** of 3D points on a 2D image plane.

All the points belonging to the optical ray projects on the same 3D points.

It is not possible to recover the depth of a point from a single image, as the **same image point** can be back-projected to **multiple plausible depths**.

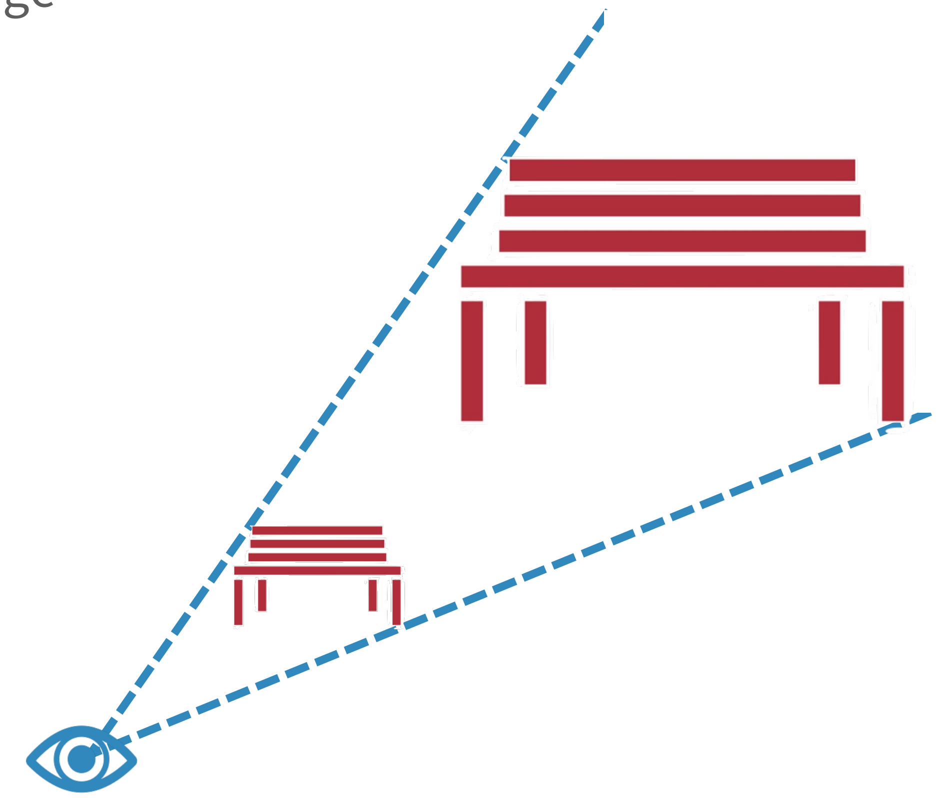


Estimating depth form a single image is an ill posed problem!

Estimating depth form a single image is an ill posed problem!

Scale-depth ambiguity:

a small close object looks the same as a much larger one further away.
Absolute scale / depth is ambiguous from a single image

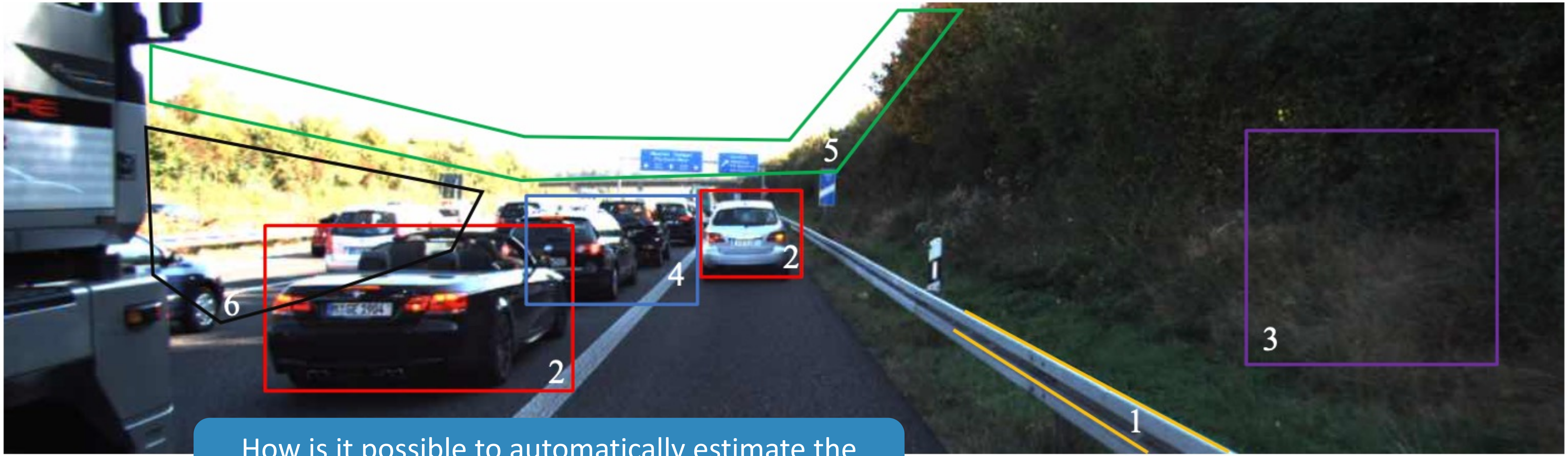


Thank for your attention!

…but humans succeed estimating depth

however not all the 3D structures are equally likely! Humans are able to infer a (nearly) correct 3D structure and relative depth, using prior experience and visual cues such as:

1. Linear perspective
2. Relative size position
3. Texture gradient
4. Occlusions (the occluded object is far away)
5. Aerial perspective
6. Light and shadows
7. Blur/defocus



How is it possible to automatically estimate the depth form a single image?

Today menu

Depth estimation from a **single image**

- Supervised methods
- Visual cues for single image depth estimation

Depth estimation from a **calibrated stereo pairs**

- Stereo self-supervision

Multiview depth estimation

- Mono-depth training & deep SfM

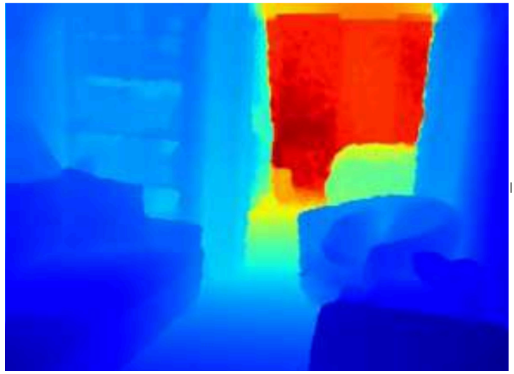
Aims:

Get an intuition of the main approach for inferring 3D data from 2D images

Fill our box with the geometric tools necessary to depth estimation

Supervised approach

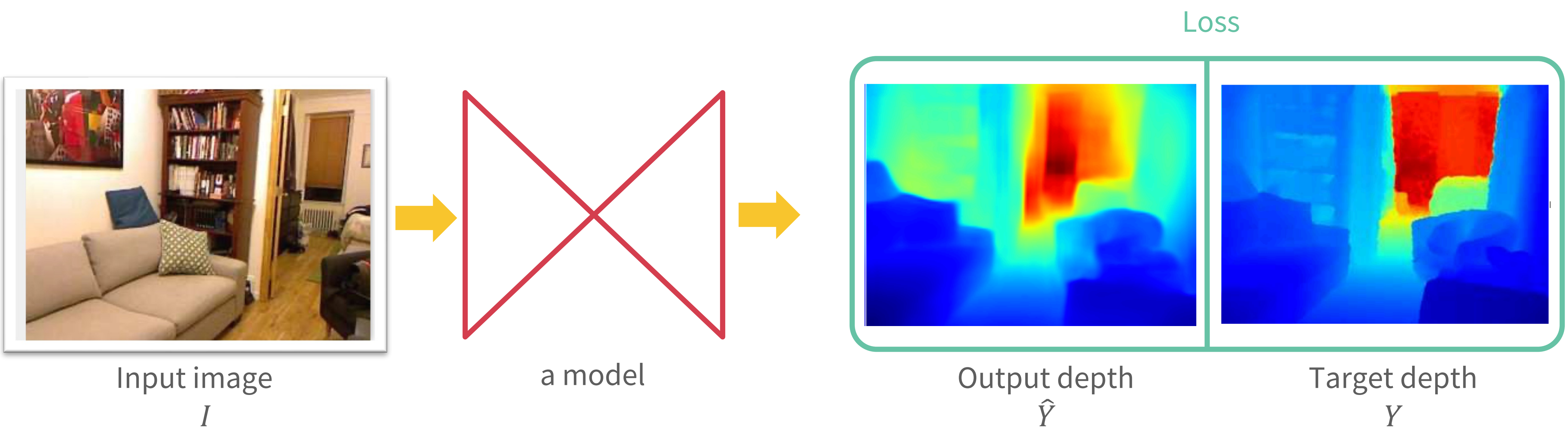
Input image
I



Target depth
Y

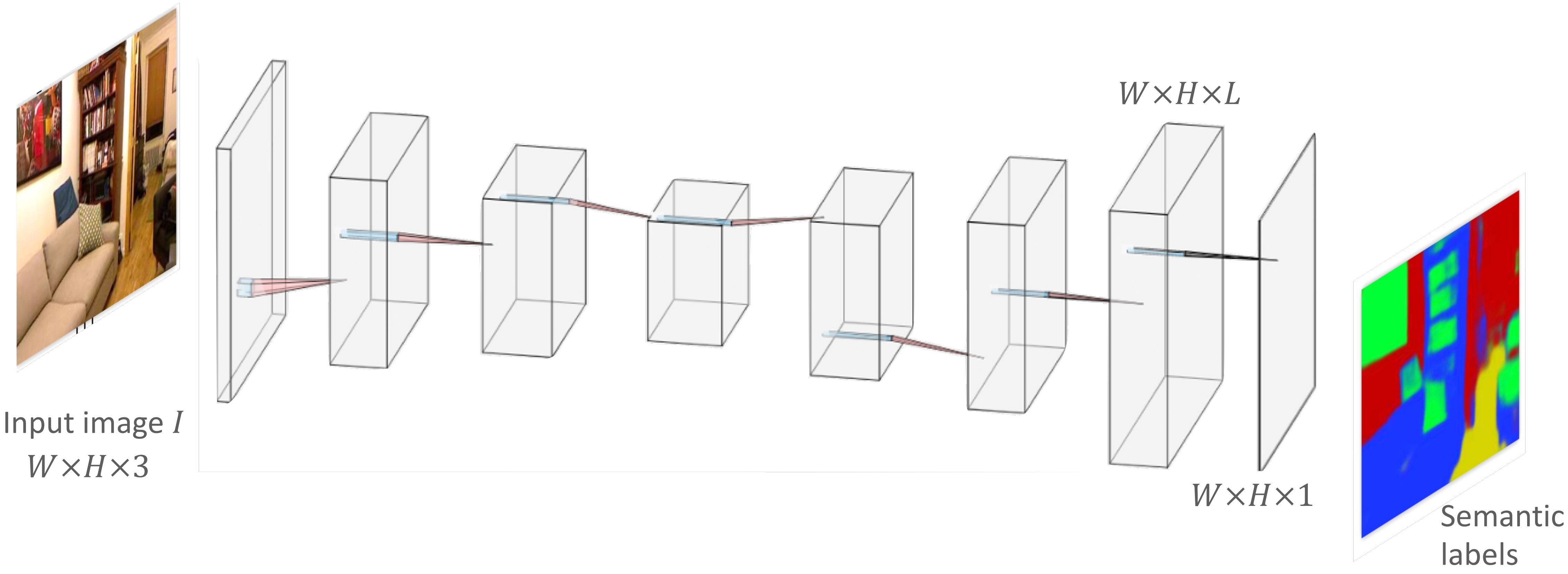
Our training set is a collection of images and depth maps (RGD images)
$TR = \{(I_i, Y_i)\}$

Supervised approach



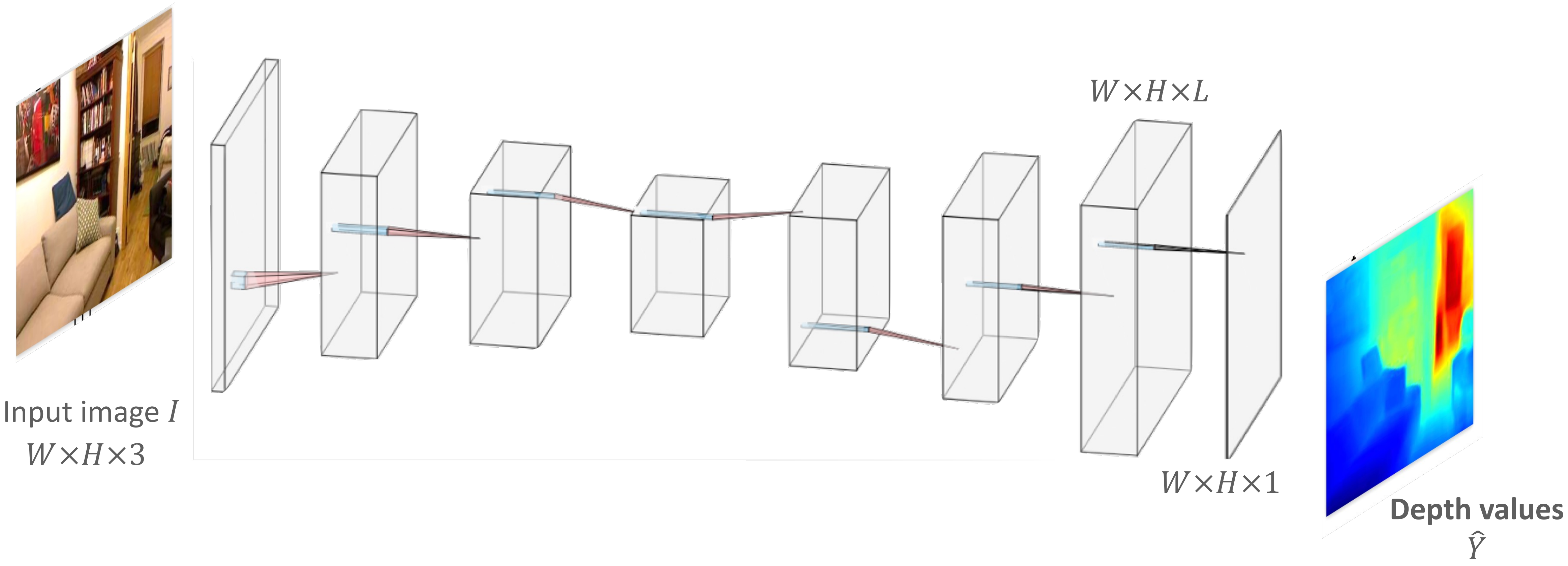
From semantic segmentation to depths

A simple solution is to start with a Fully Convolutional Neural Networks as the one used for **semantic segmentation**



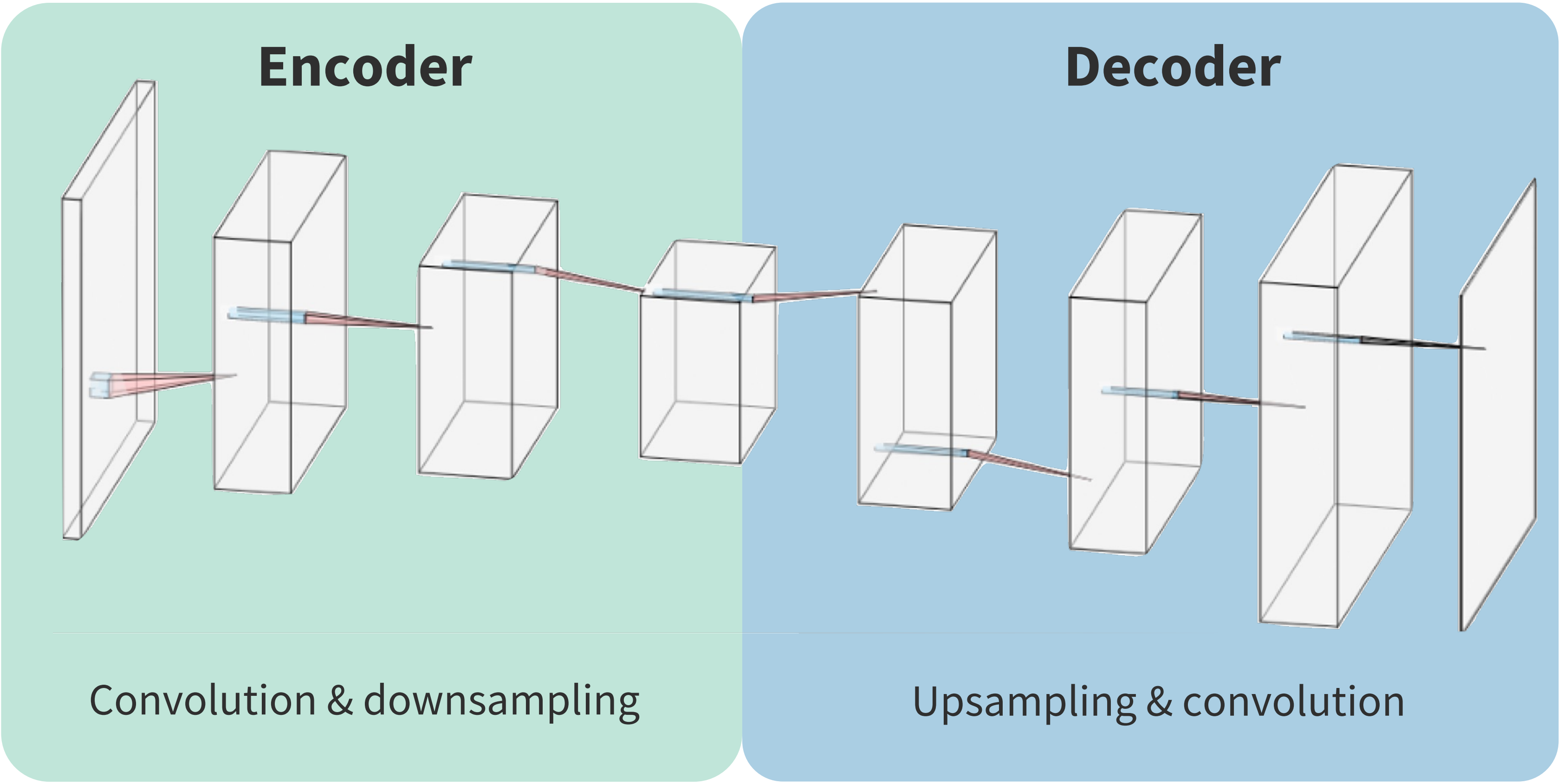
From semantic segmentation to depths

A simple solution is to start with a Fully Convolutional Neural Networks as the one used for **semantic segmentation** and **adapt** it to predict depth values instead of semantic labels.



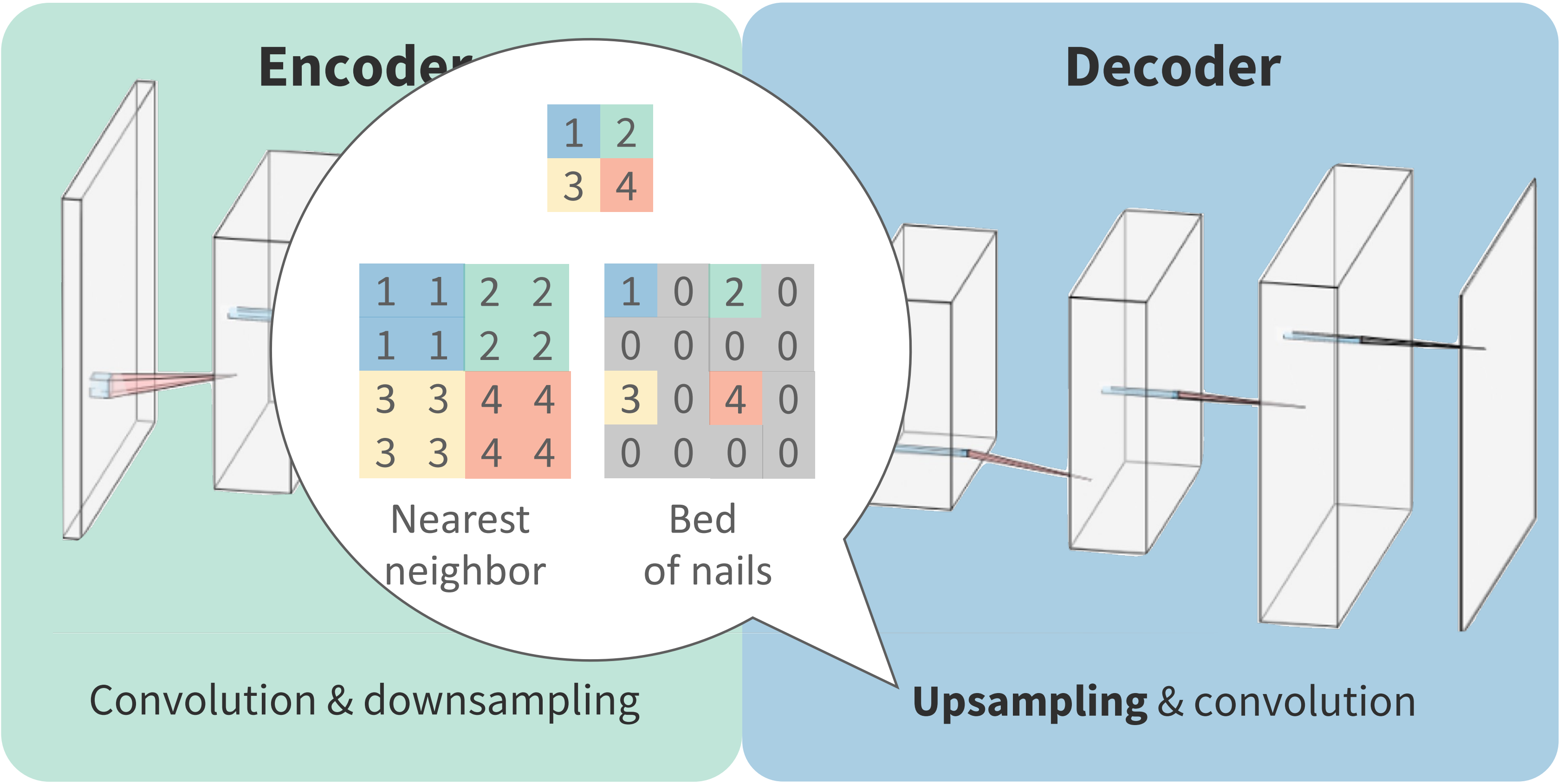
Encoder & decoder (sketch of the idea)

- The **encoder** reduces the spatial extent of the image and produce deeper features that enode richer information
- The **decoder** upsamples the predictions to cover each pixel in the image at the original resolution



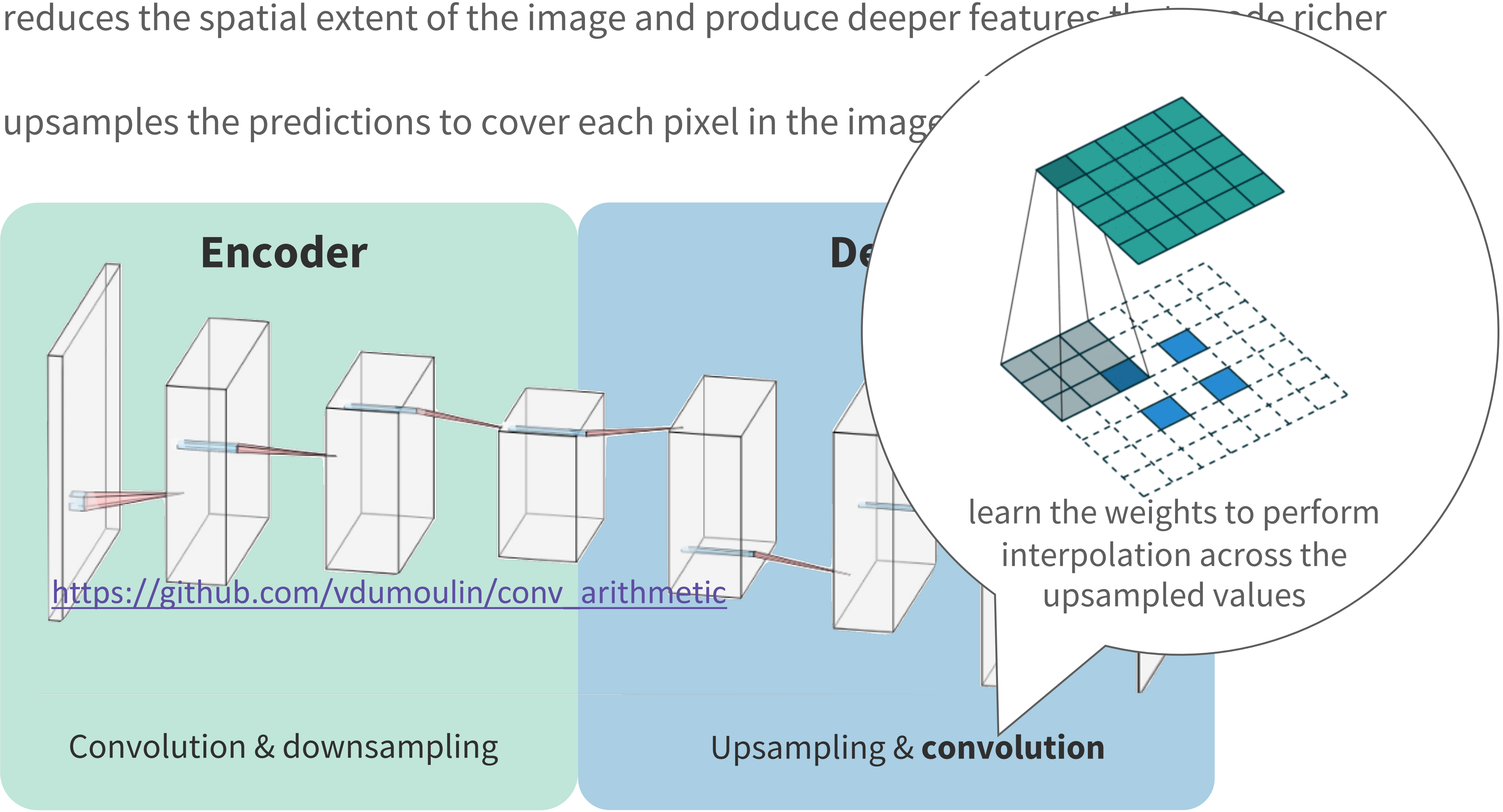
Encoder & decoder (sketch of the idea)

- The **encoder** reduces the spatial extent of the image and produce deeper features that enode richer information
- The **decoder** upsamples the predictions to cover each pixel in the image at the original resolution



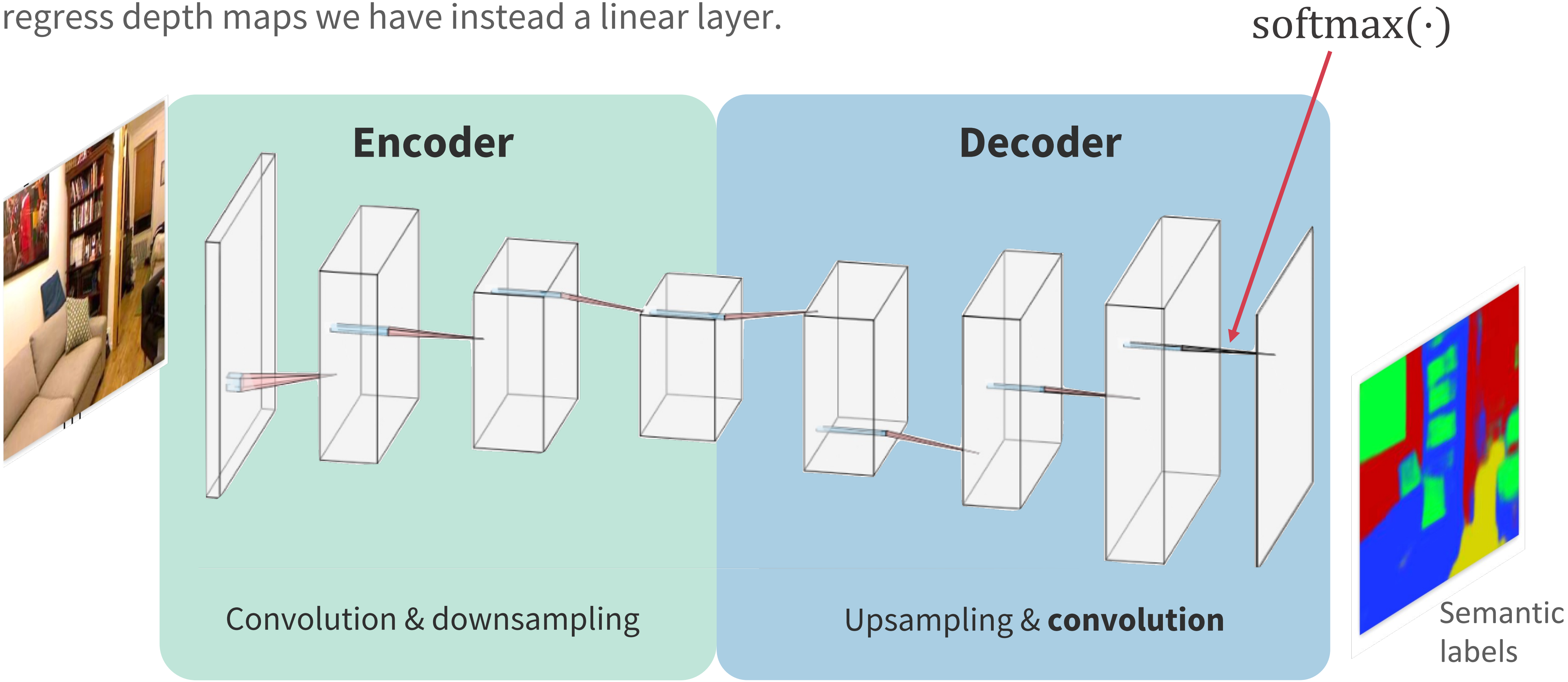
Encoder & decoder (sketch of the idea)

- The **encoder** reduces the spatial extent of the image and produce deeper features th… …de richer information
- The **decoder** upsamples the predictions to cover each pixel in the image



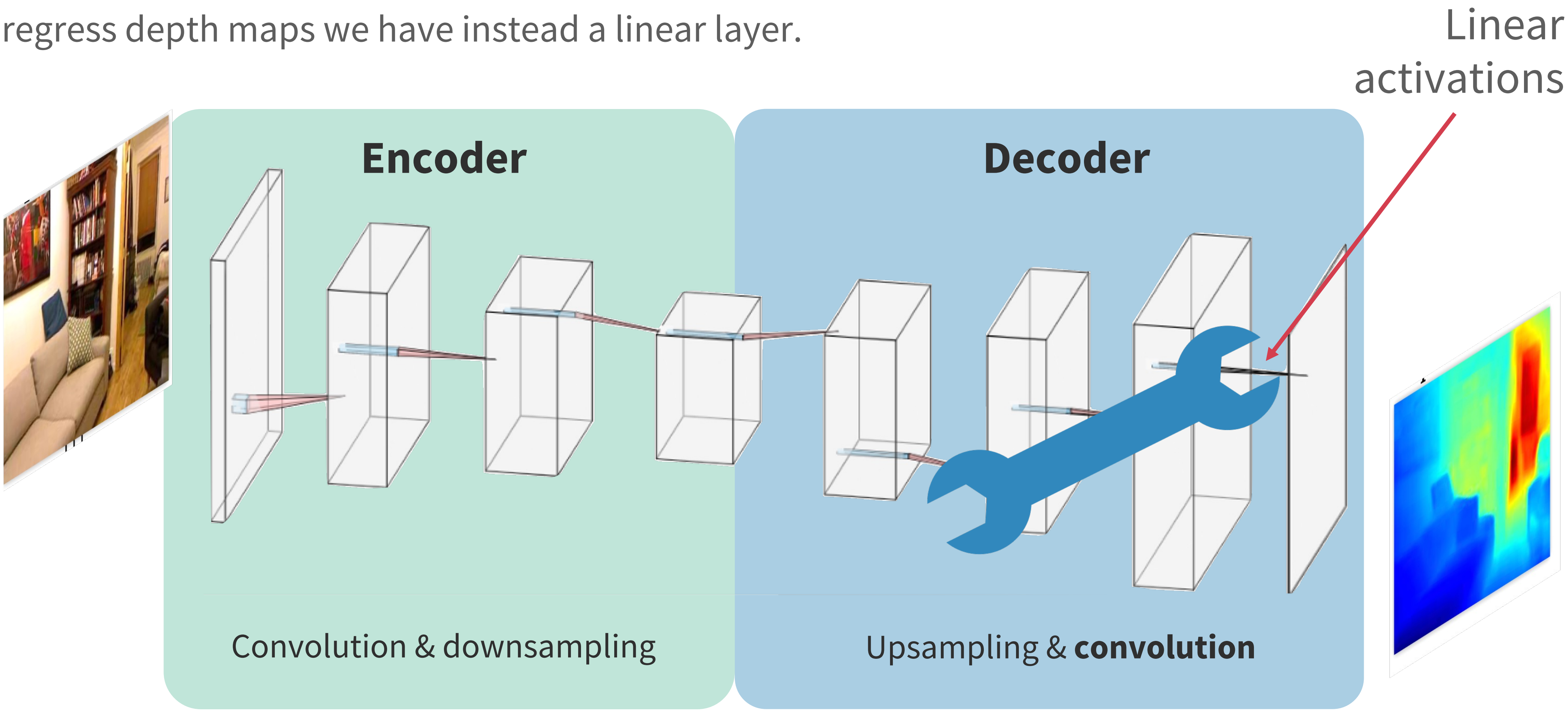
Encoder & decoder (sketch of the idea)

- In semantic segmentation a soft-max to predict the proability of belonging to one of the L classes $\Lambda = \{l_1, \dots, l_L\}$,
- To regress depth maps we have instead a linear layer.



Encoder & decoder (sketch of the idea)

- In semantic segmentation a soft-max to predict the proability of belonging to one of the L classes $\Lambda = \{l_1, \dots, l_L\}$,
- To regress depth maps we have instead a linear layer.



Other architectures

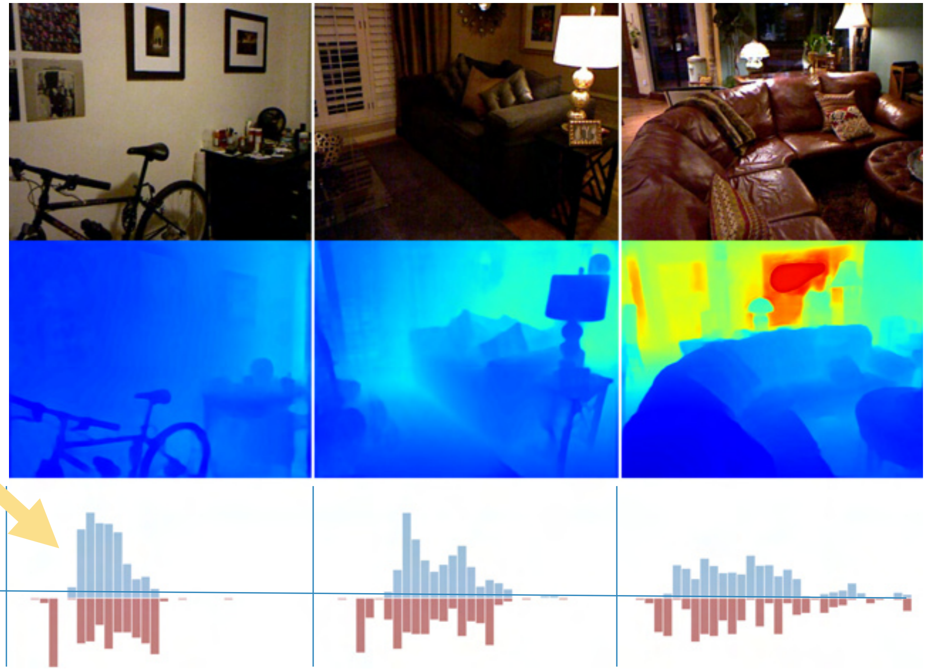
Most of the architectures for supervised depth estimation from a single image are based on this encoder-decoder paradigm, keeping this in mind you should be able to navigate the literature...

e.g., Adabins

Encoder | Decoder + Adaptive bin

a transformer- based architecture block that divides the depth range into bins whose center value is estimated adaptively per image

For close-up images bins are concentrated near smaller values



Ground truth Histograms

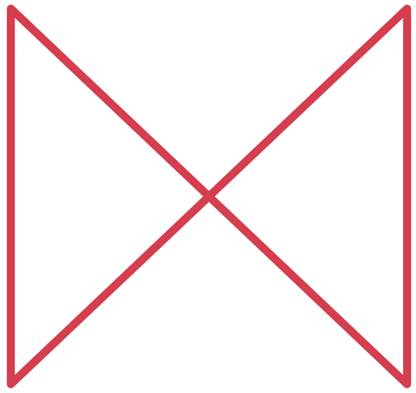
Predicted Histograms

Bhat, Shariq Farooq, Ibraheem Alhashim, and Peter Wonka. "Adabins: Depth estimation using adaptive bins." CVPR 2021.

A naïve loss

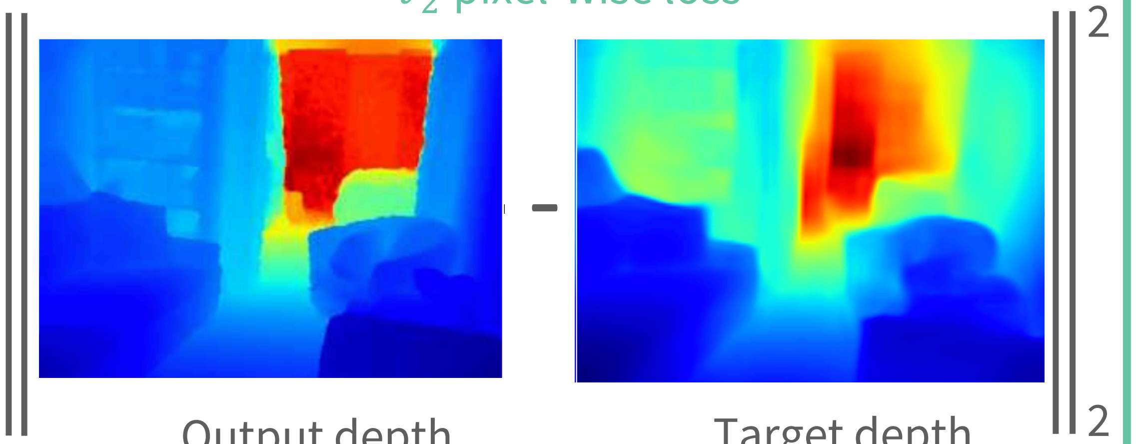
As a loss, we could minimize the ℓ_2 norm between the predicted and the regressed values.

Input image



a model

ℓ_2 pixel-wise loss



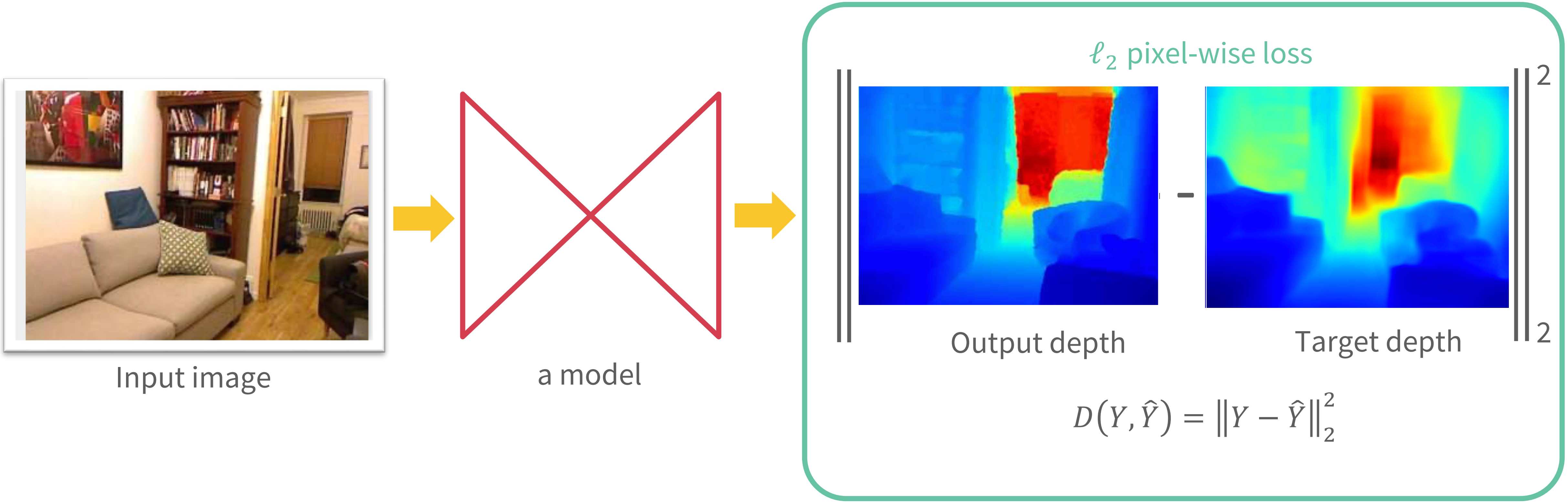
Output depth

Target depth

$$D(Y, \hat{Y}) = \|Y - \hat{Y}\|_2^2$$

A naïve loss

As a loss, we could minimize the ℓ_2 norm between the predicted and the regressed values.



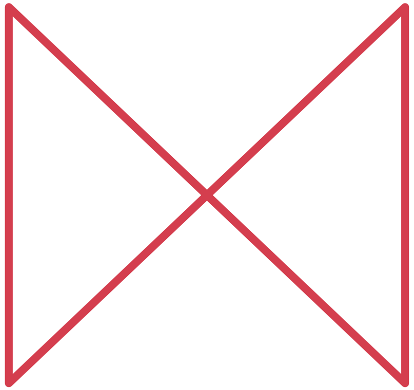
But this wouldn't work well because of the **depth-scale ambiguity**!

To predict depth effectively, we must consider the **geometric nature** of the problem

A naïve loss

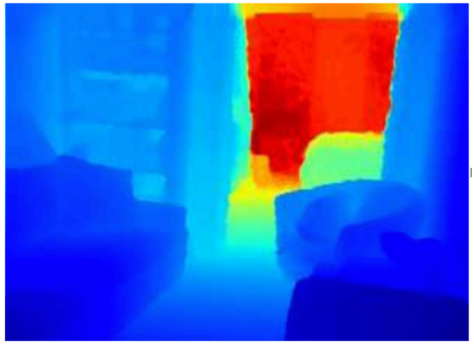
As a loss, we could minimize the ℓ_2 norm between the predicted and the regressed values.

Input image



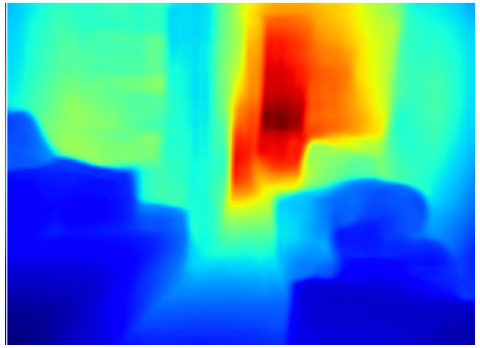
a model

Scale invariant loss



Output depth

-



Target depth

$$D(y, y^*) = D(\lambda y, y^*)$$

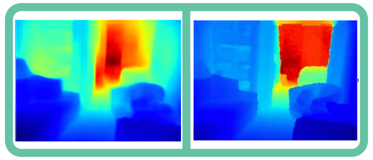
Back to the roots: a CNN for single image depth estimation

[Eigen et al, NIPS 2014] is a milestone in single image depth estimation, being the first work leveraging a CNN to estimate the depth in a *supervised* way.

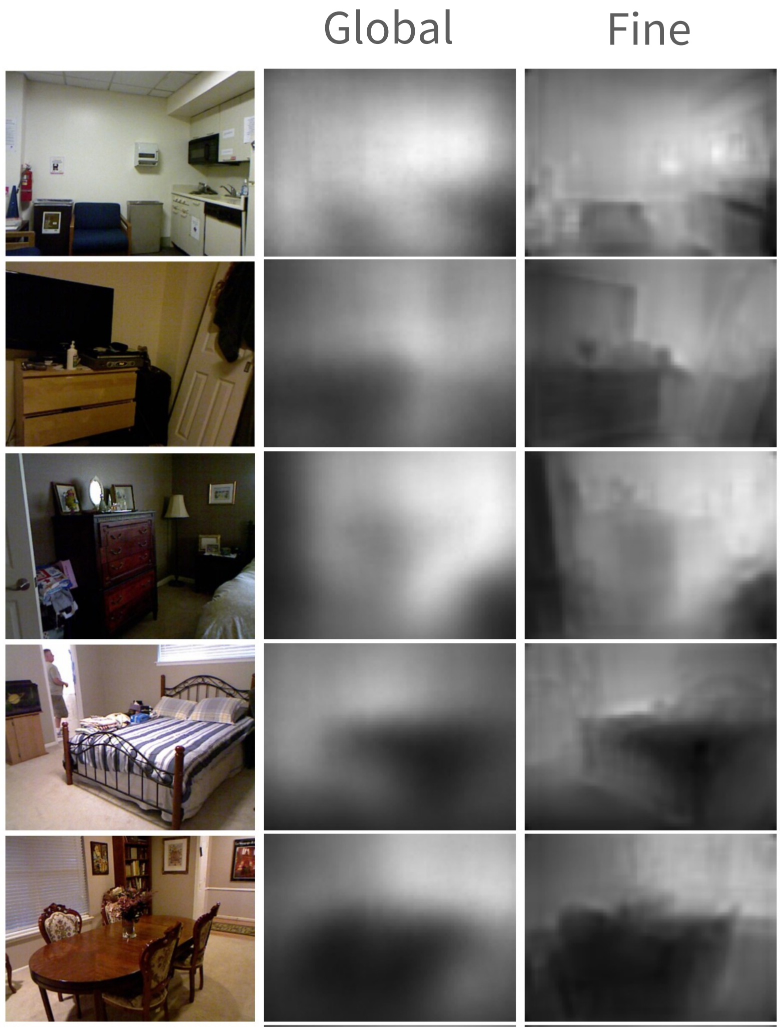
- While stereo methods rely on local disparity (see later), in single image depth estimation a **global view** is needed to relate all the available visual cues.

Two-branch architecture: global and fine scale

- One of the major ambiguity is the global scale of the scene (moderate variations in room furniture and size).



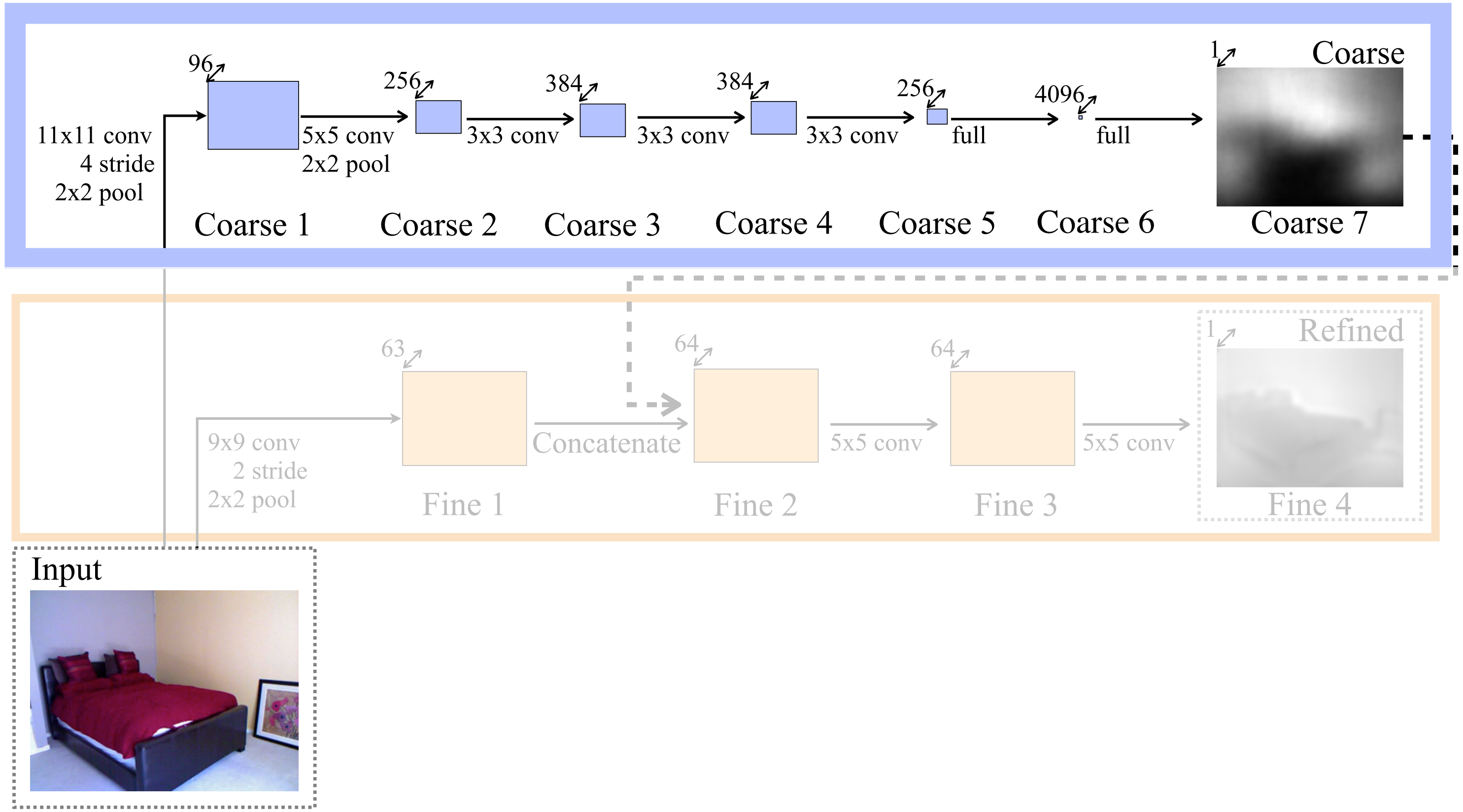
Scale invariant loss



Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Model architecture

Two branches estimate a coarse and a refined depth maps, the former used as to ease the prediction of the latter.



Coarse-scale Network:

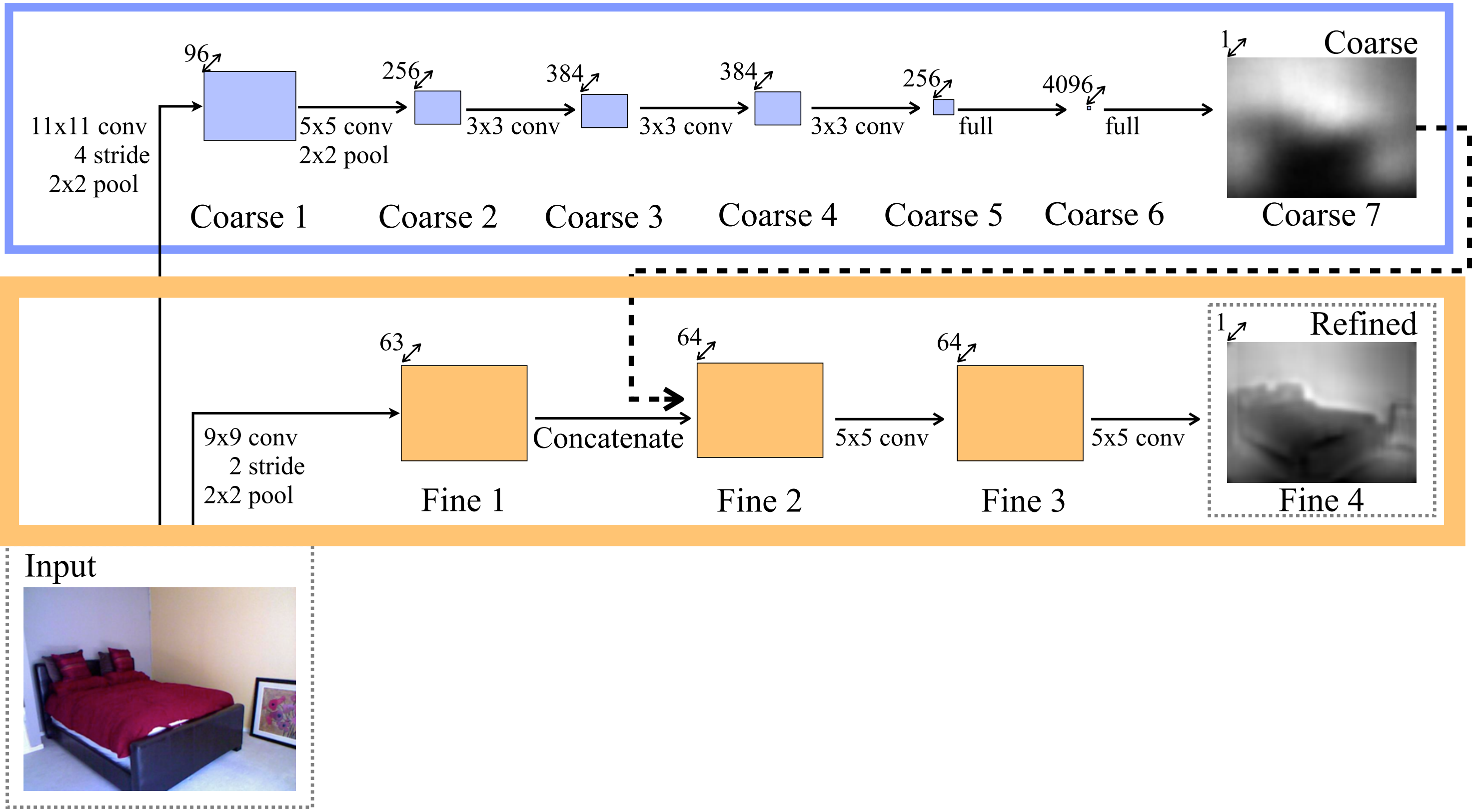
Global understanding thanks to:

- Max pooling to combine information from different part of the image
- Fully connected layers to contain the entire image in their receptive field.
- Layers 1-5 pretrained on ImageNet

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Model architecture

Two branches estimate a coarse and a refined depth maps, the former used as to ease the prediction of the latter.



Fine-scale Network:

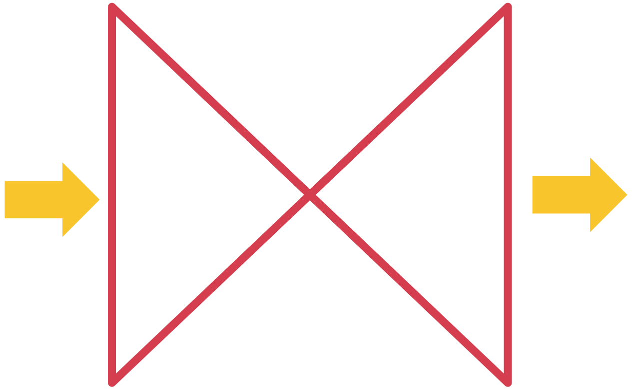
Local refinement to align coarse prediction to local details such as object and walls.

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Scale invariant loss

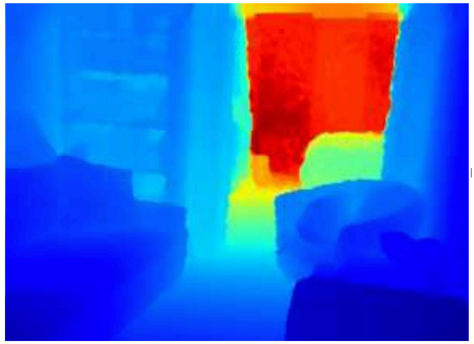
The scale invariant loss focus on the spatial relations within a scene rather than on a global scale:

Input image



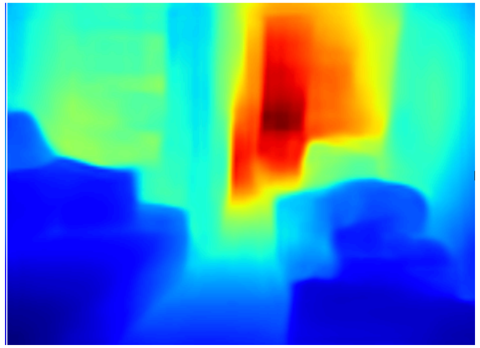
Two-branch architecture

Scale invariant loss



Output depth

-



Target depth

$$D(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log y_i^* + \alpha(y, y^*)\right)^2$$

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Scale invariant loss

The scale invariant loss focus on the spatial relations within a scene rather than on a global scale:

$$D(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log y_i^* + \alpha(y, y^*)\right)^2$$

Mean squared error

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Scale invariant loss

The scale invariant loss focus on the spatial relations within a scene rather than on a global scale:

$$D(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log y_i^* + \alpha(y, y^*)\right)^2$$

Mean squared error

Log space: reduces the impact of large depth values overpowering the smaller ones in error calculation.

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Scale invariant loss

The scale invariant loss focus on the spatial relations within a scene rather than on a global scale:

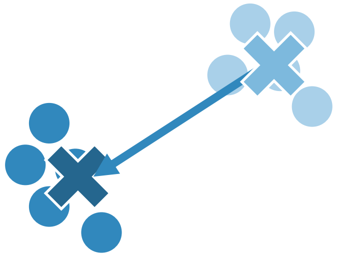
$$D(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log y_i^* + \alpha(y, y^*)\right)^2$$

Mean squared error

Log space: reduces the impact of large depth values overpowering the smaller ones in error calculation.

Scale invariant: $\alpha(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}(\log y_i - \log y_i^*)$ is the value that minimizes the error for a given pair (y, y^*). For any prediction y, e^{α} is the scalar that best align it to the groundtruth.

All scalar multiples of y have the same error, here the scale invariance.



Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Scale invariant

The scale invariant loss focus on the spatial relations within a scene rather than on a global scale:

$$D(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log y_i^* + \alpha(y, y^*)\right)^2$$

Mean squared error

The loss has other equivalent forms, i.e:

$$D(y, y^*) = \frac{1}{n^2}\sum_{i,j}\left(\left(\log y_i - \log y_j\right) - \left(\log y_i^* - \log y_j^*\right)\right)^2$$

to have low error, each pair of pixel in the prediction must differ in depth by an amount that is comparable to the ground truth. Setting $d_i = \log y_i - \log y_j$, we have

$$D(y, y^*) = \frac{1}{n}\sum_i d_i^2 - \frac{1}{n}\left(\sum_i d_i\right)^2$$

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Scale invariant loss mean squared error in log space

The scale invariant loss focus on the spatial relations within a scene rather than on a global scale:

$$D(y, y^*) = \frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log y_i^* + \alpha(y, y^*)\right)^2$$

Mean squared error

The loss has other equivalent forms, i.e:

$$D(y, y^*) = \frac{1}{n^2}\sum_{i,j}\left(\left(\log y_i - \log y_j\right) - \left(\log y_i^* - \log y_j^*\right)\right)^2$$

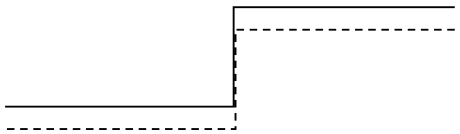
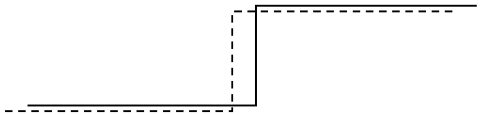
to have low error, each pair of pixel in the prediction must differ in depth by an amount that is comparable to the ground truth. Setting $d_i = \log y_i - \log y_j$, we have

$$D(y, y^*) = \frac{1}{n}\sum_{i} d_i^2 - \frac{\lambda}{n}\left(\sum_{i} d_i\right)^2$$

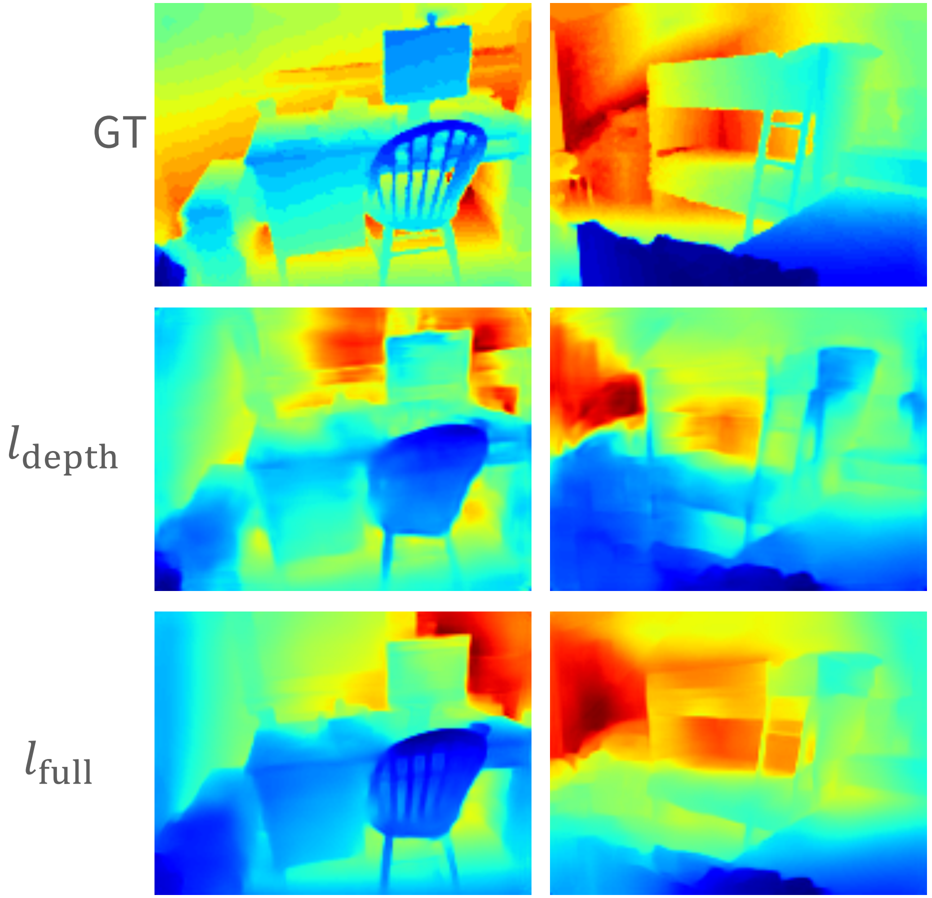
The loss used during training is a linear combination $\lambda = 0.5$ of the mean square error and the invariant loss

Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Other losses

sensitivity	l_{depth}	l_{grad}	l_{normal}
	✓	✗	✗
	✗	✓	✓
	✗	✓	✓

	RMS	REL	$\delta < 1.25$	F1
w/ l_{depth}	0.580	0.133	0.830	0.525
w/ $l_{\text{depth}} + \lambda l_{\text{grad}}$	0.563	0.128	0.841	0.543
w/ full loss	**0.555**	**0.126**	**0.843**	**0.548**



J. Hu, et al. Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In WACV, 2019.

Loss on gradients

Let $\nabla_x d_i$ and $\nabla_y d_i$ be the horizontal and vertical image gradients of the log difference $d_i = \log y_i - \log y_i^*$

$$l_{\text{grad}} = \frac{1}{n}\sum[(\nabla_x d_i)^2 + \left(\nabla_y d_i\right)^2]$$

compares image gradients of the prediction with the ground truth.

This encourages predictions to have

- close values,
- but also similar local structure,

resulting in depthmaps that better follow depth gradients, with no degradation in measured l2 performance.

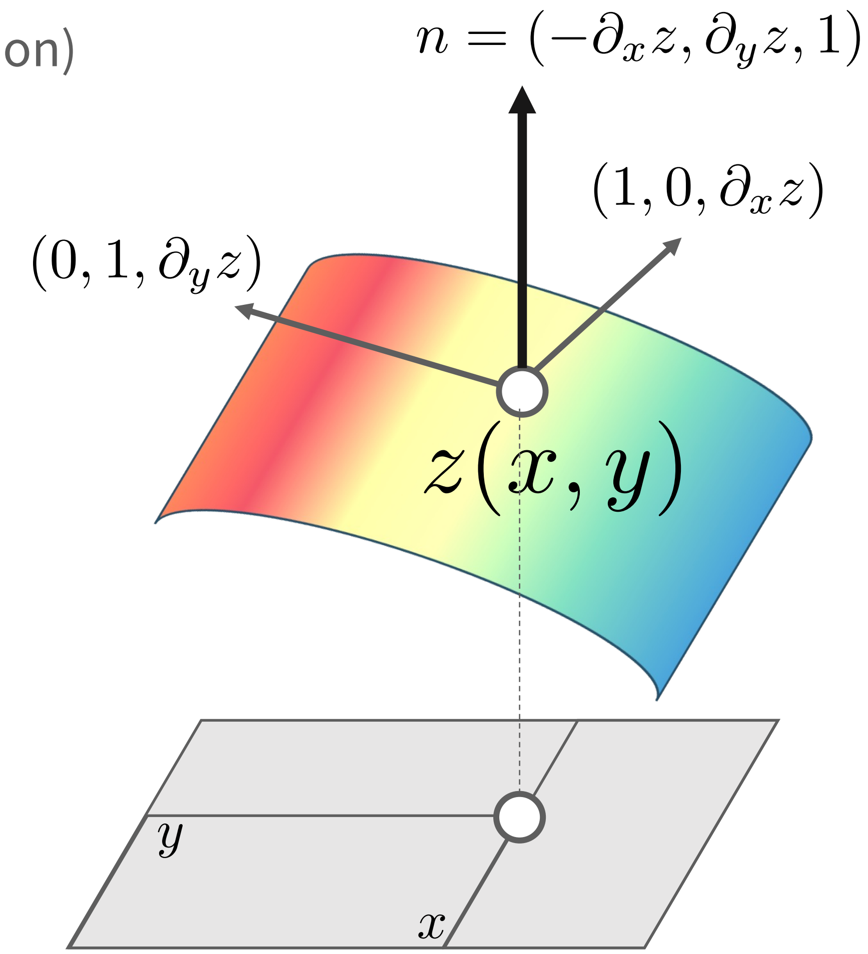
Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

Loss on normals

Depth can be seen as a surface $z = z(x, y)$ defined on the image pixel gird (x, y)

The normal in (x, y) can be computed from the depth by taking the cross product between the tangent vectors to $z(x, y)$.

(Viceversa, from normals it is possible to retrieve the depth by integration)



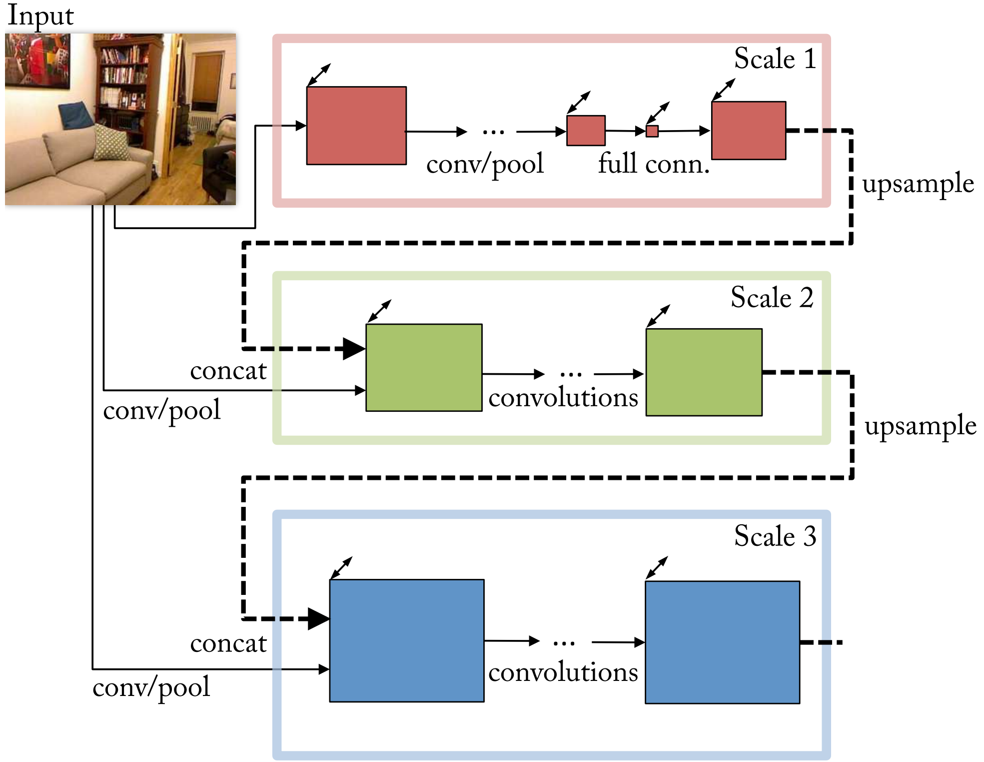
Let n and n^* be the estimated and the predicted normals

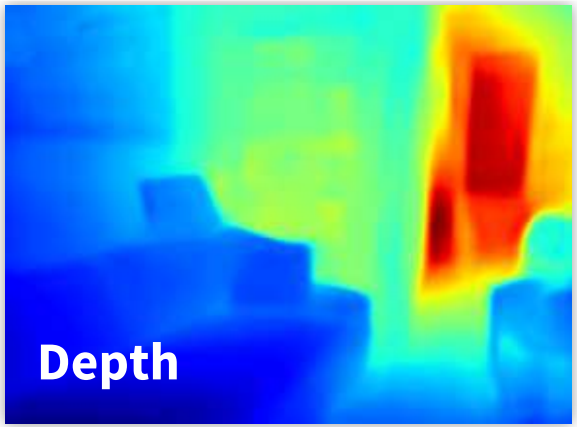
$$l_{norm} = \cos(n, n^*)$$

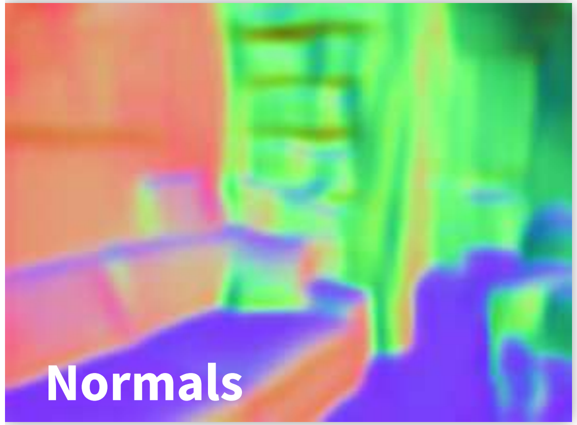
Predicting depth, normal and semantic labels

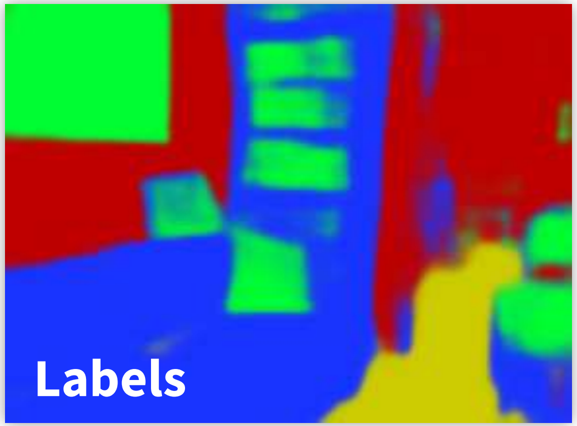
According to the loss employed,
the **very same network** can be used to regress:

- Depth
  (loss on depth and gradients)
- Normals
  (loss on normals)
- Semanitc labels
  (per pixel cross entropy)





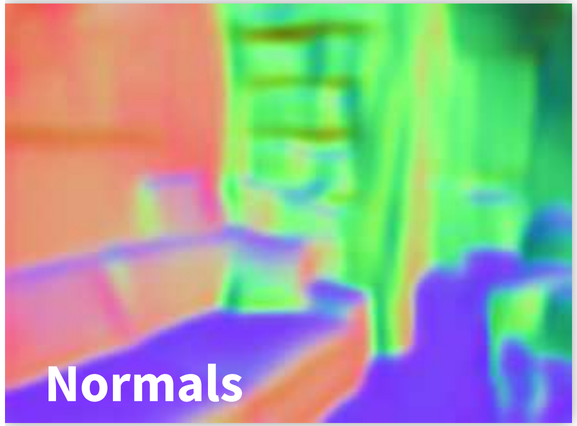


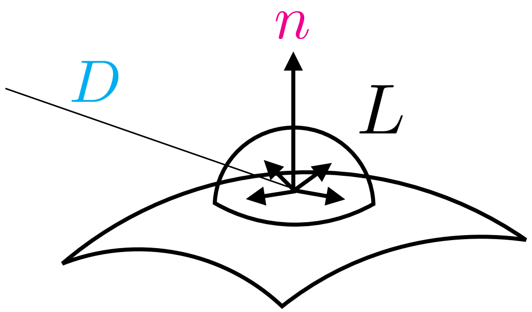


Eigen et al. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network, NIPS 2014

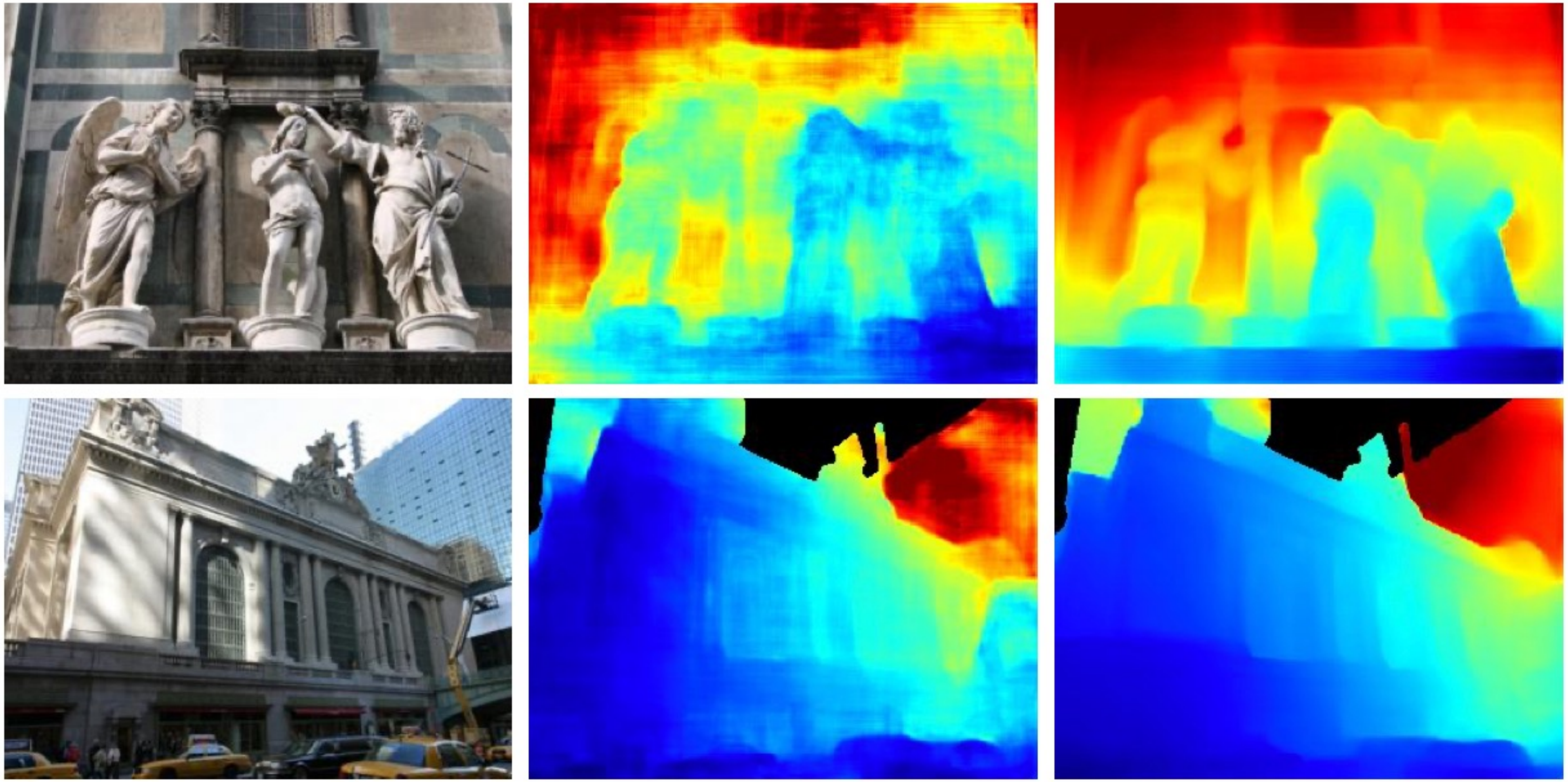
Predicting depth, normal and semantic labels

Inferring normals from a **single** image typically requires several images of the same scene acquired under varying and controlled illumination conditions (this problem is known as **Photometric Stereo**).





The benefits of combining losses

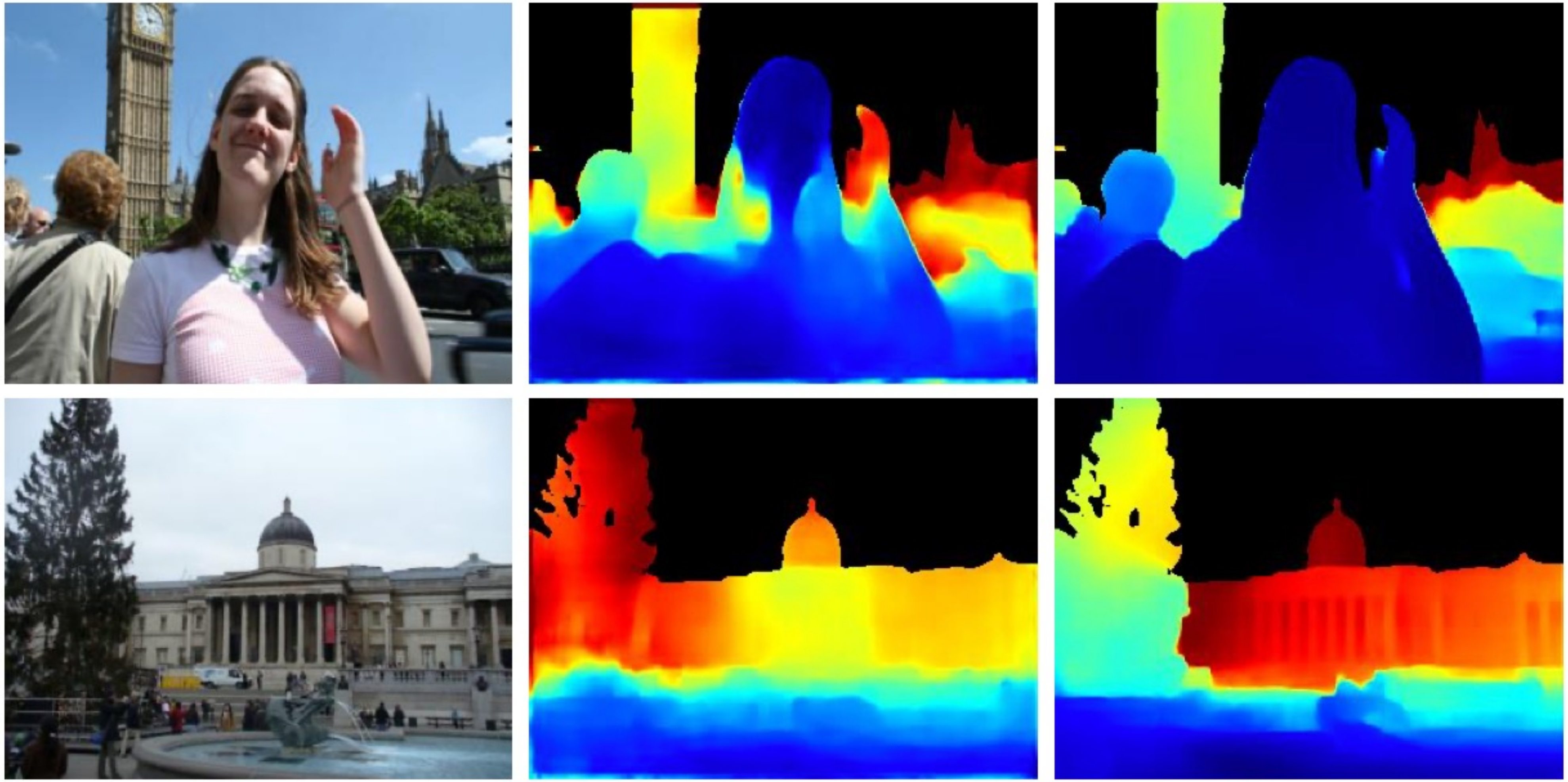


Input photo | Output w/o $\mathcal{L}_{\text{grad}}$ | Output w/ $\mathcal{L}_{\text{grad}}$

Zhengqi Li and Noah Snavely. "MegaDepth: Learning Single-View Depth Prediction from Internet Photos» CVPR18.

The benefits of combining losses

Annotation



Input photo   Output w/o $\mathcal{L}_{ord}$ Output w/ $\mathcal{L}_{ord}$

Zhengqi Li and Noah Snavely. "MegaDepth: Learning Single-View Depth Prediction from Internet Photos» CVPR18.

Evaluation metrics

- Accuracy scores: the percentage of pixels having a relative error δ lower than a threshold ϵ (typical values of ϵ are $1{,}25$, 1.25^2, 1.25^3)

$$\delta = \max\left(\frac{y}{y^*}, \frac{y^*}{y}\right) < \epsilon$$

- Absolute Relative error: to normalize per-pixel errors according to real depth, reducing the impact of large errors with the distance

$$\frac{1}{n}\sum |y_i - y_i^*|/y_i^*$$

- Squared Relative Error: to penalize larger depth errors (e.g. near discontinuities)

$$\frac{1}{n}\sum ||y_i - y_i^*||^2/y_i^*$$

- Root Mean Squared Error: $\sqrt{\frac{1}{n}\sum ||y_i - y_i^*||^2}$

- Roor Mean Squared Logarithmic Error: $\sqrt{\frac{1}{n}\sum ||\log y_i - \log y\_i^\wedge * ||^2}$

3D data for supervision

A training set

$$TR = \{(I_i, , Y_i)\}$$

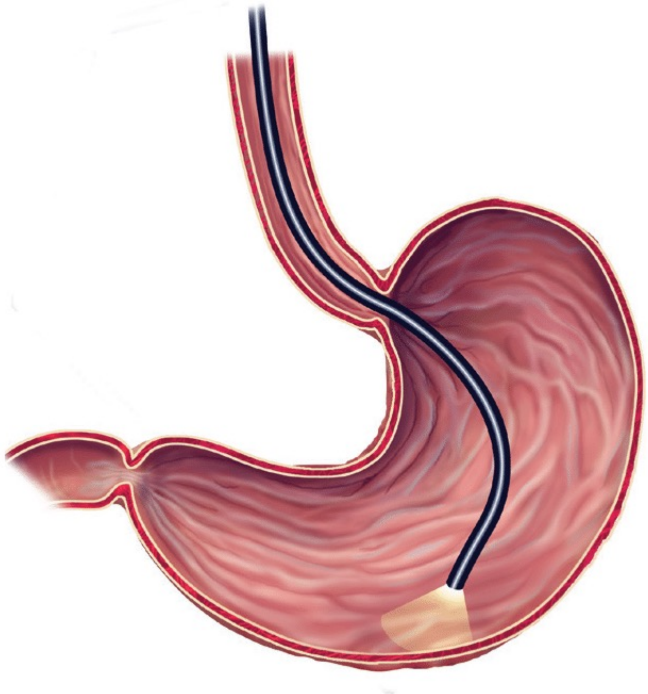
of RGB-D image can be acquired using dedicated sensors.

Why depth from a single image?

What is the advantage of having a deep network (4K€ for a GPU) that works on a single image rather than using a sensor that costs a few hundred €?

There are good reasons… would you rather have a Kinect or a small endoscopic probe in your belly?

We can rely on deep network when acquiring 3D data is not possible!



The need of 3D data for supervision (and where to find them)

Deep learning method have recently driven significant progress in supervised depth estimation from a single image, but being entirely data driven, their potential grows with the amount of data using during training.

Popular datasets:

Dataset	Indoor	Outdoor	Dynamic	Video	Dense	Accuracy	Diversity	Annotation	Depth	# Images
DIML Indoor [31]	✓			✓	✓	Medium	Medium	RGB-D	**Metric**	220K
MegaDepth [11]		✓	(✓)		(✓)	Medium	Medium	SfM	No scale	130K
ReDWeb [32]	✓	✓	✓		✓	Medium	**High**	Stereo	No scale & shift	3600
WSVD [33]	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	1.5M
3D Movies	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	75K
DIW [34]	✓	✓	✓			Low	**High**	User clicks	Ordinal pair	496K
ETH3D [35]	✓	✓			✓	**High**	Low	Laser	**Metric**	454
Sintel [36]	✓	✓	✓	✓	✓	**High**	Medium	Synthetic	(Metric)	1064
KITTI [28], [29]		✓	(✓)	✓	(✓)	Medium	Low	Laser/Stereo	**Metric**	93K
NYUDv2 [30]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	407K
TUM-RGBD [37]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	80K

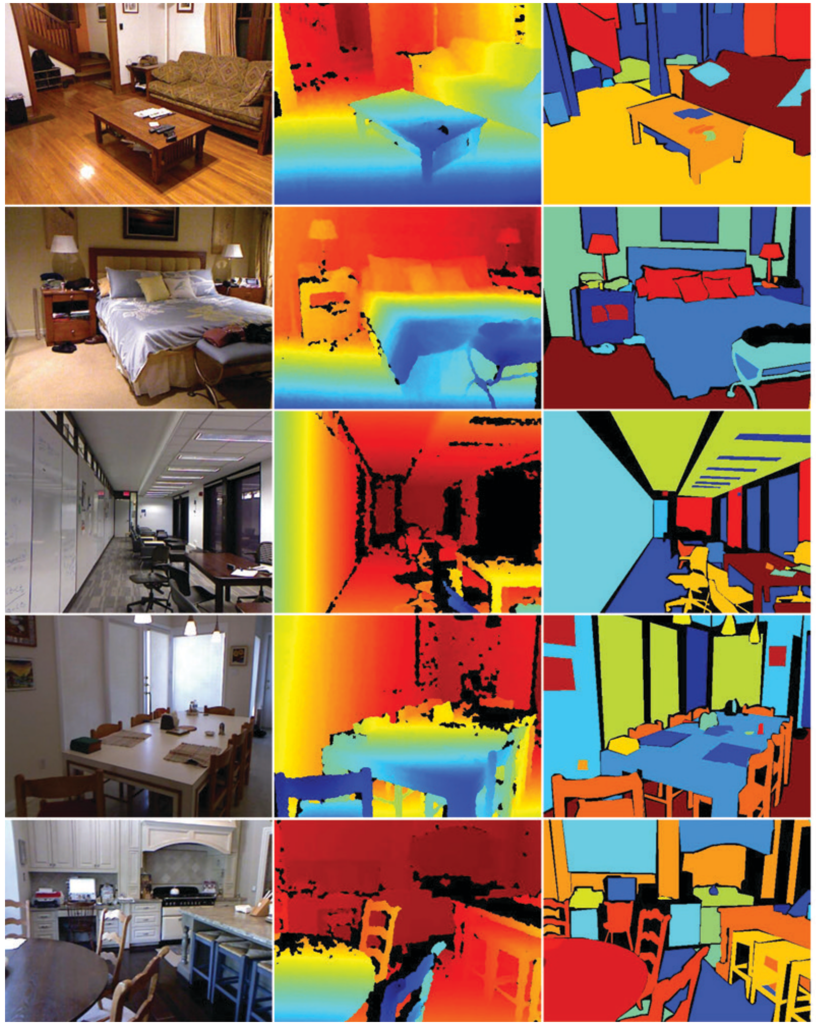
Ranftl et al. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer. TPAMI 2020

NYU v2

Video sequences form a variety of indoor scenes acquired with a Microsoft Kinect to record both the RGB and Depth Map.

- 1449 densely labeled pairs of aligned RGB and depth images
- 464 new scenes taken from 3 cities
- 407,024 new unlabeled frames
- Each object is labeled with a class and an instance number (cup1, cup2, cup3, etc)

Dataset	Indoor	Outdoor	Dynamic	Video	Dense	Accuracy	Diversity	Annotation	Depth	# Images
DIML Indoor [31]	✓			✓	✓	Medium	Medium	RGB-D	**Metric**	220K
MegaDepth [11]		✓	(✓)		(✓)	Medium	Medium	SfM	No scale	130K
ReDWeb [32]	✓	✓	✓		✓	Medium	**High**	Stereo	No scale & shift	3600
WSVD [33]	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	1.5M
3D Movies	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	75K
DIW [34]	✓	✓	✓			Low	**High**	User clicks	Ordinal pair	496K
ETH3D [35]	✓	✓			✓	**High**	Low	Laser	**Metric**	454
Sintel [36]	✓	✓	✓	✓	✓	**High**	Medium	Synthetic	(Metric)	1064
KITTI [28], [29]		✓	(✓)	✓	(✓)	Medium	Low	Laser/Stereo	**Metric**	93K
NYUDv2 [30]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	407K
TUM-RGBD [37]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	80K



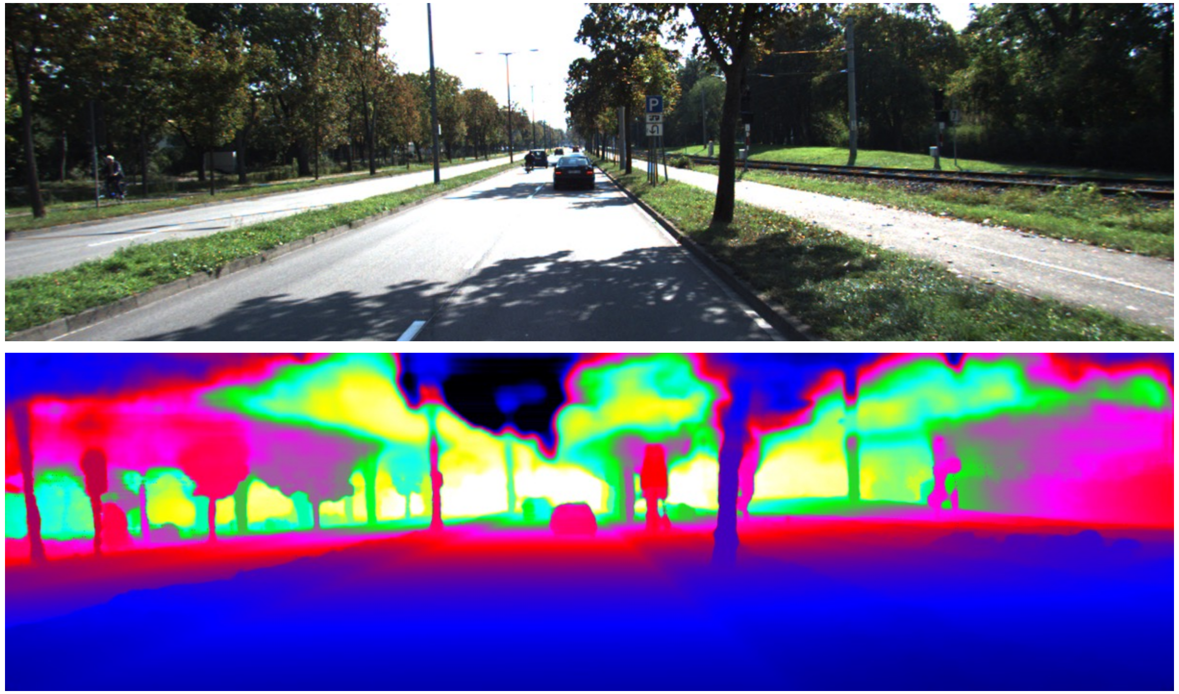
https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html

KITTI dataset

A large collection of images acquired in driving environment.

3D point cloud acquired by a LiDAR registered with RGB.

A subset of 200 images (KITTI 2015 stereo training set) with accurate ground truth (moving object replaced with accurate cad model)



Dataset	Indoor	Outdoor	Dynamic	Video	Dense	Accuracy	Diversity	Annotation	Depth	# Images
DIML Indoor [31]	✓			✓	✓	Medium	Medium	RGB-D	**Metric**	220K
MegaDepth [11]		✓	(✓)		(✓)	Medium	Medium	SfM	No scale	130K
ReDWeb [32]	✓	✓	✓		✓	Medium	**High**	Stereo	No scale & shift	3600
WSVD [33]	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	1.5M
3D Movies	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	75K
DIW [34]	✓	✓	✓			Low	**High**	User clicks	Ordinal pair	496K
ETH3D [35]	✓	✓			✓	**High**	Low	Laser	**Metric**	454
Sintel [36]	✓	✓	✓	✓	✓	**High**	Medium	Synthetic	(Metric)	1064
KITTI [28], [29]		✓	(✓)	✓	(✓)	Medium	Low	Laser/Stereo	**Metric**	93K
NYUDv2 [30]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	407K
TUM-RGBD [37]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	80K

https://www.cvlibs.net/datasets/kitti/

MegaDepth

A large scale dataset generated from Internet photo collections for a set of well-photographed landmarks.

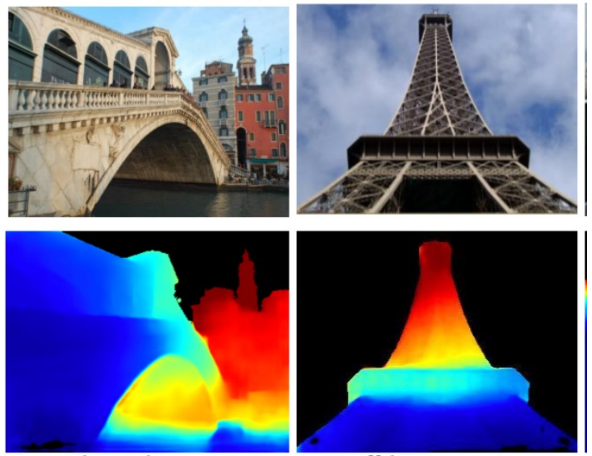
The idea is to feed multiple images with overlapping viewpoint to **Structure from Motion** and **Multi-view Stereo** to automatically produce depth maps.

Multiple filtering steps are necessary.

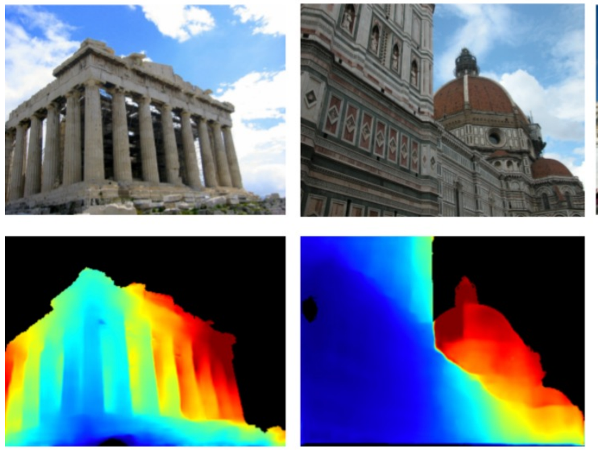
3D data is only up to unknown scale factor, which could be problematic for applications requiring scaled values.

Dataset	Indoor	Outdoor	Dynamic	Video	Dense	Accuracy	Diversity	Annotation	Depth	# Images
DIML Indoor [31]	✓			✓	✓	Medium	Medium	RGB-D	**Metric**	220K
MegaDepth [11]		✓	(✓)		(✓)	Medium	Medium	SfM	No scale	130K
ReDWeb [32]	✓	✓	✓		✓	Medium	**High**	Stereo	No scale & shift	3600
WSVD [33]	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	1.5M
3D Movies	✓	✓	✓	✓	✓	Medium	**High**	Stereo	No scale & shift	75K
DIW [34]	✓	✓	✓			Low	**High**	User clicks	Ordinal pair	496K
ETH3D [35]	✓	✓			✓	**High**	Low	Laser	**Metric**	454
Sintel [36]	✓	✓	✓	✓	✓	**High**	Medium	Synthetic	(Metric)	1064
KITTI [28], [29]		✓	(✓)	✓	(✓)	Medium	Low	Laser/Stereo	**Metric**	93K
NYUDv2 [30]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	407K
TUM-RGBD [37]	✓		(✓)	✓	✓	Medium	Low	RGB-D	**Metric**	80K

Geometric wisdom



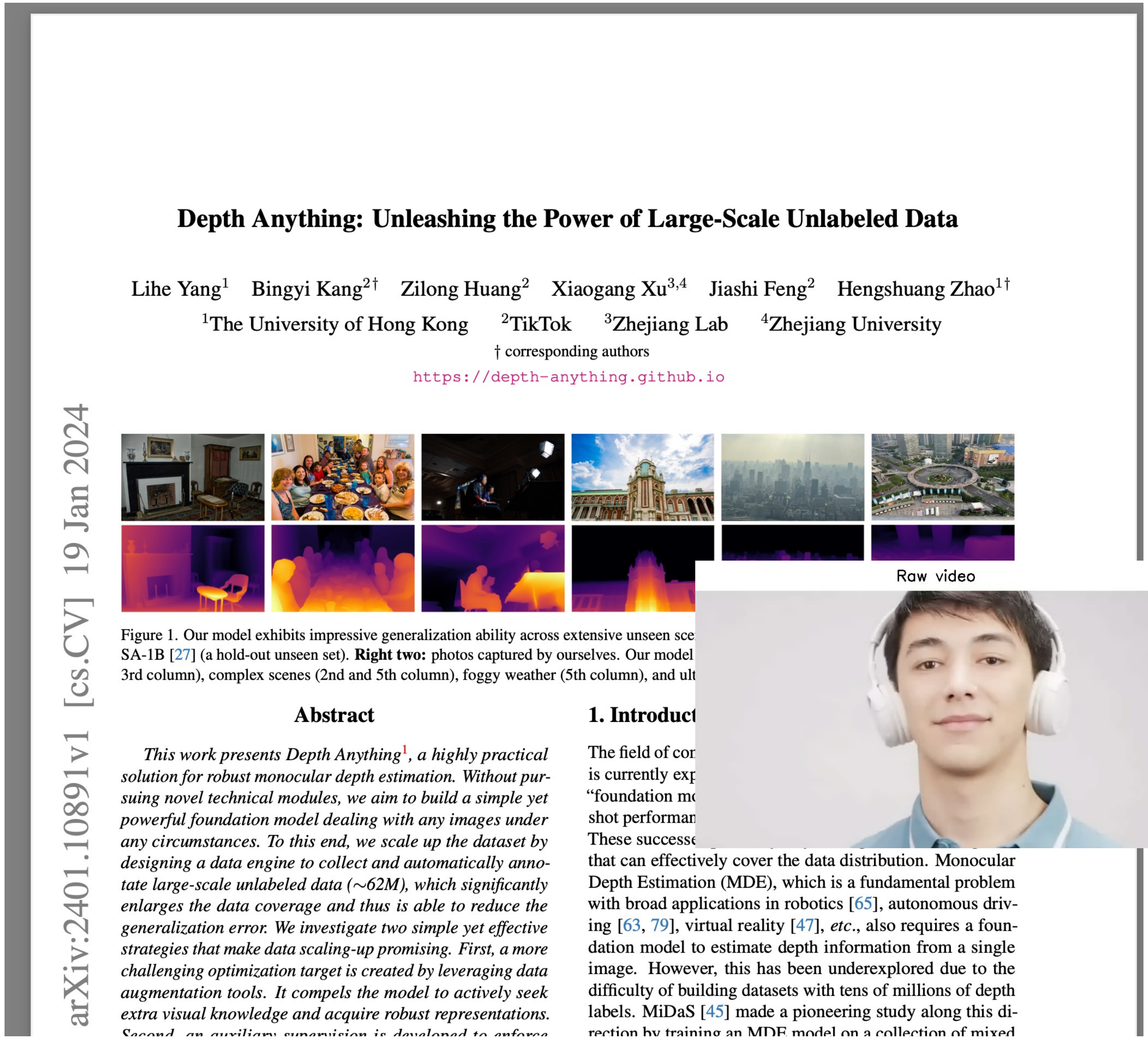
Rialto Bridge, Venice — Eiffel Tower, Paris



Parthenon, Athens — Florence Cathedral, Florence

https://www.cs.cornell.edu/projects/megadepth/

Depth anything



arXiv:2401.10891v1 [cs.CV] 19 Jan 2024

Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data

Lihe Yang[1] Bingyi Kang[2†] Zilong Huang[2] Xiaogang Xu[3,4] Jiashi Feng[2] Hengshuang Zhao[1†]

[1]The University of Hong Kong [2]TikTok [3]Zhejiang Lab [4]Zhejiang University

† corresponding authors

https://depth-anything.github.io

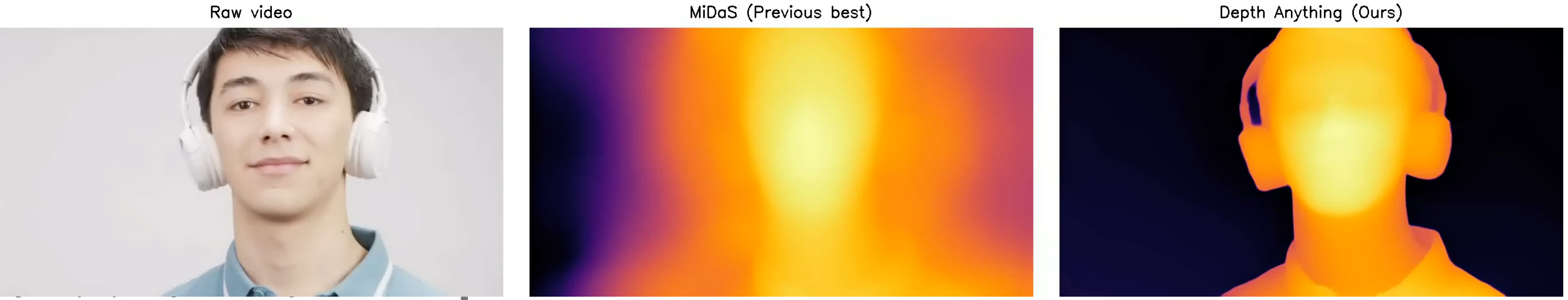
Figure 1. Our model exhibits impressive generalization ability across extensive unseen sce
SA-1B [27] (a hold-out unseen set). **Right two:** photos captured by ourselves. Our model
3rd column), complex scenes (2nd and 5th column), foggy weather (5th column), and ult

Abstract

This work presents Depth Anything[1], a highly practical solution for robust monocular depth estimation. Without pursuing novel technical modules, we aim to build a simple yet powerful foundation model dealing with any images under any circumstances. To this end, we scale up the dataset by designing a data engine to collect and automatically annotate large-scale unlabeled data (~62M), which significantly enlarges the data coverage and thus is able to reduce the generalization error. We investigate two simple yet effective strategies that make data scaling-up promising. First, a more challenging optimization target is created by leveraging data augmentation tools. It compels the model to actively seek extra visual knowledge and acquire robust representations.

1. Introduct

The field of cor
is currently exp
"foundation mo
shot performan
These successe
that can effectively cover the data distribution. Monocular Depth Estimation (MDE), which is a fundamental problem with broad applications in robotics [65], autonomous driving [63, 79], virtual reality [47], *etc.*, also requires a foundation model to estimate depth information from a single image. However, this has been underexplored due to the difficulty of building datasets with tens of millions of depth labels. MiDaS [45] made a pioneering study along this di-



https://depth-anything.github.io

Single image depth estimation

- ✓ Obtaining training data for deep learning models in the wild is nowadays possible
- ✓ Single-image depth estimation models can be effectively trained on such data

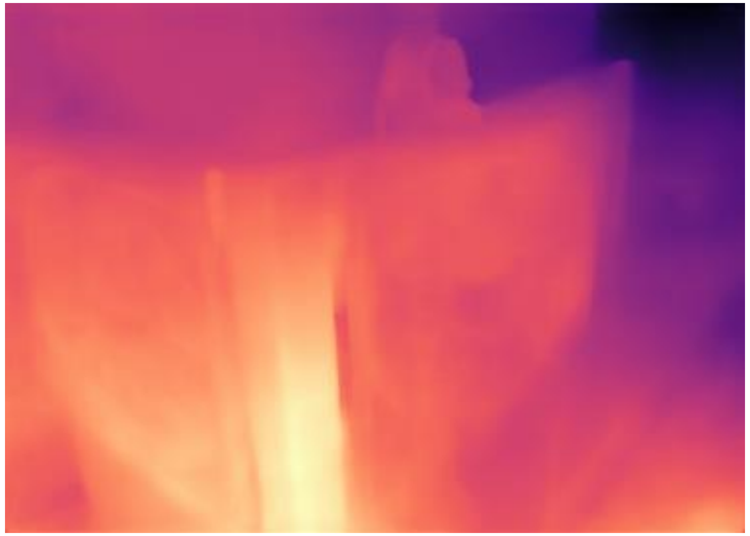
MiDaS Depth Estimation is a machine learning model from Intel Labs for monocular depth estimation. It was trained on up to 12 datasets and covers both in-and outdoor scenes. Multiple different MiDaS models are available, ranging from high quality depth estimation to lightweight models for mobile downstream tasks

Ranftl et al. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer. TPAMI 2020

Single image depth estimation

- ✓ Obtaining training data for deep learning models in the wild is nowadays possible
- ✓ Single-image depth estimation models can be effectively trained on such data
- ✗ It's challenging to collect data that capture the diversity of the visual world to ensure generalization
- ✗ RGB+ depth data are difficult to collect (Kinect is limited to indoor use, LiDAR are expensive and produce only a sparse depth map)
- ✗ These methods can be easily fooled-out by out-of-distribution samples

Picture from
artedelporfido.wordpress.com



Depth map from MegaDepth [1]
megadepthdemo.pythonanywhere.com

Do NN learns from the same visual cues used by humans?

Explaining depth estimation

It is crucial to understand how NNs estimate depths to safely apply them in critical application as autonomous driving…

Understanding single image depth estimation

It is crucial to understand how NNs estimate depths in order to safely apply them in critical application as autonomous driving…

1. *Which are the most **relevant visual cues** in image?*
2. *How **biased** are depth values in presence of specific objects, shadows, camera orientations?*
3. *How **reliable** are depth values?*

Which pixels of an image I are relevant for depth estimation?

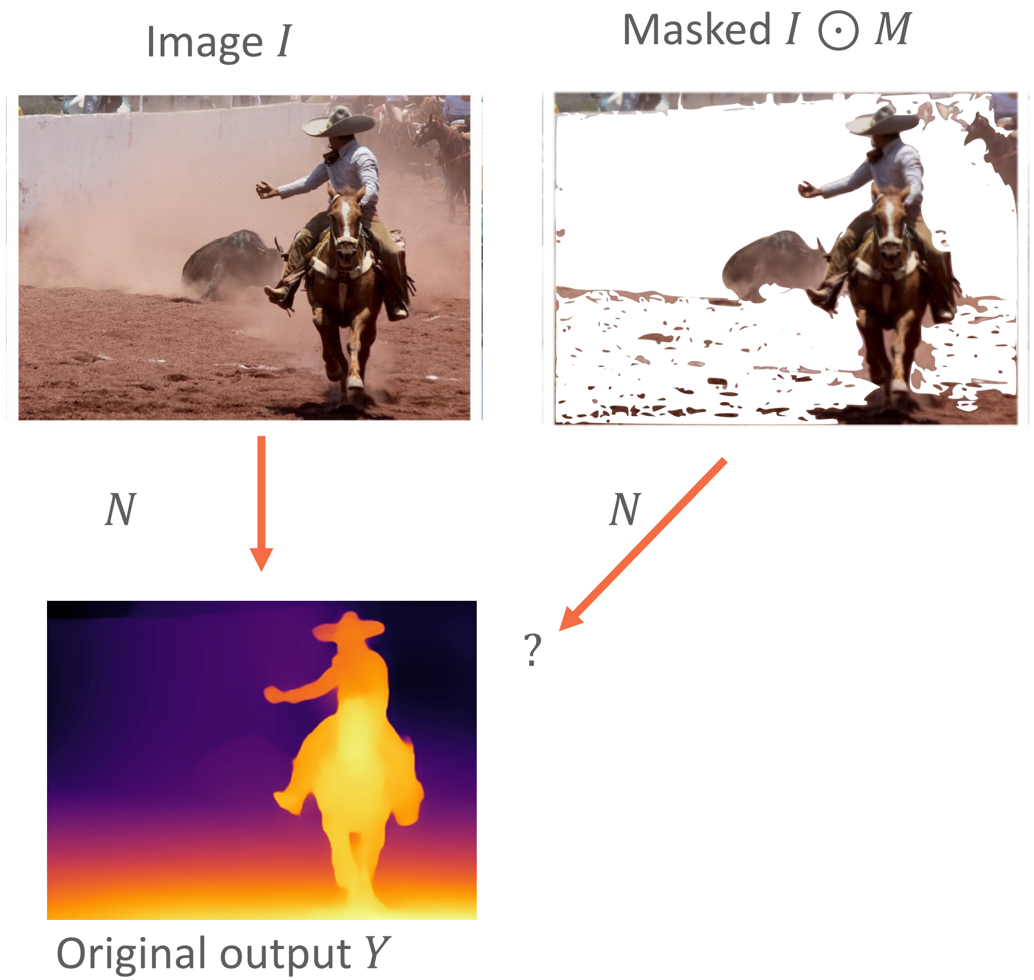
Cast the question as an optimization problem:

select a mask M with the smallest set of pixels from which N, a target CNN, produces the maximally similar depth $\hat{Y} = N(I \odot M)$ to the original output $Y = N(I)$.

in formulas

$$\min \ell(Y, \hat{Y}) + \lambda ||M||_0$$

The idea is that CNNs can infer depth map equally well from a selected set of sparse pixels, as long as they are relevant to depth estimation.



Hu et al., Visualization of Convolutional Neural Networks for Monocular Depth Estimation, ICCV, 2019

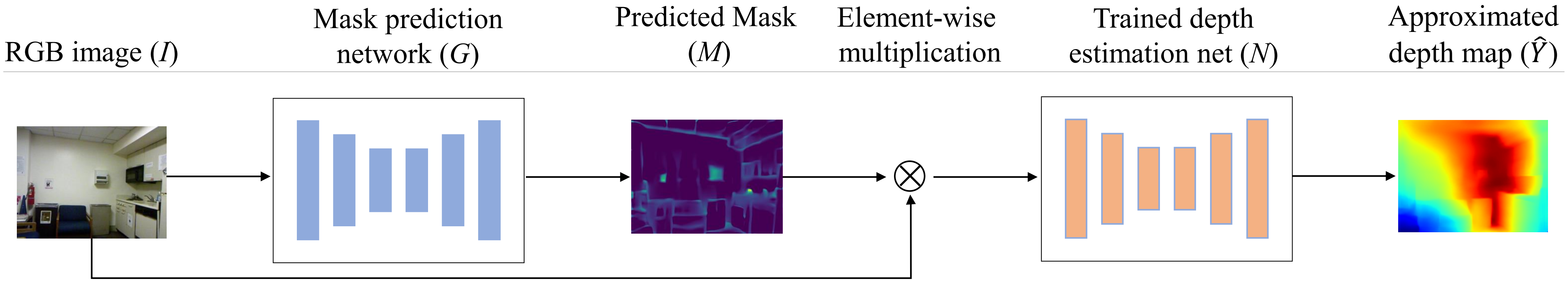
Visualization of CNN for depth estimation

In practice, this problem requires to optimize the output of the CNN with respect to its input that can lead to noisy visualization or even to adversarial examples.
Thus:

- Rather than directly optimizing the elements of M, obtain the mask by processing I via a network G,
- Relax the entries of the matrix to be in $[0,1]$

thus we have the following problem:

$$\min_{G} \ell(D, \hat{D}) + \lambda ||G(I)||_1$$

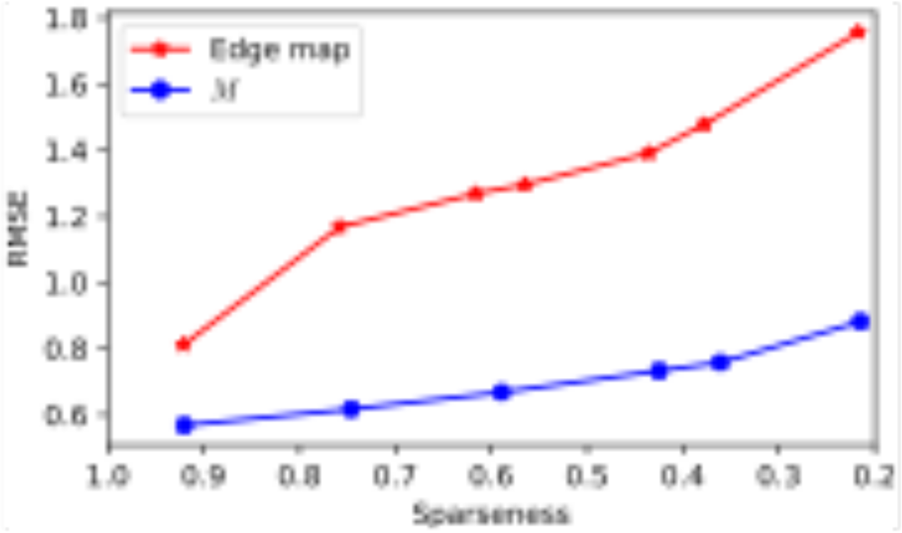


Hu et al., Visualization of Convolutional Neural Networks for Monocular Depth Estimation, ICCV, 2019

Visualization of CNN for depth estimation: results

The network concentrate on edges, but with some differences

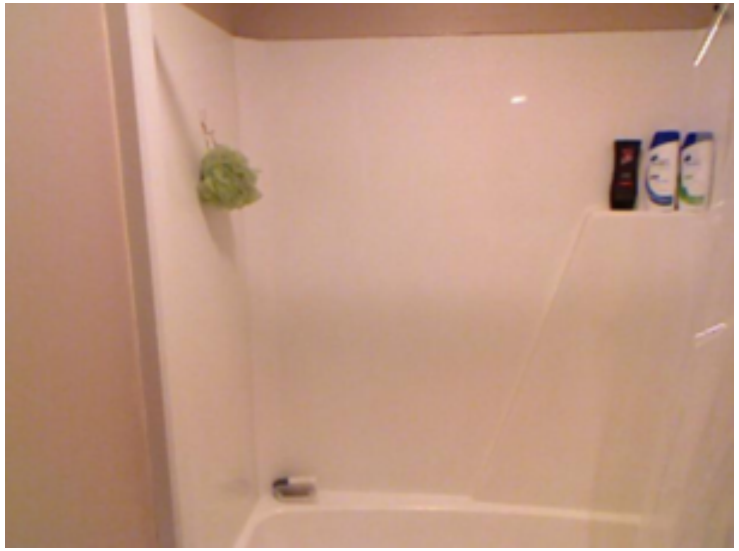
Edge map Mask



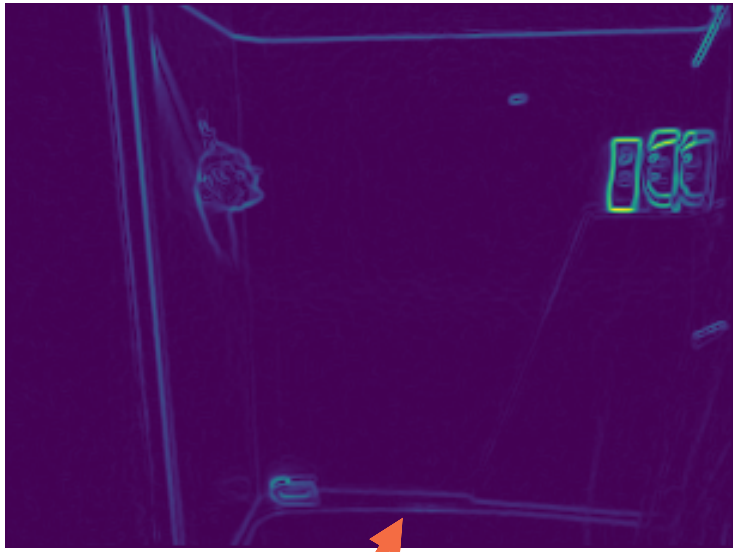
Comparison of accuracy of depth estimation when se- lecting input image pixels using M and using the edge map of input images.

Visualization of CNN for depth estimation: indoor results

The network concentrate on edges, but consider some edge that are important for the understanding the 3D geometry and neglects others

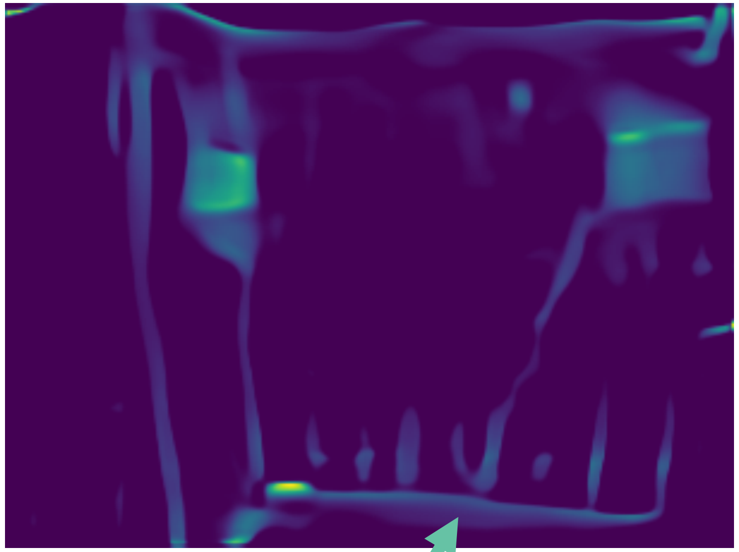


Edge map



Weak edge

Mask

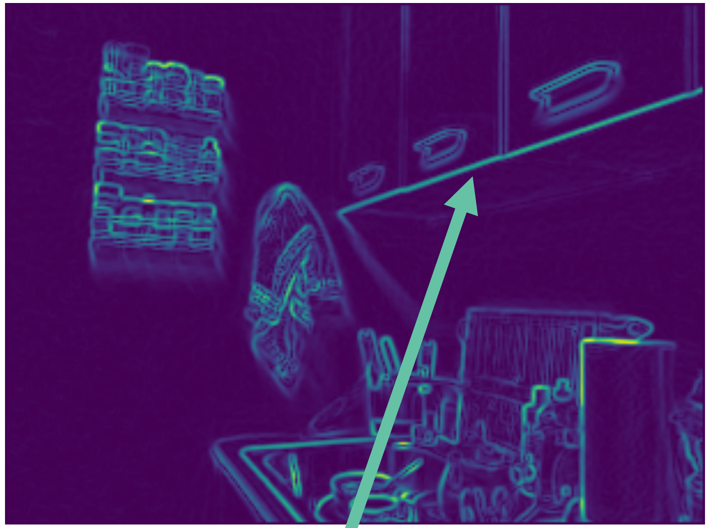


Strong visual cue

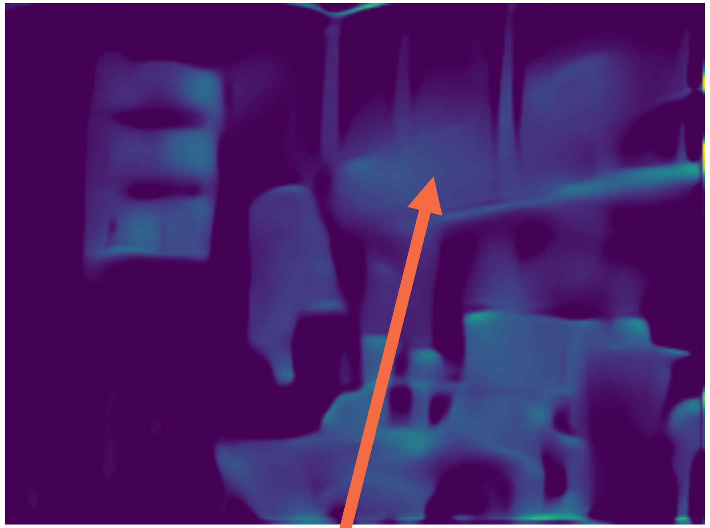
Visualization of CNN for depth estimation: indoor results

The network concentrate on edges, but consider some edge that are important for the understanding the 3D geometry and neglects others

Edge map



Mask



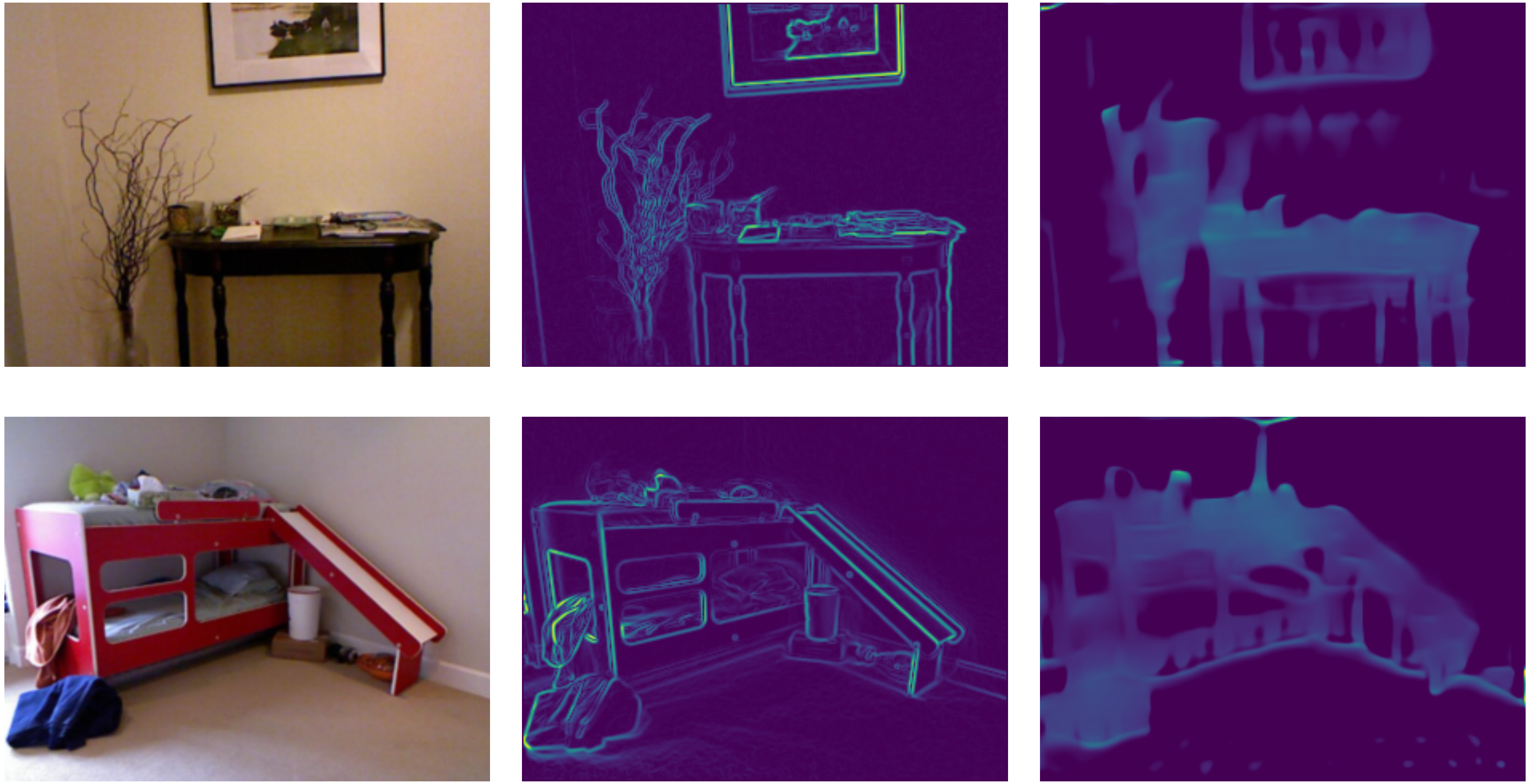
Strong edge

Less evident

Visualization of CNN for depth estimation: indoor results

Not boundary alone but filled region is highlighted for small objects.

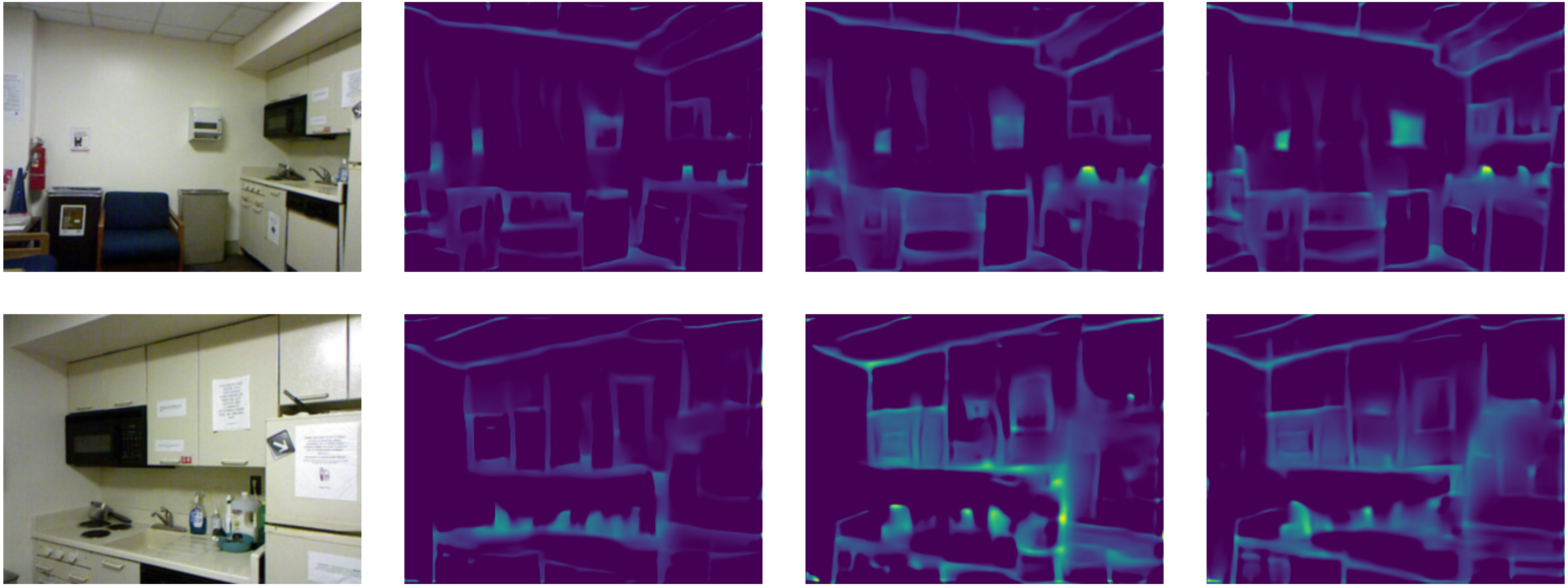
The CNNs recognize the objects and somehow utilize it for depth estimation.



Visualization of CNN for depth estimation: indoor results

By using different losses we get different results

Depth | Depth + gradient | Depth + gradient + normals

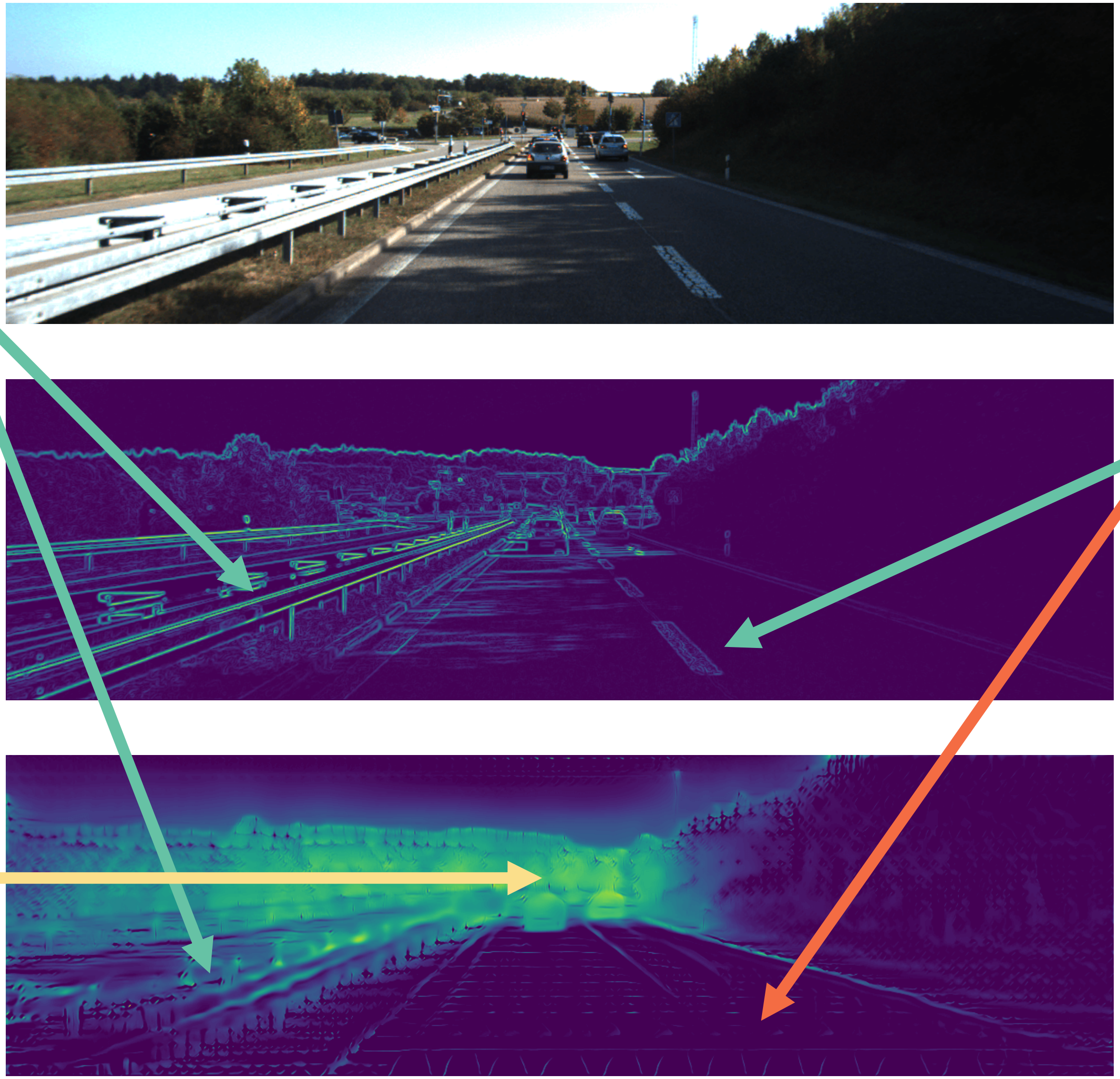


Emphasis on surfaces

Emphasis on objects & straight edges

Visualization of CNN for depth estimation: outdoor results

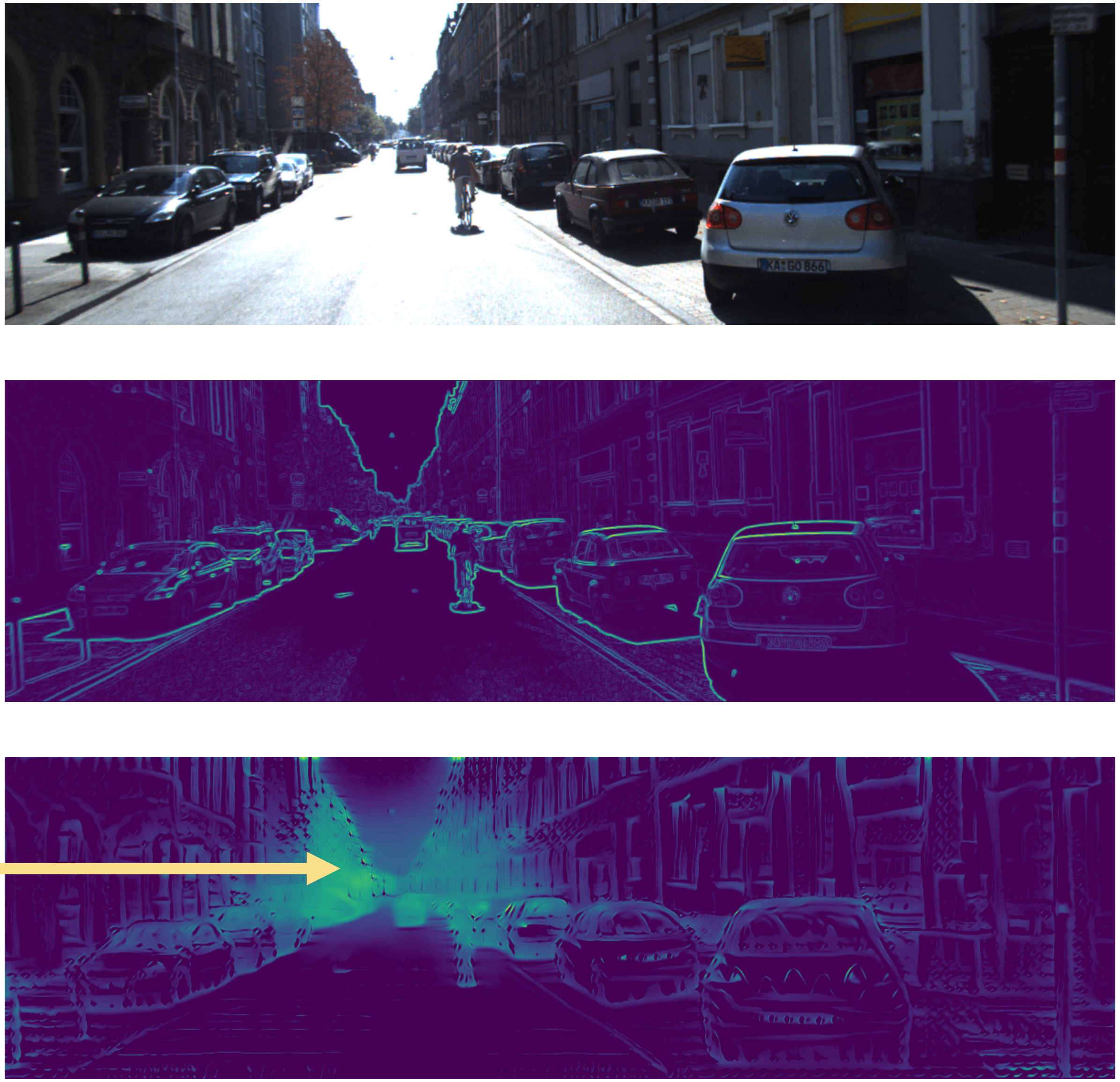
The guard rail is relevant strong edge



The white lines are strong edges but are not relevant

A lot of attention near vanishing ponts

Visualization of CNN for depth estimation: outdoor results



A lot of attention near vanishing ponts

Biases in training set?

Being completely data driven, depth estimation from a single image might inherits the **biases** encoded in the training set.

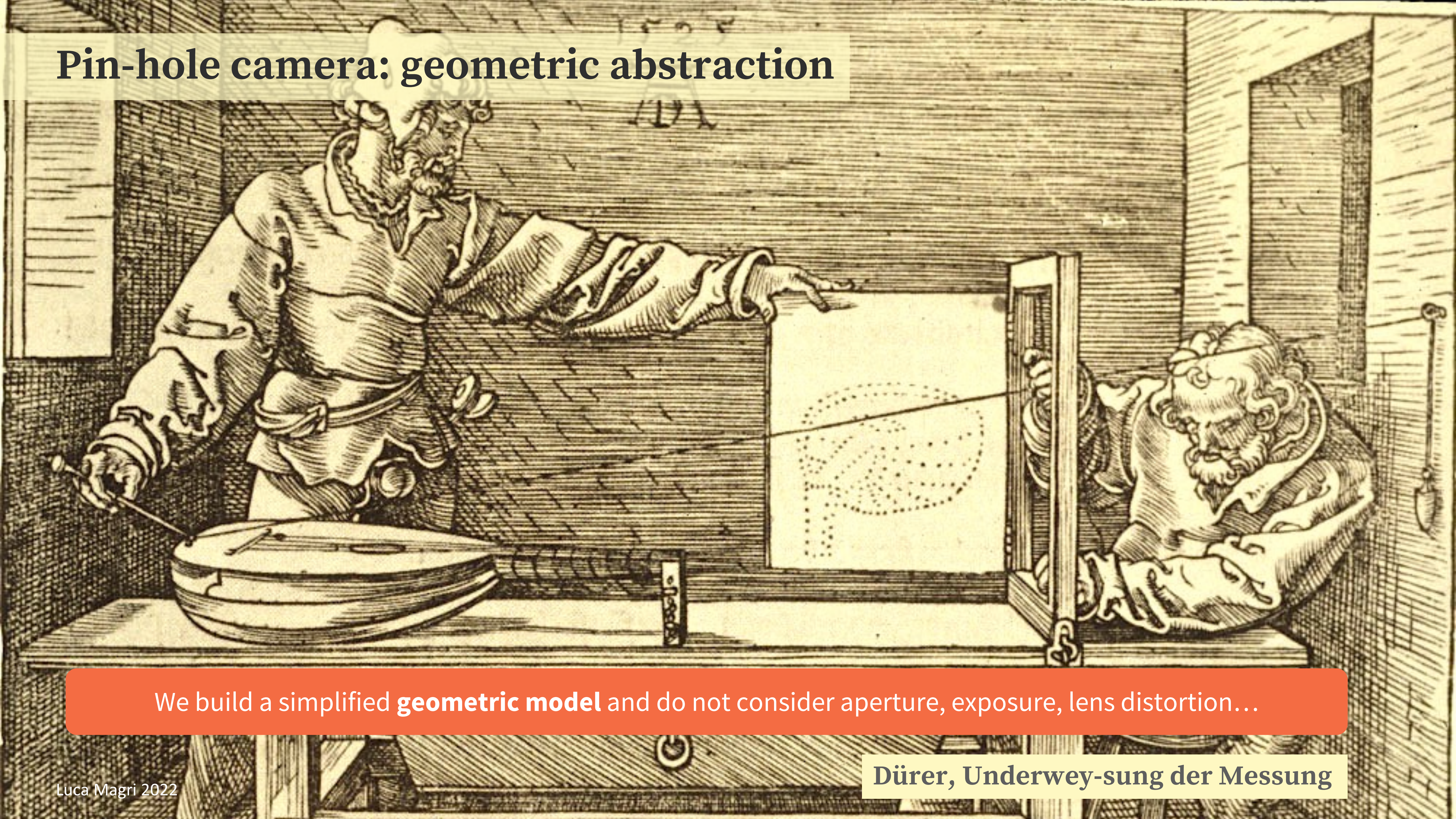
Let's investigate how some cues (e.g., relative position, apparent size…) affect depth estimation.

Understanding this aspect is crucial for the generalization of the model.

Dijk, Tom van, and Guido de Croon. "How do neural networks see depth in single images?." CVPR 2019

Geometric interlude

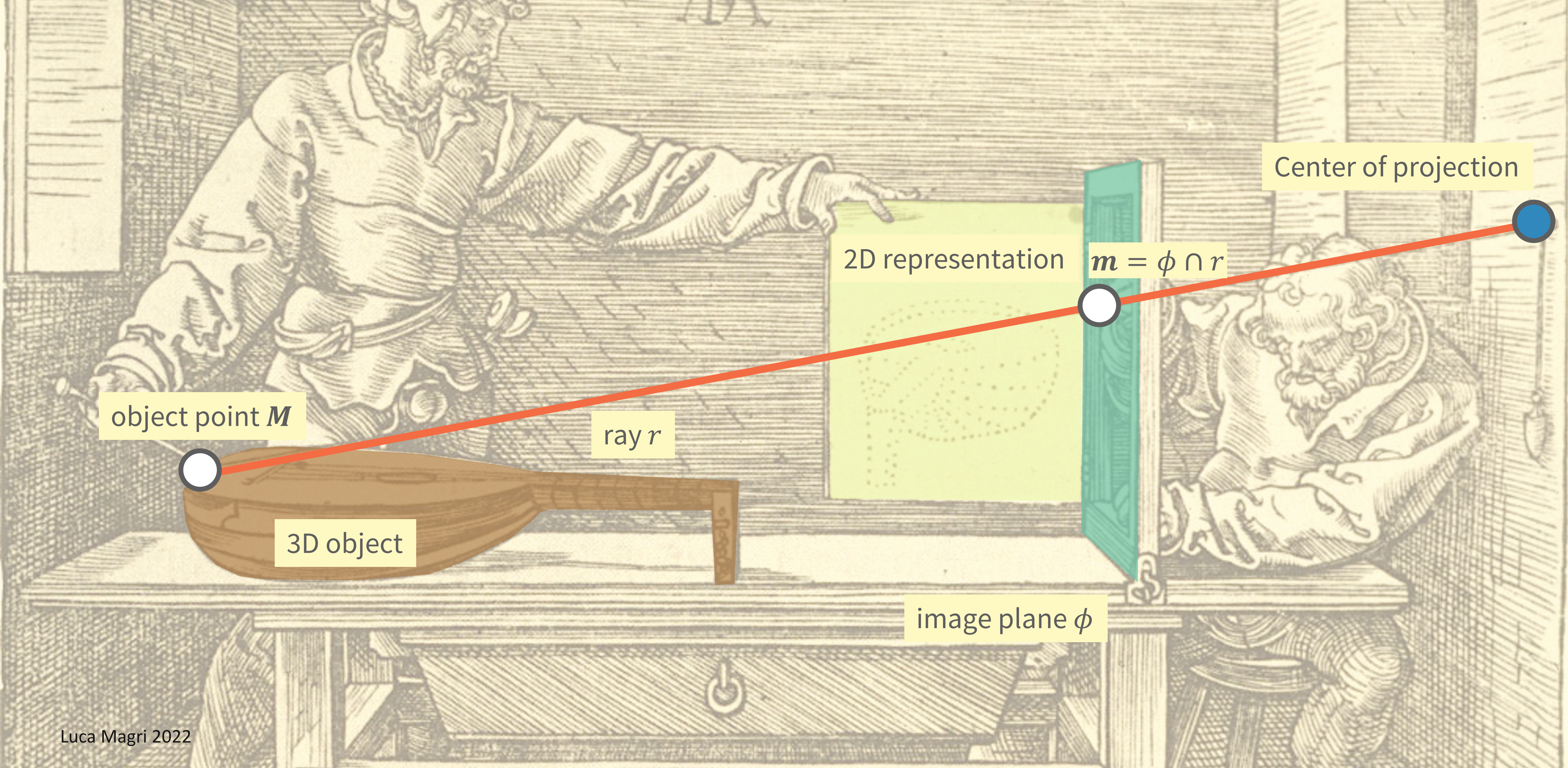
Pin-hole camera: geometric abstraction



We build a simplified **geometric model** and do not consider aperture, exposure, lens distortion…

Dürer, Underwey-sung der Messung

Pin-hole camera: central perspective projection

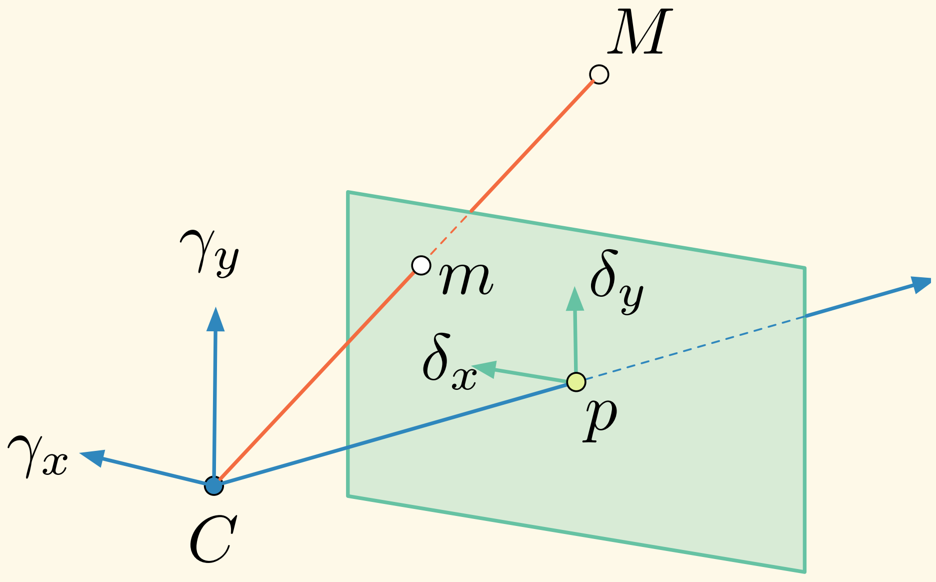


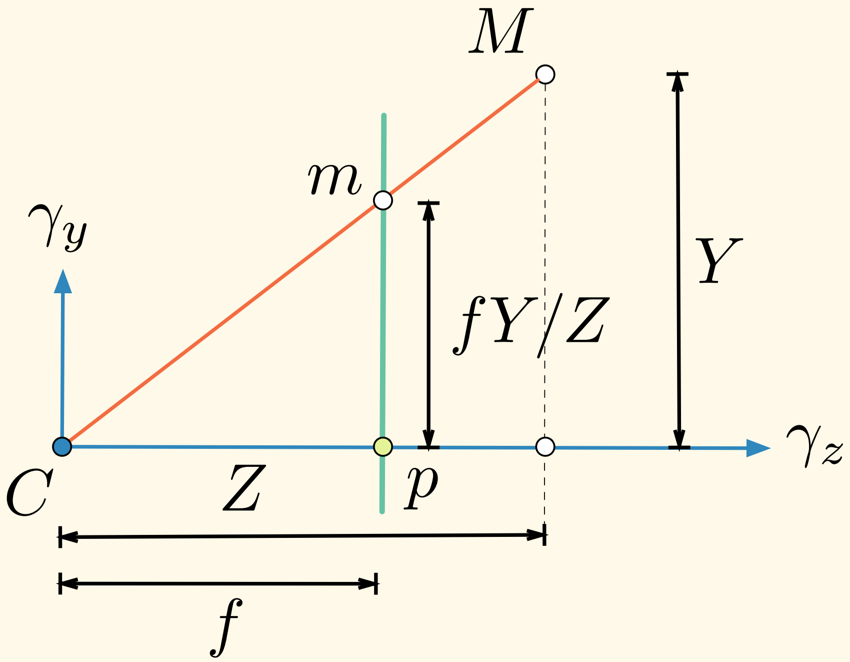
Pin-hole camera geometry

Is described by its optical center C and the image plane ϕ.

The distance of the image plane from C is the f, the focal length.

The relation between M the 3D coordinates of a scene point and m the coordinates of its projection onto the image plane is described by the perspective projection

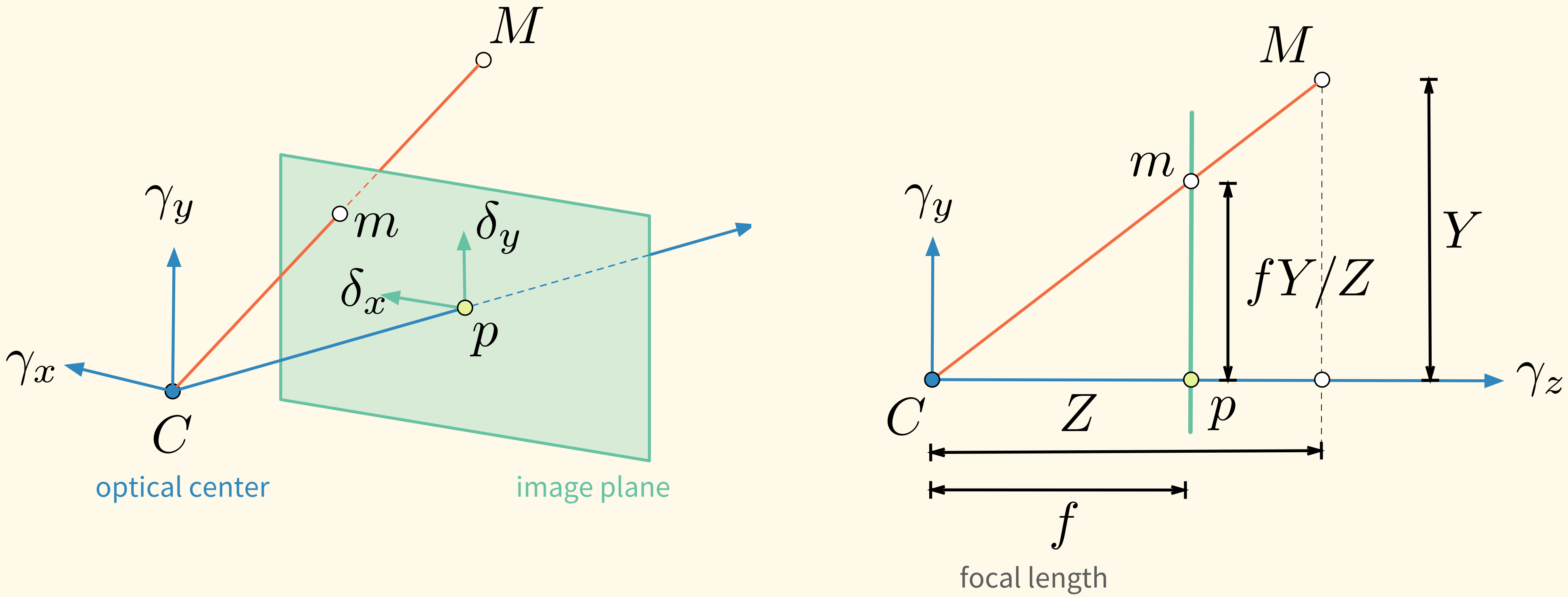




Perspective equations from triangle similarity

Fix a Cartesian coordinate system $\{\gamma_x, \gamma_y, \gamma_z\}$ in the optical center, with γ_z perpendicular to the image plane. By similar triangles, $M = (X_M, Y_M, Z_M)$ is mapped to point $m = (\frac{f X_M}{Z_M}, \frac{f Y_M}{Z_M})$

$$\boldsymbol{M} = (X_M, Y_M, Z_M) \mapsto \boldsymbol{m} = (x_m, y_m), \text{where } \begin{cases} x_m = f\ X_M/Z_M \\ y_m = f\ Y_M/Z_M \end{cases}$$

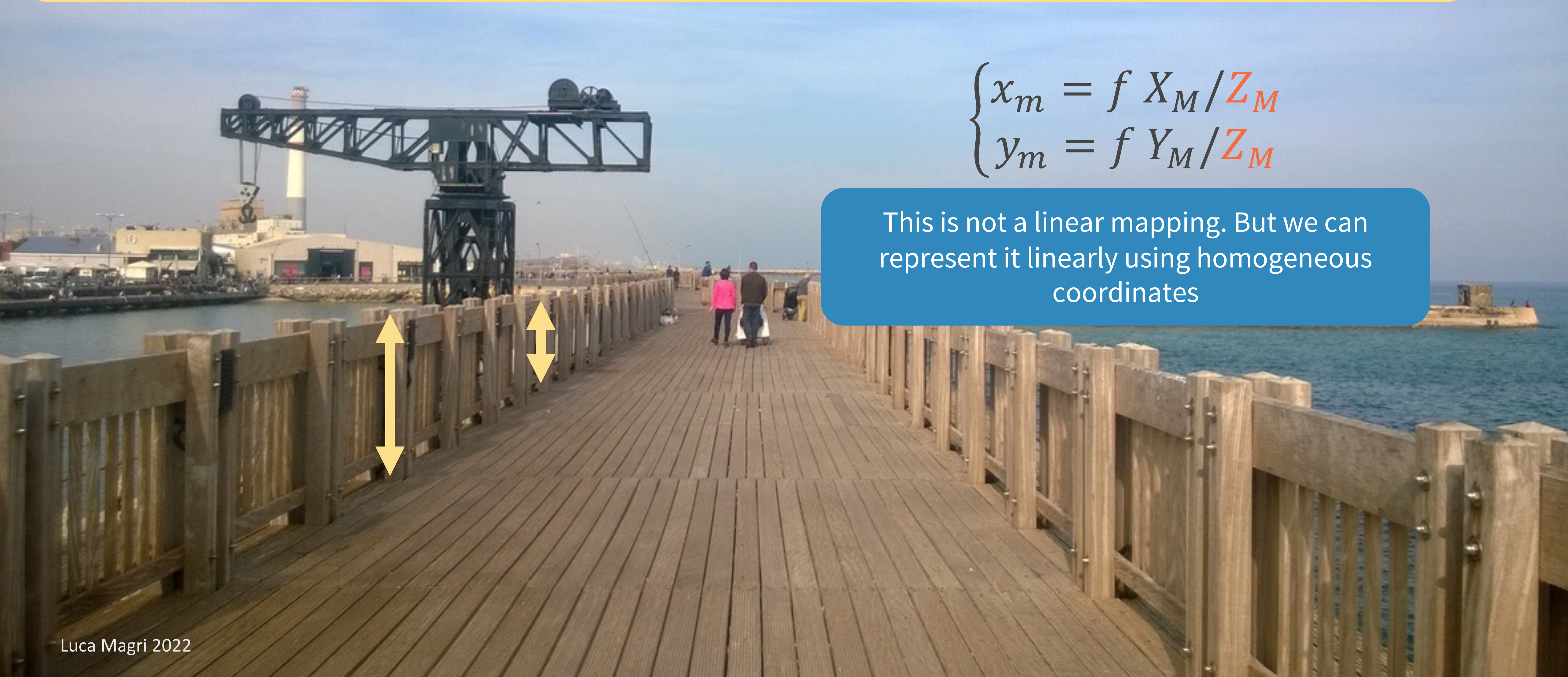


Camera projection equations are non linear

Perspective: division by Z_M is responsible of perspective effects. The size of images in the image plane actually depends on their depth in the scene (*i.e.*, distance from the camera center)

$$\begin{cases} x_m = f\ X_M / Z_M \\ y_m = f\ Y_M / Z_M \end{cases}$$

This is not a linear mapping. But we can represent it linearly using homogeneous coordinates



Vanishing points

Parallel lines in 3D space appear to converge towards the horizon.
The line of horizon is formed by infinitely distant points or “vanishing direction”.

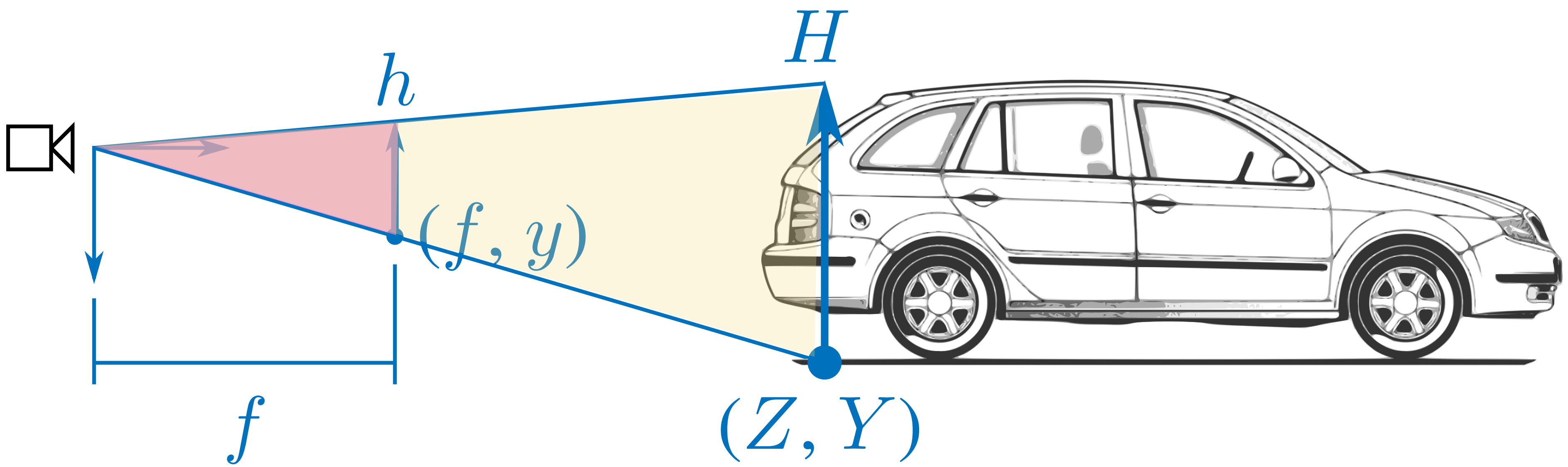
Apparent size

The **apparent size** of objects are strong visual cues that can be used by a network.

If the object size H is known, given the apparent size h and the focal distance, it is possible to compute the depth as

$$Z = \frac{f}{h} H$$

Most of the objects in KITTI are from a limited number of classes (e.g. cars, lorries, pedestrians) having approximately the same size. The networks can learn to recognize objects and use their apparent size to estimate their distance.

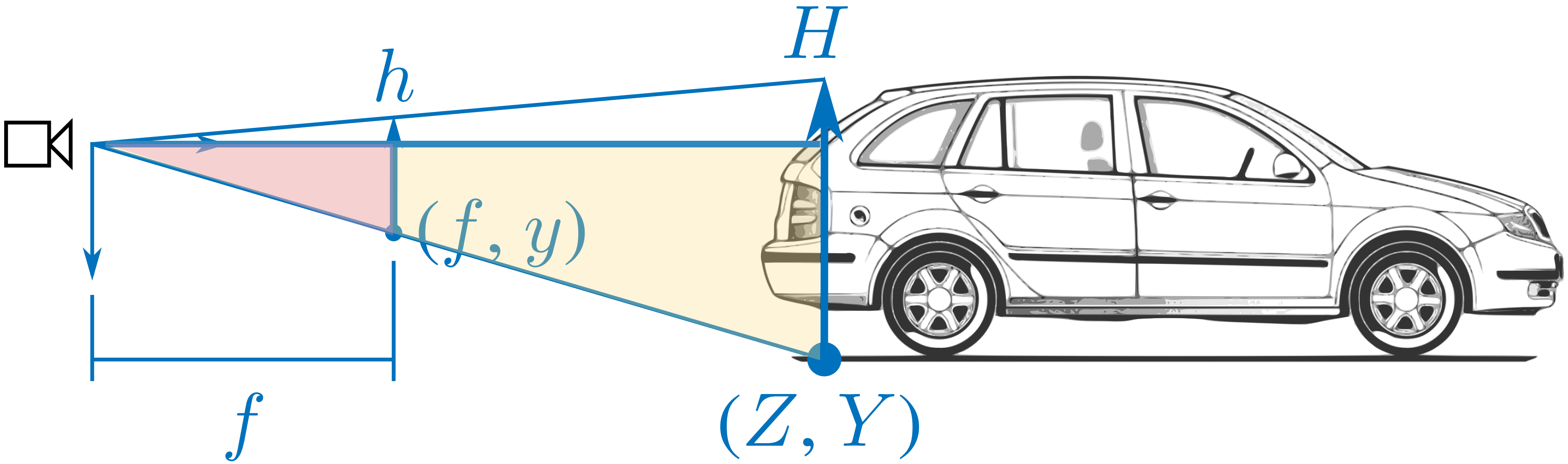


Vertical position (in terms of horizon)

Also, the vertical position is an important cue.

If the camera position is known and assuming a flat plane (as in KITTI), the distance can be computed in terms of the height of horizon y_h as:

$$Z = \frac{f}{(y - y_h)} Y$$

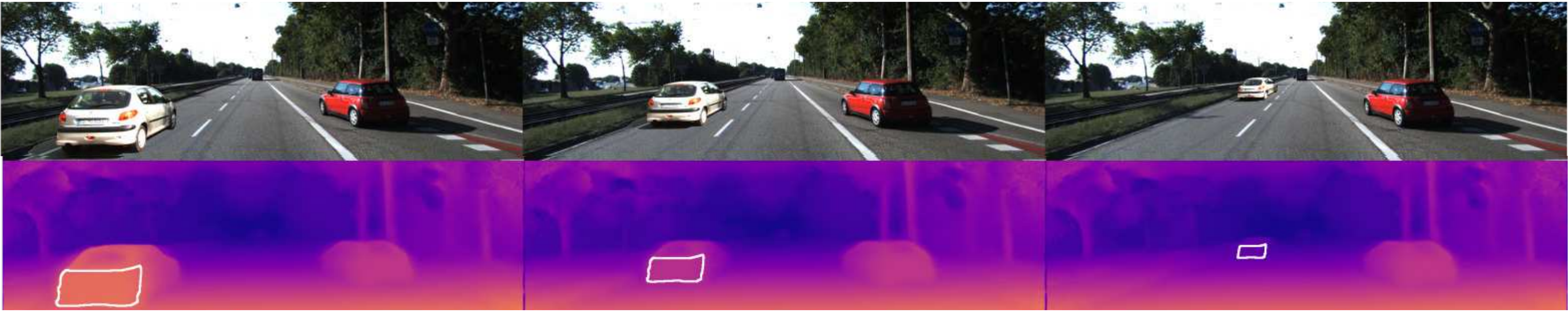


Position vs apparent size

Cropped cars are overimposed with ground contact point at (x, y) and relative depth Z.

When moving to Z', scale factor s and new ground contact point (x', y') can be obtained by knowing the horizon height y_h.

We can modify the position and the apparent size of the white car in a principled manner.

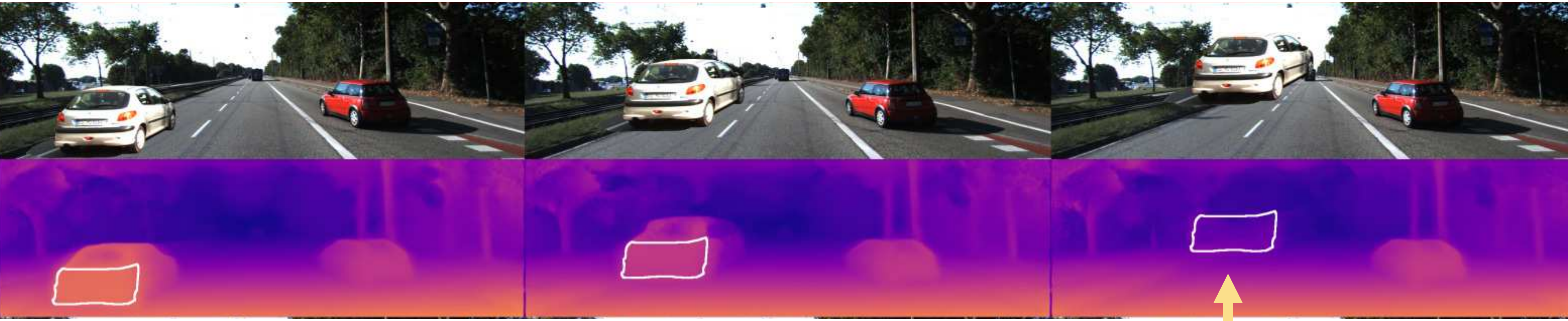


What happen when we use an apparent size that does not conform with the position and viceversa?

Position vs apparent size

Apparent size is fixed, but position changes. Can you guess the result?

Position vs apparent size



The wrong apparent size doesn't have a great impact on the depth estimates.

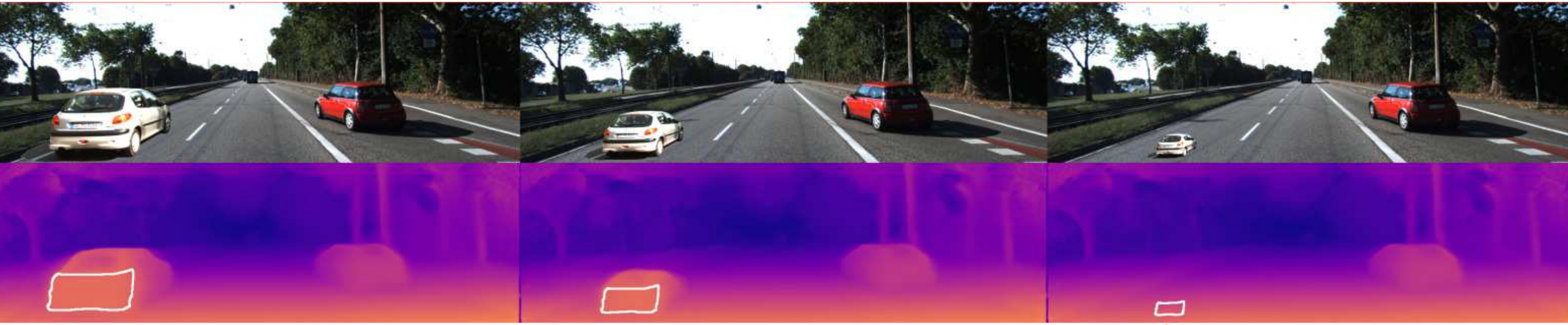
According to the apparent size alone this should be a close object, but it should be far according to the position. Is predicted as far

Position vs apparent size

Position is fixed, but apparent size changes… can you guess the result?

Position vs apparent size

Position is fixed, but apparent size changes… can you guess the result?



According to the apparent size alone this should be a far object, but it should be close according to the position. Is predicted as close

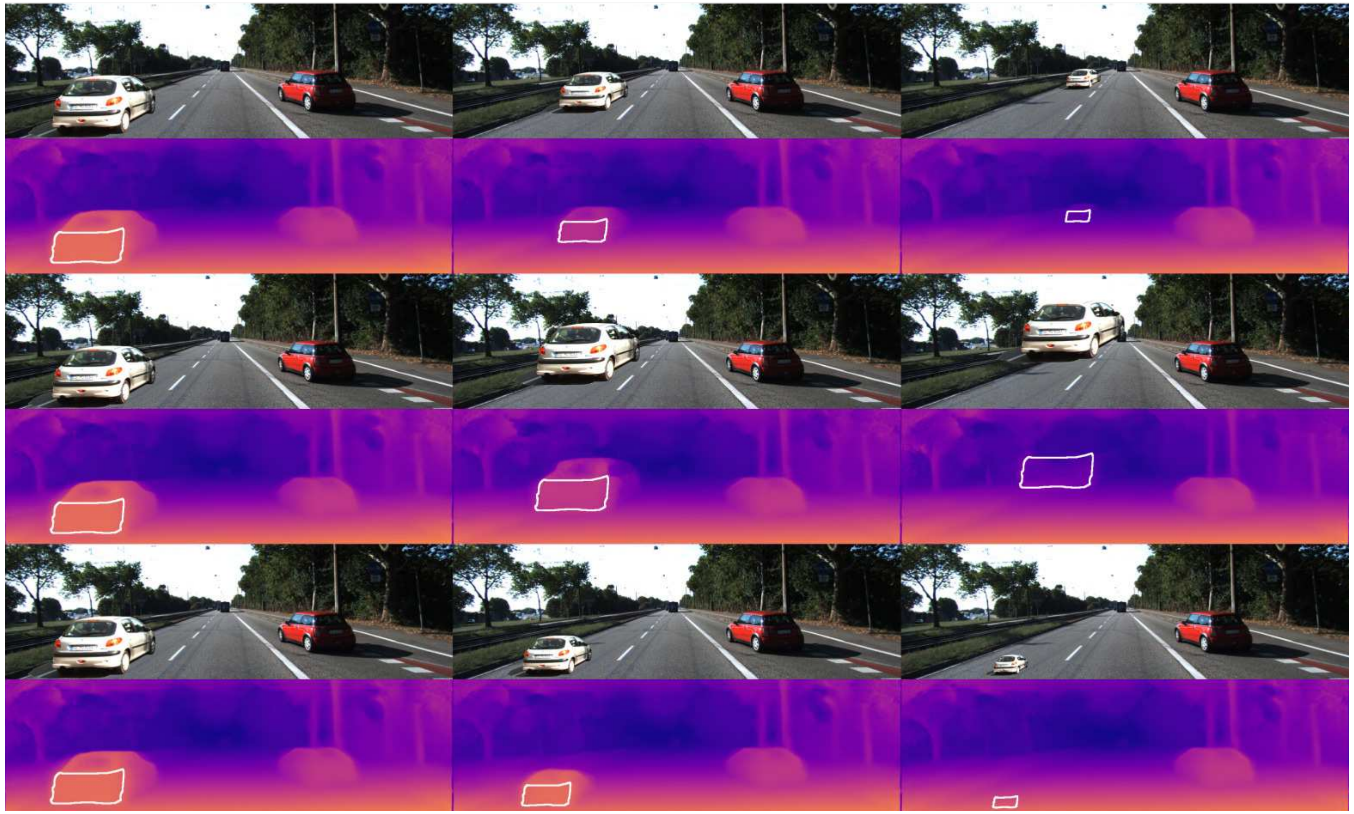
Position is a stronger visual cue!

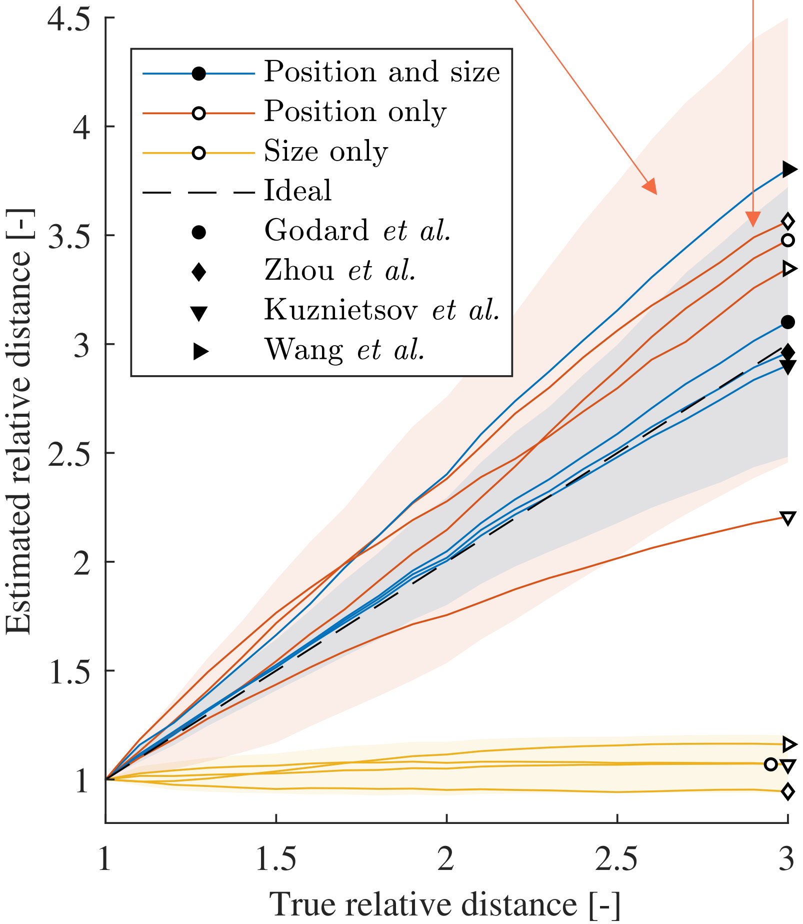
- Overestimate relative depth
- std increased

Position & size

Position only

Size only





ok

ko!

Note that the use of vertical position as a depth cue implies that the networks have some knowledge of the camera's **pose**…

Geometric interlude

Camera and poses

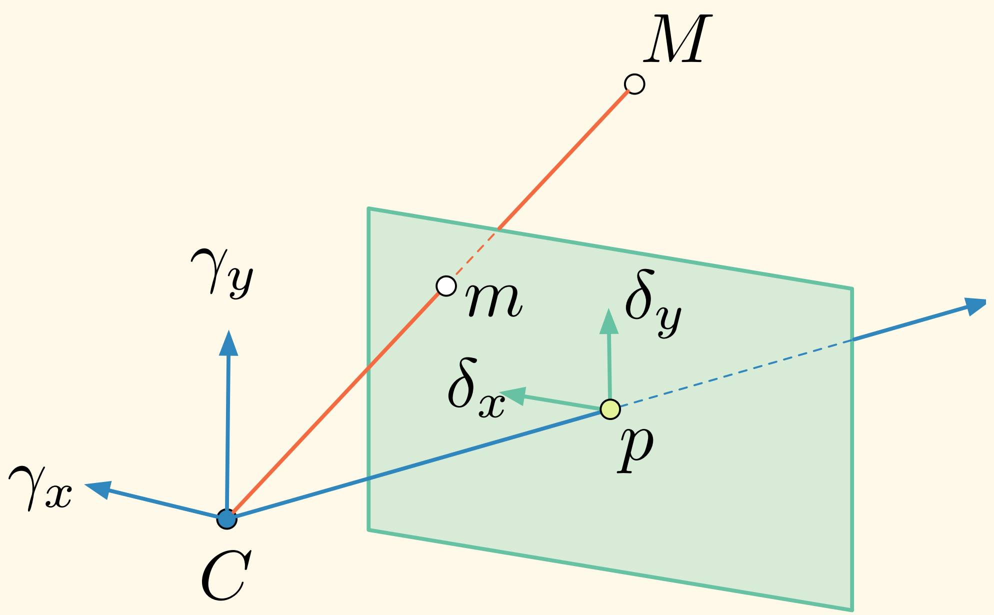
General camera

$zm = PM$

We have chosen:

- a 3D reference frame in the camera center;
- a 2D reference frame in the center of the image;

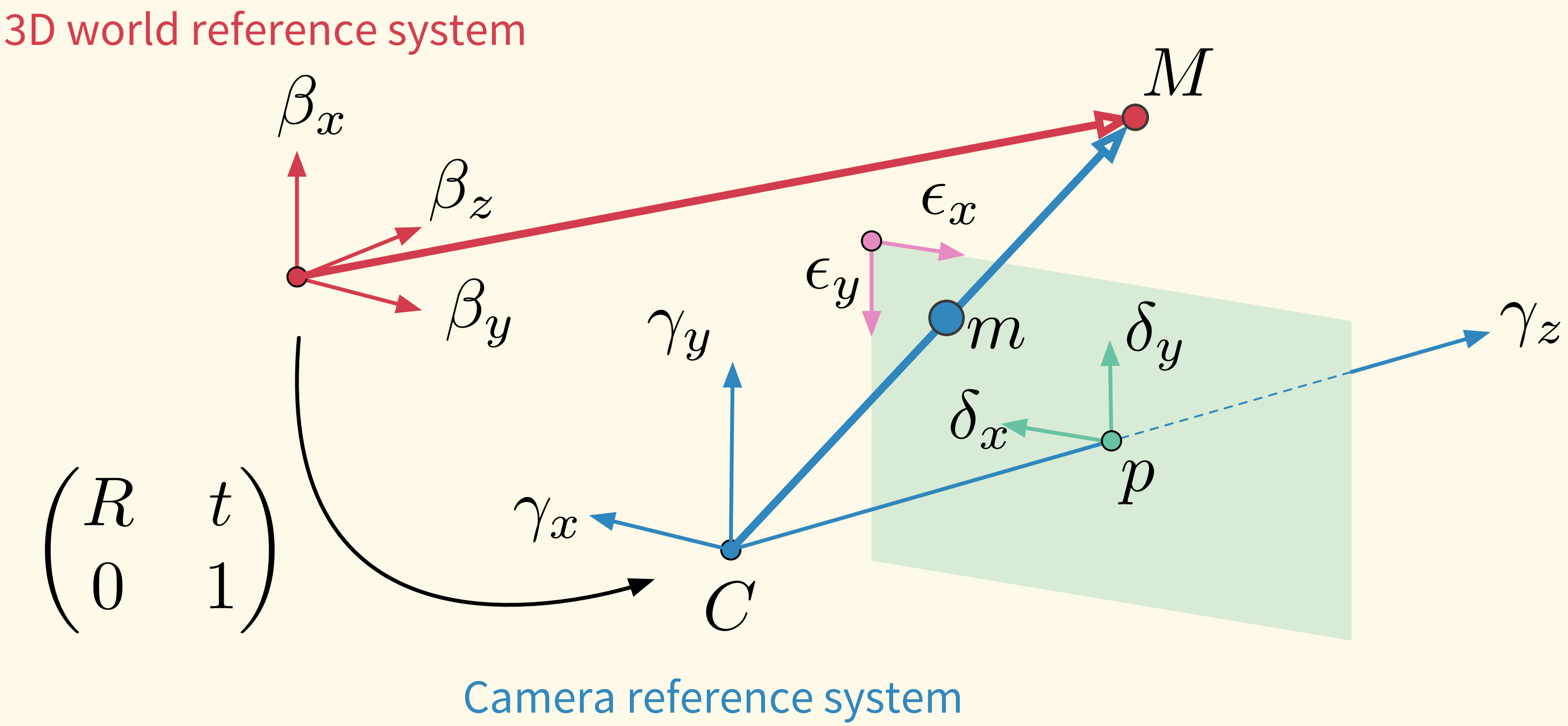
The projection matrix can be generalized to account for other choices of the reference systems.



General camera

1. From world reference frame to camera reference frame using a roto-translation

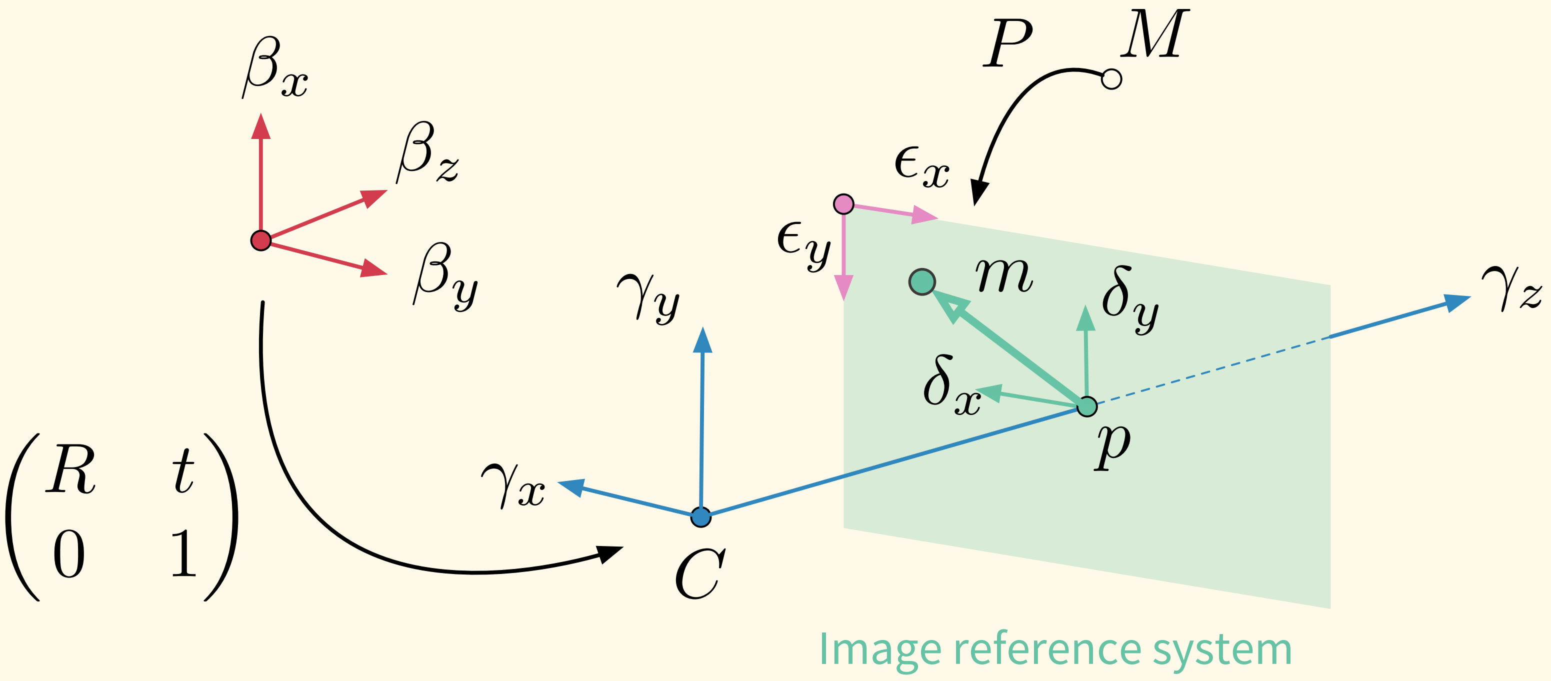
$$\boldsymbol{M} \mapsto R\boldsymbol{M} + t$$



General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix

$$\boldsymbol{M} \mapsto R\boldsymbol{M} + \boldsymbol{t} \mapsto \hat{P}(R\boldsymbol{M} + \boldsymbol{t})$$

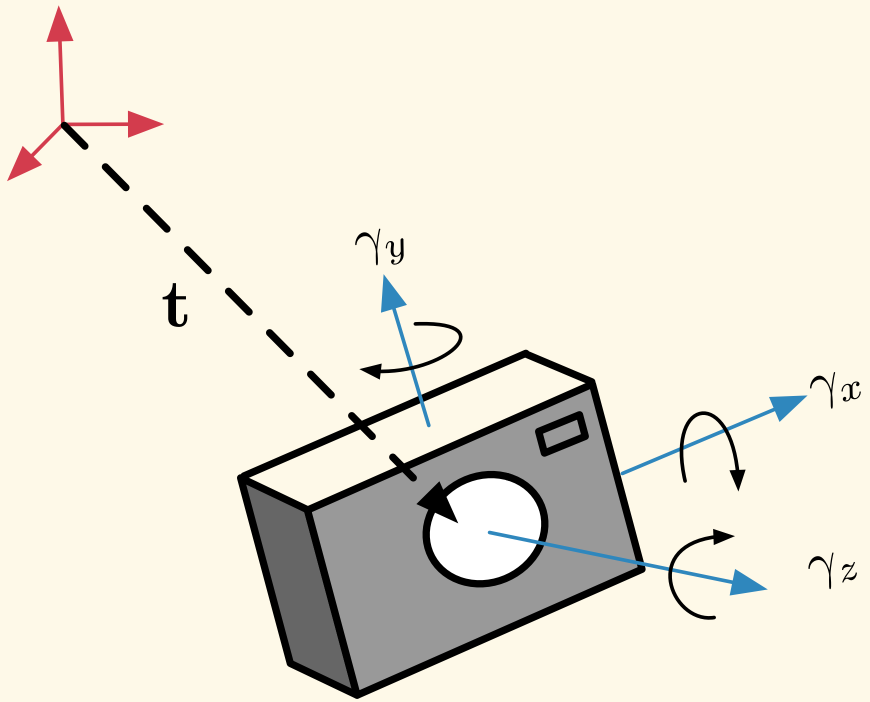


General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix

word r.f. camera r.f. image r.f.

$$\boldsymbol{M} \mapsto G\boldsymbol{M} \mapsto \hat{P}G\boldsymbol{M}$$



External orientations:
Changing coordinates in space is equivalent to multiplying the matrix P to the right by a 4×4 matrix

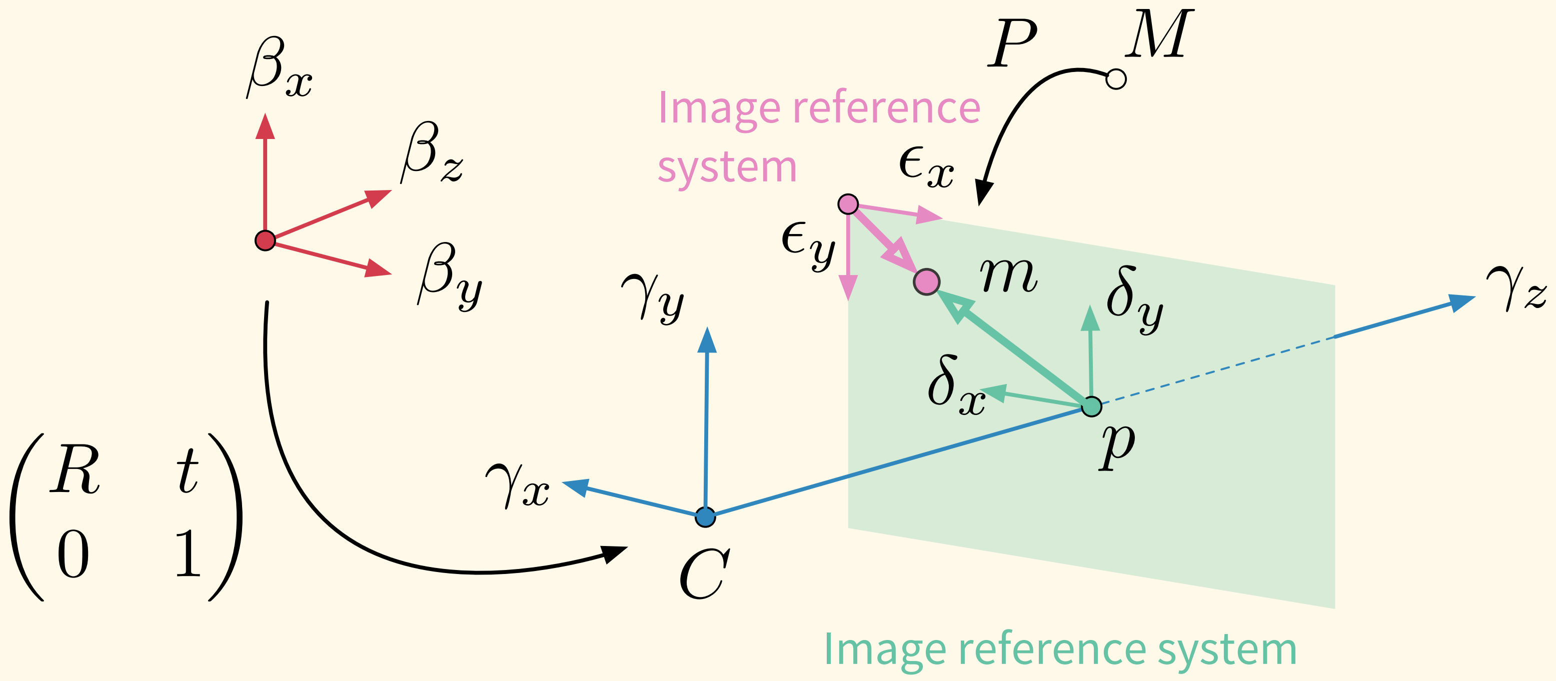
$$G = \begin{bmatrix} R & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$$

composed by a rotation matrix R and a translation vector $\boldsymbol{t}$.
It describes the position and the attitude of the camera with respect to the external reference system.
It depends on six parameters called external orientations.

General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix
3. Express the image point in a different image reference system

$$\boldsymbol{M} \mapsto \boldsymbol{GM} \mapsto \hat{P}\boldsymbol{GM} \mapsto K\,\hat{P}\boldsymbol{GM}$$



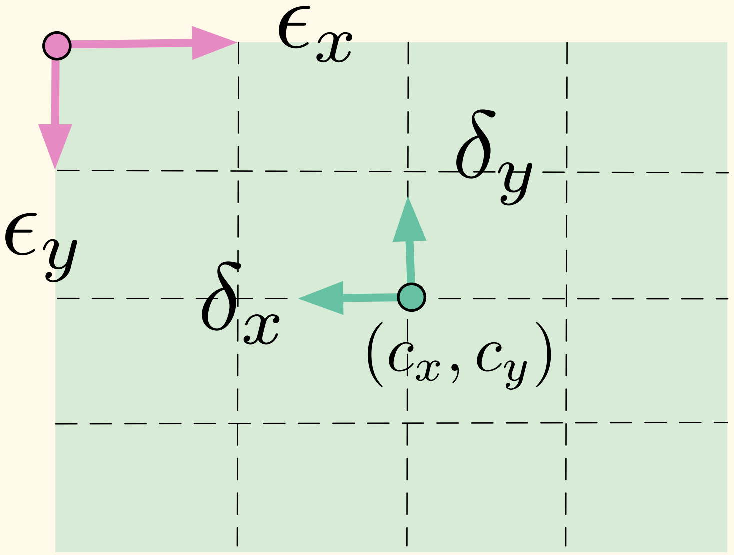
General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix
3. Express the image point in a different image reference system

$$\boldsymbol{M} \mapsto \boldsymbol{GM} \mapsto \hat{P}\boldsymbol{GM} \mapsto K\,\hat{P}\boldsymbol{GM}$$

2D points in the image plane and 2D point in image coordinates differ by an offset and are expressed in pixels and may have an aspect ratio $\neq 1$. These can be accommodated in the camera projection equations

$$\begin{cases} x_m = \sigma_x \dfrac{X_M}{Z_M} + c_x \\ y_m = \sigma_y \dfrac{Y_M}{Z_M} + c_y \end{cases}$$



General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix
3. Express the image point in a different image reference system

$$\boldsymbol{M} \mapsto \boldsymbol{GM} \mapsto \hat{P}\boldsymbol{GM} \mapsto K\hat{P}\boldsymbol{GM}$$

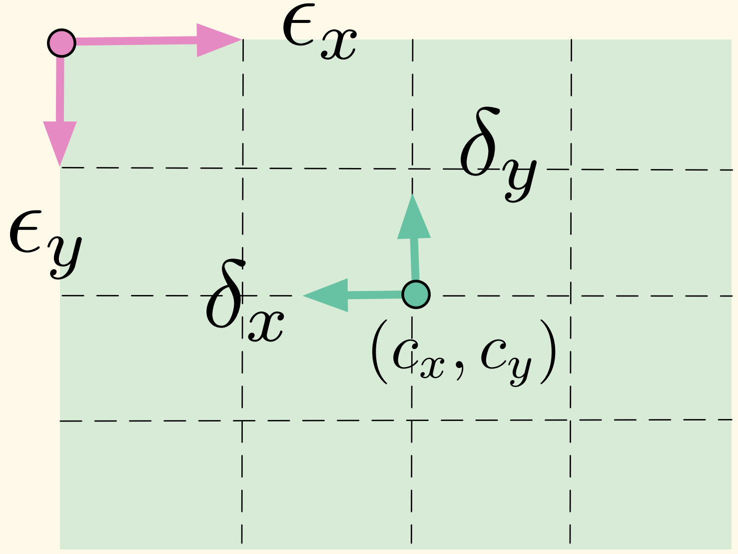
Camera calibration matrix:

In matrix form, this is equivalent of multiplying the matrix P to the left by a 3×3 matrix K representing an affine transform. It is customary to include also the focal length (providing a uniform scaling)

$$K = \begin{bmatrix} \alpha_u & s\alpha_u & c_x \\ 0 & r\alpha\_u & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

It depends on the *interior parameter*:

- Focal length α_u expressed in pixel units
- Principal point (c_x, c_y) (image center)
- Aspect ratio r (typical value 1)
- Skew s (typical value 0)



General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix
3. Express the image point in a different image reference system

$$\boldsymbol{M} \mapsto \boldsymbol{G}\boldsymbol{M} \mapsto \hat{P}\boldsymbol{G}\boldsymbol{M} \mapsto K\hat{P}\boldsymbol{G}\boldsymbol{M}$$

Camera calibration matrix:
In matrix form, this is equivalent of multiplying the matrix P to the left by a 3×3 matrix K representing an affine transform. It is customary to include also the focal length (providing a uniform scaling)

$$K = \begin{bmatrix} \alpha_u & s\alpha_u & c_x \\ 0 & r\alpha\_u & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

It depends on the *interior parameter*:

- Focal length α_u expressed in pixel units
- Principal point (c_x, c_y) (image center)
- Aspect ratio r (typical value 1)
- Skew s (typical value 0)

Image cords $\tilde{\boldsymbol{x}}$

- Accessible
- Are measured in the digital image in pixels

Normalized image cords (NIC) $\tilde{\boldsymbol{p}} = \boldsymbol{K}^{-1}\tilde{\boldsymbol{x}}$

- Not accessible without the knowledge of K
- Normalized image coordinates would be measured on an ideal image plane at unit distance from the camera center. Their unit is the same of 3D points.

General camera

1. From world reference frame to camera reference frame using a roto-translation
2. Project from camera reference frame to image plane using the projection matrix
3. Express the image point in a different image reference system

$$\boldsymbol{M} \mapsto G\boldsymbol{M} \mapsto \hat{P}G\boldsymbol{M} \mapsto K\hat{P}G\boldsymbol{M}$$

$$P = K[I|\mathbf{0}]G = K[R|\boldsymbol{t}]$$

$$\zeta\, \tilde{x} = P\, \tilde{X}$$

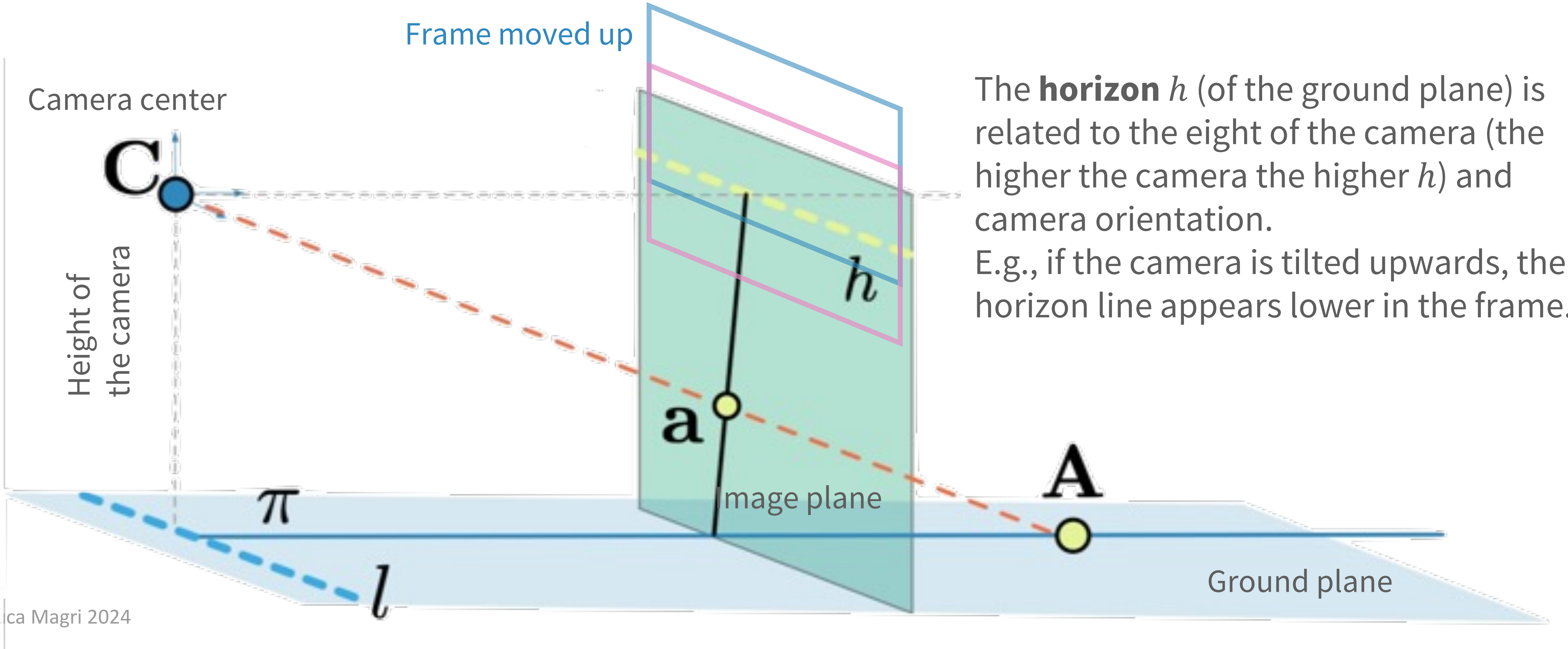
Remarks:

- P has rank 3 since is a 3×4 matrix.
- KR is non singular, since K is upper triangular with nonzero diagonal and R is a rotation matrix
- The Right Null Space of the projection matrix is the camera center (the point for which the projection is not defined)

Camera pose: pitch and roll

Does the NN assume a fixed camera pose or estimate this on the fly?

This is strictly related to the location of the **horizon** and of the **vanishing points**

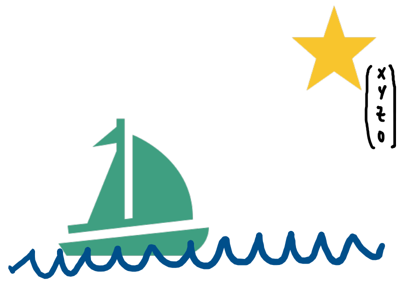


The **horizon** h (of the ground plane) is related to the eight of the camera (the higher the camera the higher h) and camera orientation.
E.g., if the camera is tilted upwards, the horizon line appears lower in the frame.

Camera pose: pitch and roll

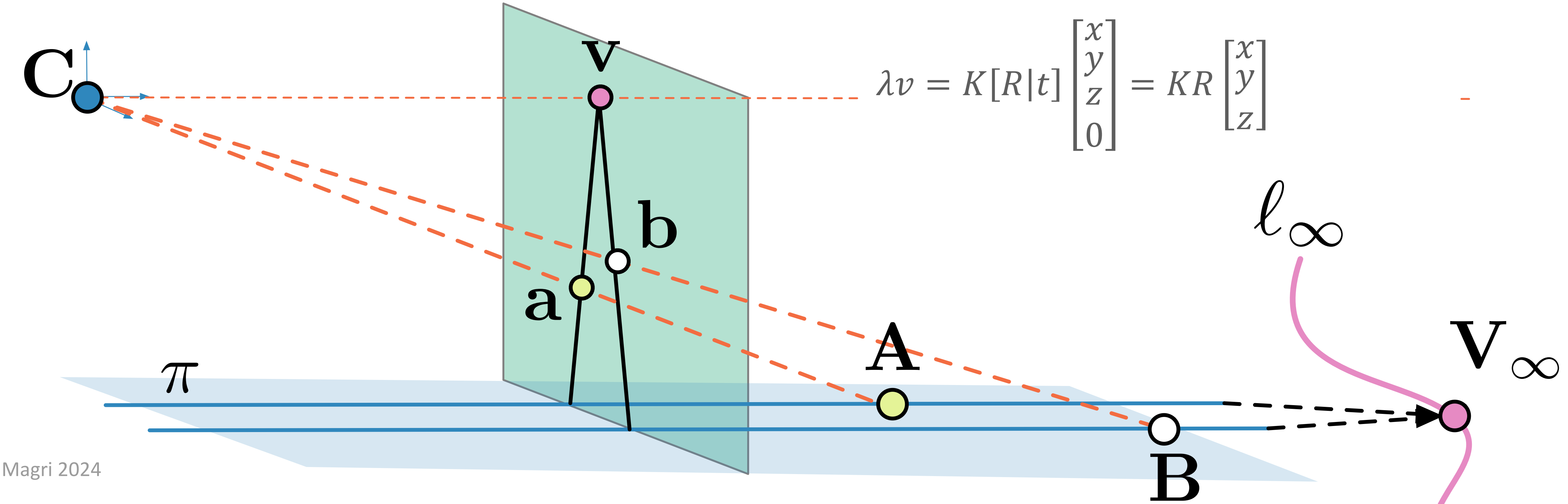
Does the NN assume a fixed camera pose or estimate this on the fly?

This is strictly related to the location of the **horizon** and of the **vanishing points**



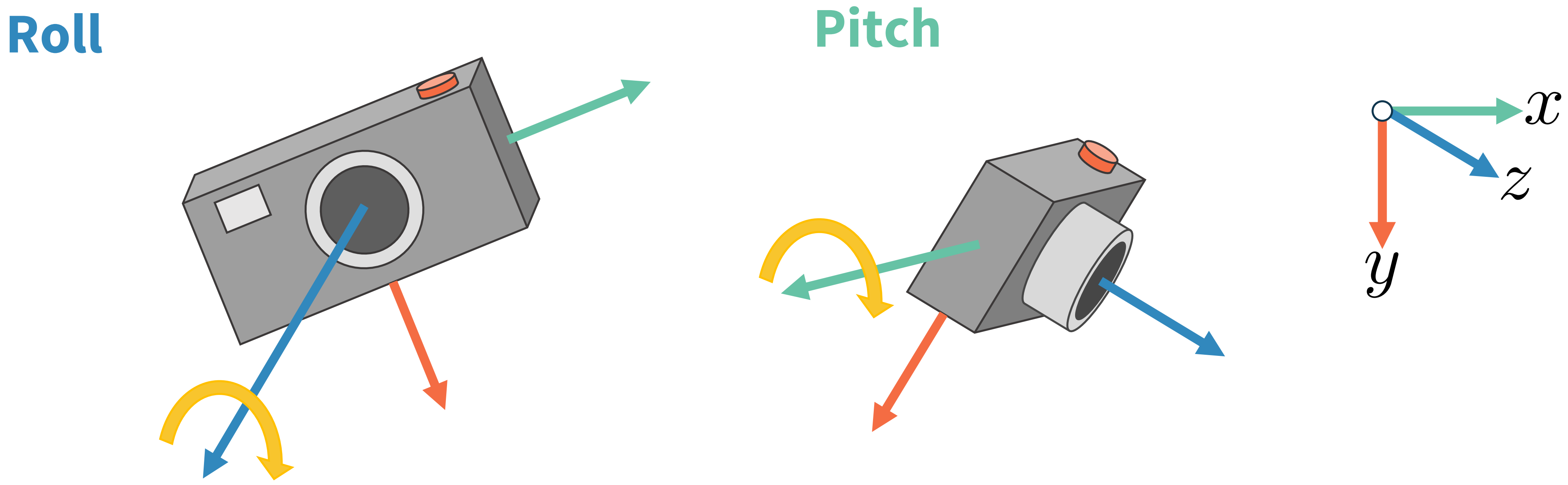
Vanishing points are direclty related with the orientation R of the camera

$$\lambda v = K[R|t]\begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} = KR\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



Camera pose: pitch and roll

Does the NN assume a fixed camera pose or estimate this on the fly?

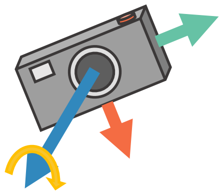


This is strictly related to the position of the horizon and of the vanishing points that depends on the orientation of the cameras

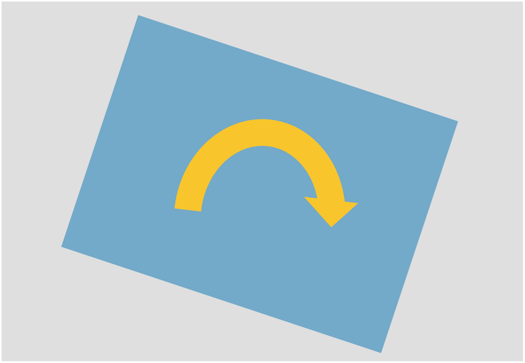
Camera pose: pitch and roll

Does the NN assume a fixed camera pose or estimate this on-the-fly?

Roll

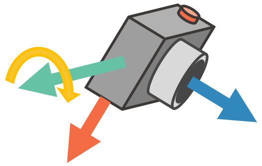


- A different roll is simulated

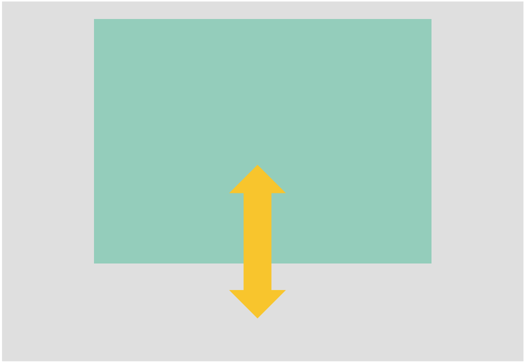


cropping image with +/- 10 degrees rotation

Pitch



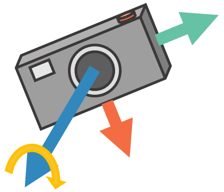
- A different pitch is simulated



cropping images with +/-30 pixels vertical offset.

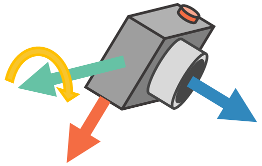
Camera pose: pitch and roll

Does the NN assume a fixed camera pose or estimate this on-the-fly?



Roll

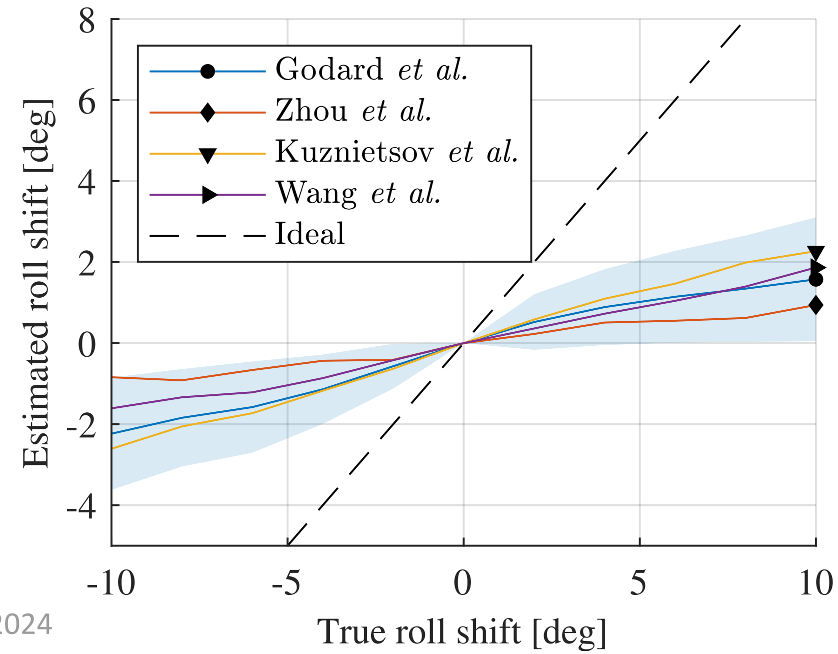
- A different roll is simulated
- Estimate roll angle from depth map

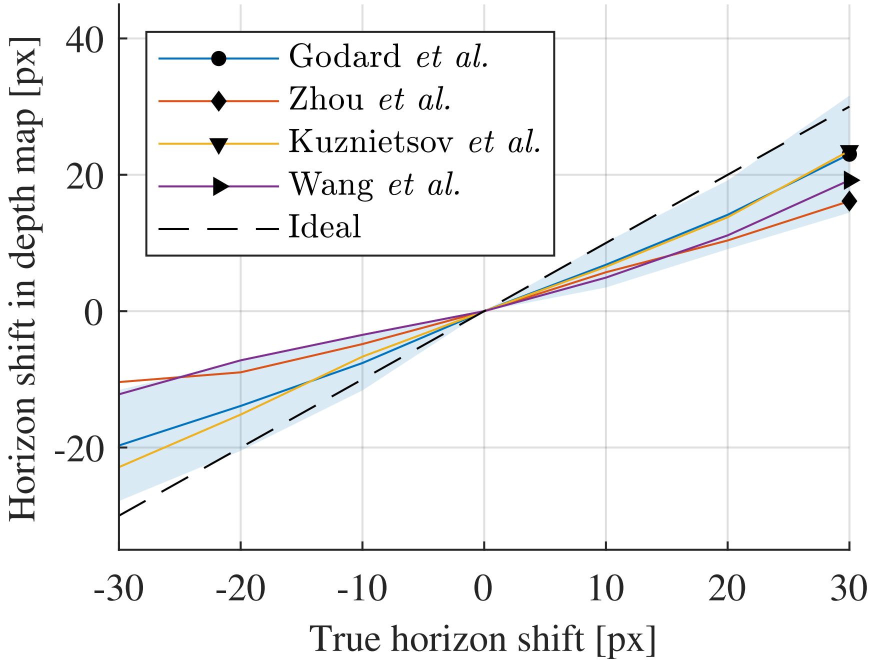


Pitch

- A different pitch is simulated.
- Estimate horizon by fitting a line at infinite depth (0 disparity).

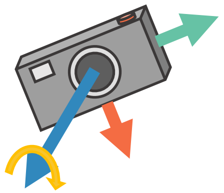
The network **underestimate** both the roll and the pitch





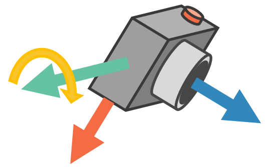
Camera pose: pitch and roll

Does the NN assume a fixed camera pose or estimate this on-the-fly?



Roll

- A different roll is simulated
- Estimate roll angle from depth map



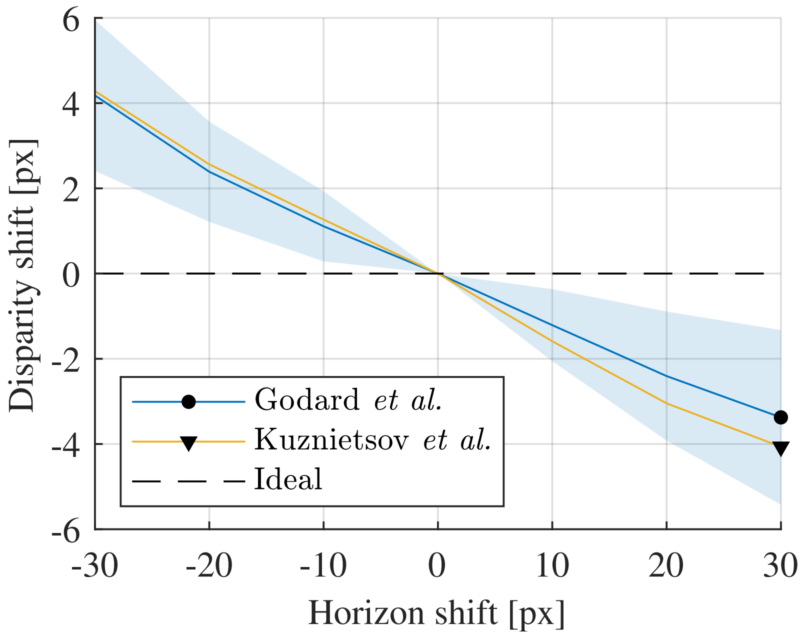
Pitch

- A different pitch is simulated.
- Estimate horizon by fitting a line at infinite depth (0 disparity).

The network underestimate both the roll and the pitch

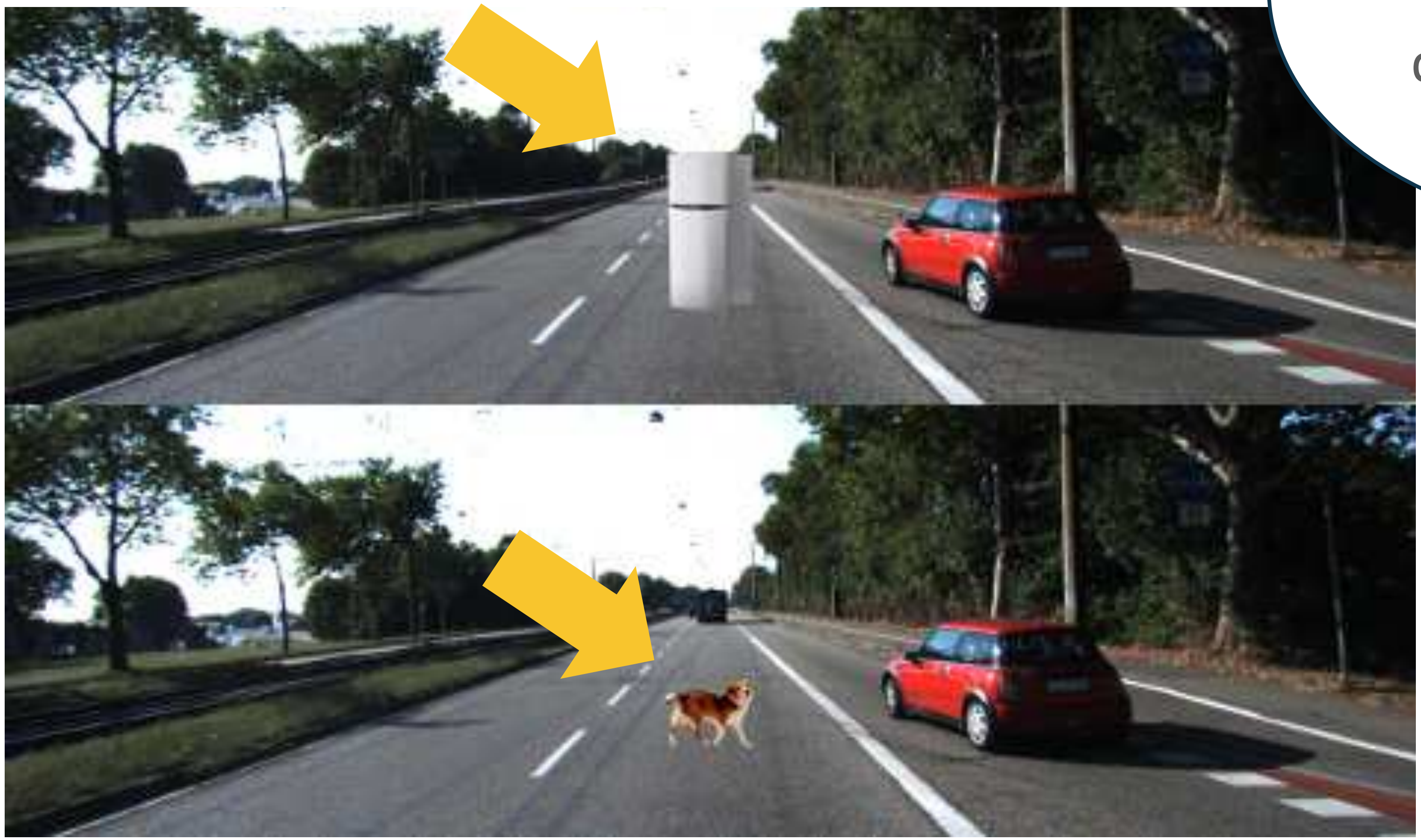
The underestimation of the horizon impact the estimation of the depth (measured in terms of disparity).

The networks look at the vertical image position rather than their distance to the horizon, since the latter does not change when the images are cropped



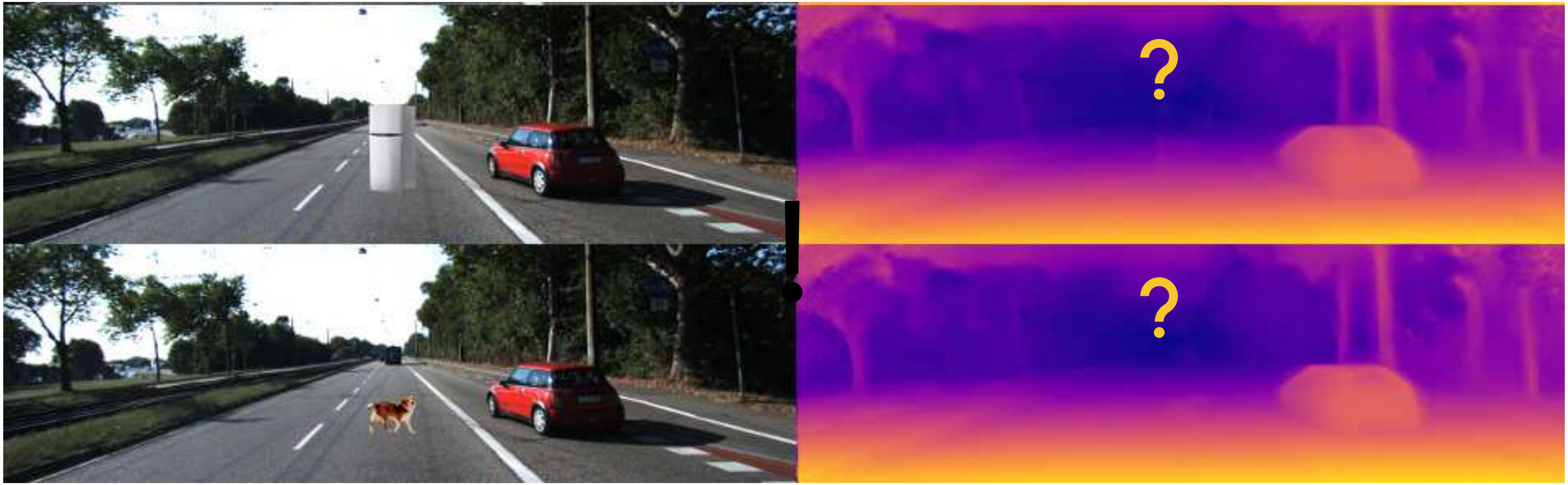
Obstacles

1. only the ground contact point matter
2. no information about the object scale is required



Do you think the NN would be able to estimate the depth of the fridge and of the dog?

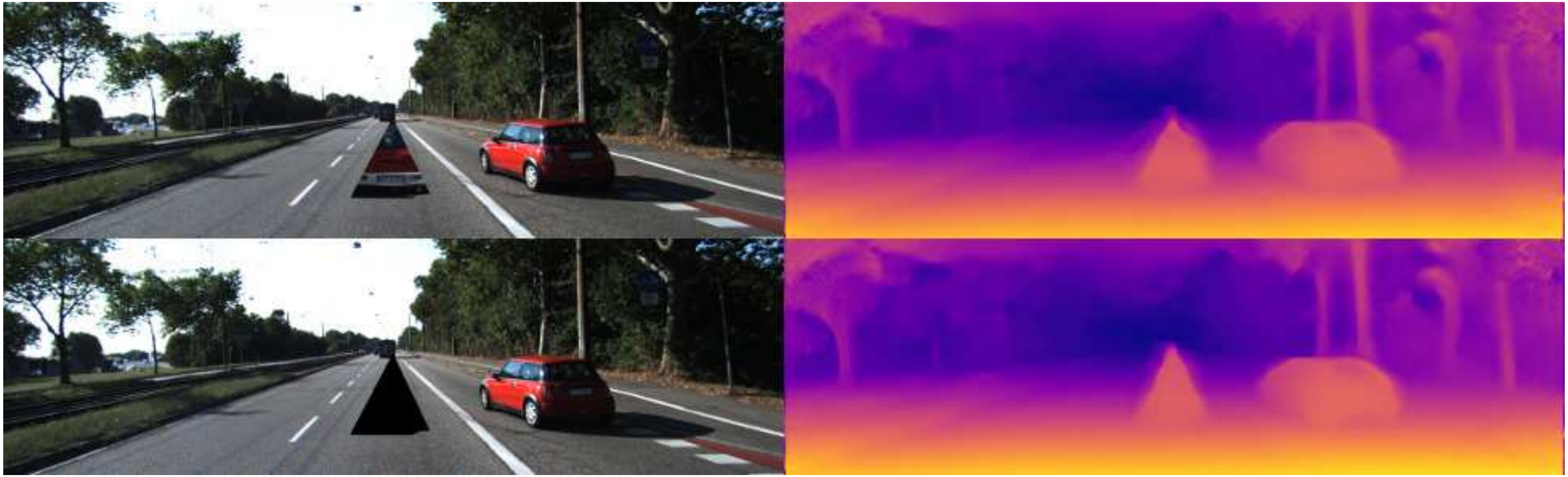
Obstacles



No! Out of distribution objects are not recognized! The network struggle in finding the ground contact and to segment the object to fill in the depth.

Texture

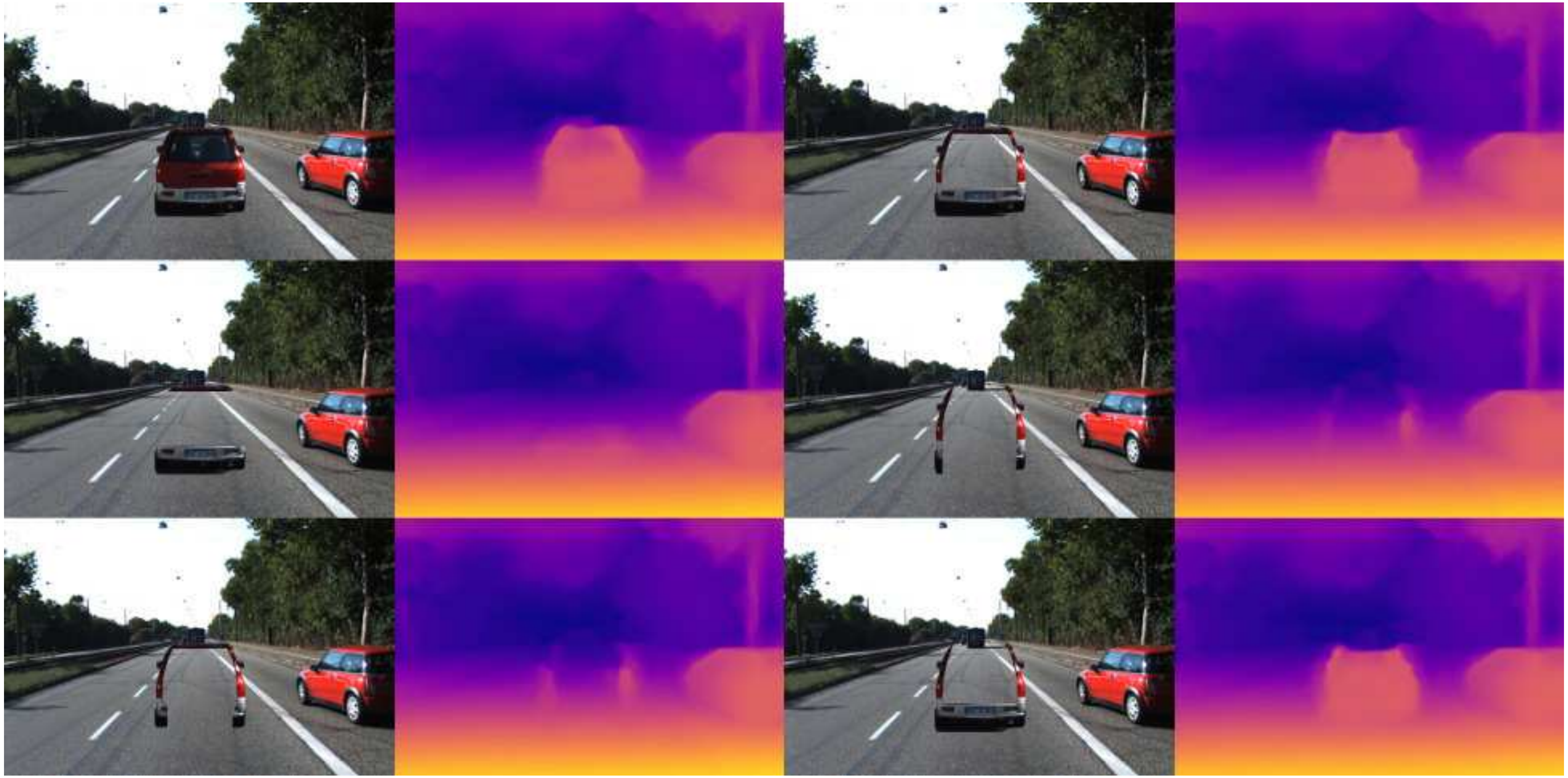
What matters is the ground contact point!



Objects with unfamiliar shapes, either with or without color, as long as their **gorund contact point** can be located effectively are detected and their depth is predicted based on their lower extent.

Texture

What matters is the ground contact point!

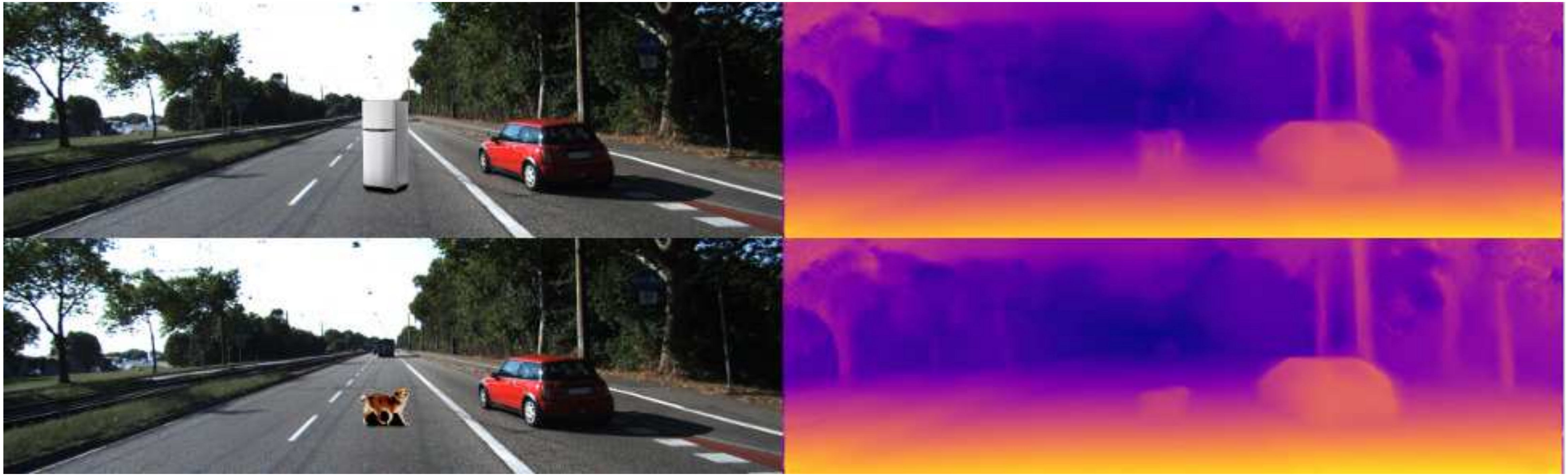


By removing the inner texture of the object, it remains detected in case of a strong bottom edge.

Shadows

Varying the thickness and intensity of the bottom edge impacts on estimated depth. Objects with thick and dark bottom edges are detected.

This suggests that the networks learn to exploit shadows…



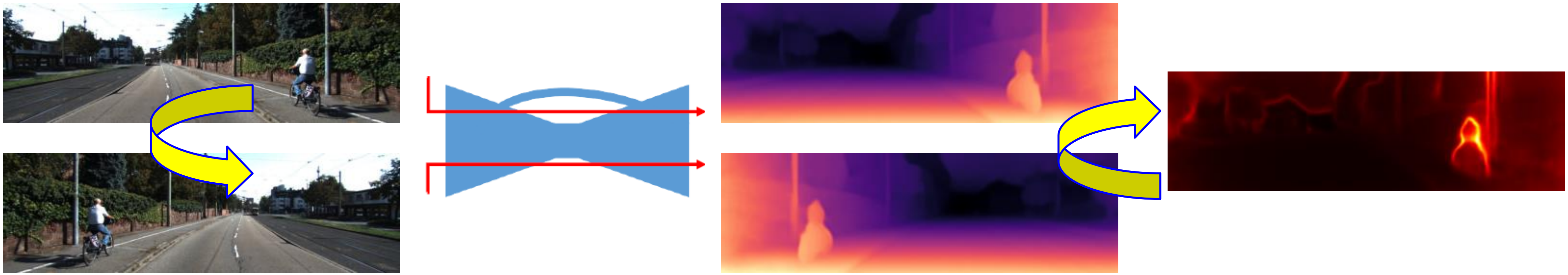
Adding shadows to pasted objects make them appear in the depth map as well.

Uncertainty estimation

A naïve uncertainty estimate can be obtained as a post-processing:

- Estimate two depth maps one from the input image and one from a flipped version
- Measure the difference

This provide an estimate of the depth uncertainty



Aleatoric and epistemic uncertainty

Interestingly Kendall and Gal distinguish between two typo of uncertainty:

- Aleatoric uncertainty captures noise inherent in the observations.

  It's important for:
  - *Large scale data*, where epistemic uncertainty is explained away,
  - *Real-time applications*, to bypass expensive Monte Carlo computations.

- Epistemic uncertainty accounts for uncertainty in the model – uncertainty which can be explained away given enough data.

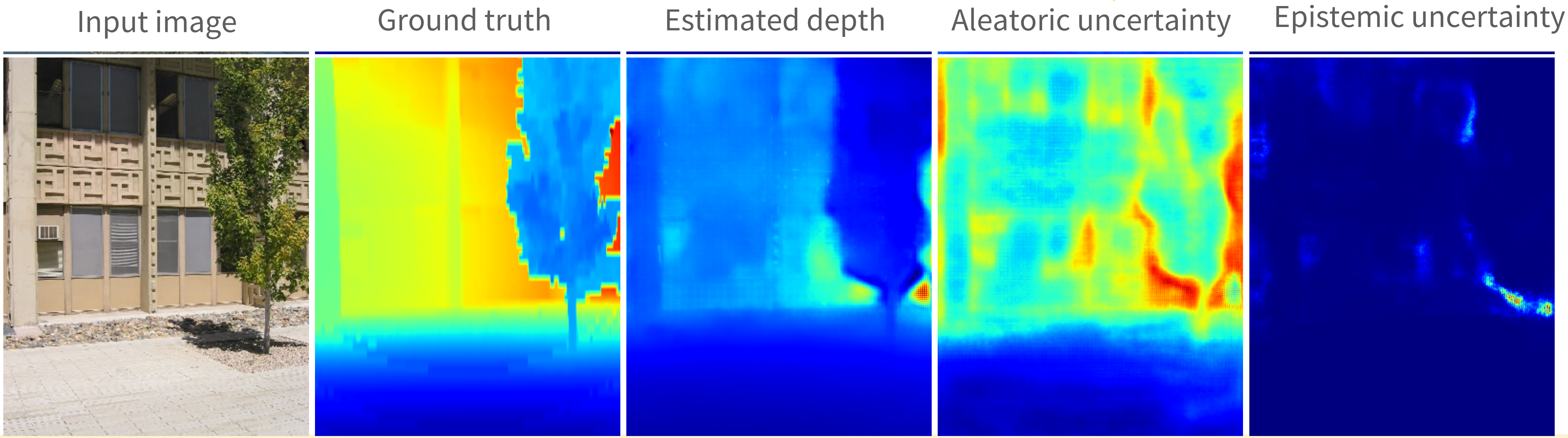
  It's important:
  - *Small datasets* where the training data is sparse.
  - Saf*ety-critical applications*, because epistemic uncertainty is required to understand examples which are different from training data,

Kendall and Gal. "What uncertainties do we need in bayesian deep learning for computer vision?." NIPS 2017

Aleatoric and epistemic uncertainty

- Aleatoric uncertainty captures noise inherent in the observations.
  is modeled by placing a distribution over the output of the model.
  We are interested in how this distribution change w.r.t. the input.
- Epistemic uncertainty accounts for uncertainty in the model

Higher for
- large depths,
- reflective surfaces,
- occlusion boundaries



Kendall and Gal. "What uncertainties do we need in bayesian deep learning for computer vision?." NIPS 2017

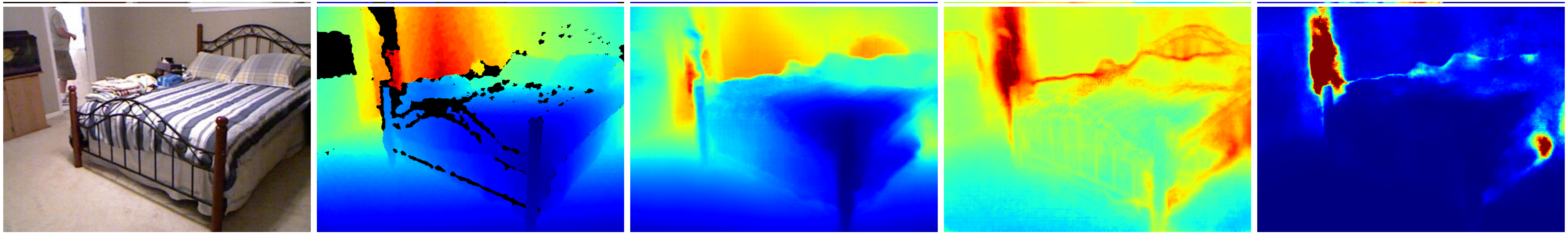
Aleatoric and epistemic uncertainty

- Aleatoric uncertainty captures noise inherent in the observations.
- Epistemic uncertainty accounts for uncertainty in the model:

  is modeled by placing a prior distribution over a model's weights,
  and then trying to capture how much these weights vary given some data.

Higher for object that are rare in the training set

Input image | Ground truth | Estimated depth | Aleatoric uncertainty | pistemic uncertainty



Kendall and Gal. "What uncertainties do we need in bayesian deep learning for computer vision?." NIPS 2017

Understanding single image depth estimation

It is crucial to understand how NNs estimate depths in order to safely apply them in critical application as autonomous driving…

1. *Which are the most **relevant visual cues** in image?*
   - It depends on the type of images indoor/outdoor, mainly a subsets of **edges** and **vanishing points**
2. *How **biased** are depth values in presence of specific objects, shadows, camera orientations?*
   - The **vertical position** is more important than the apparent size
   - Depth depends on the pose of the camera, but changes to the pose are not fully accounted for
   - Objects that do not appear in the training set can be detected, but this detection is not always reliable and depends on factors such as the **presence of a shadow** under the object.
3. *How **reliable** are depth values?*
   - Depth estimation can be fooled by out of distribution objects (epistemic uncertainty)
   - And are typically less reliable on distant objects and at boundaries of objects
   - Several methods exists to assess the reliability of depth estimates

Demo

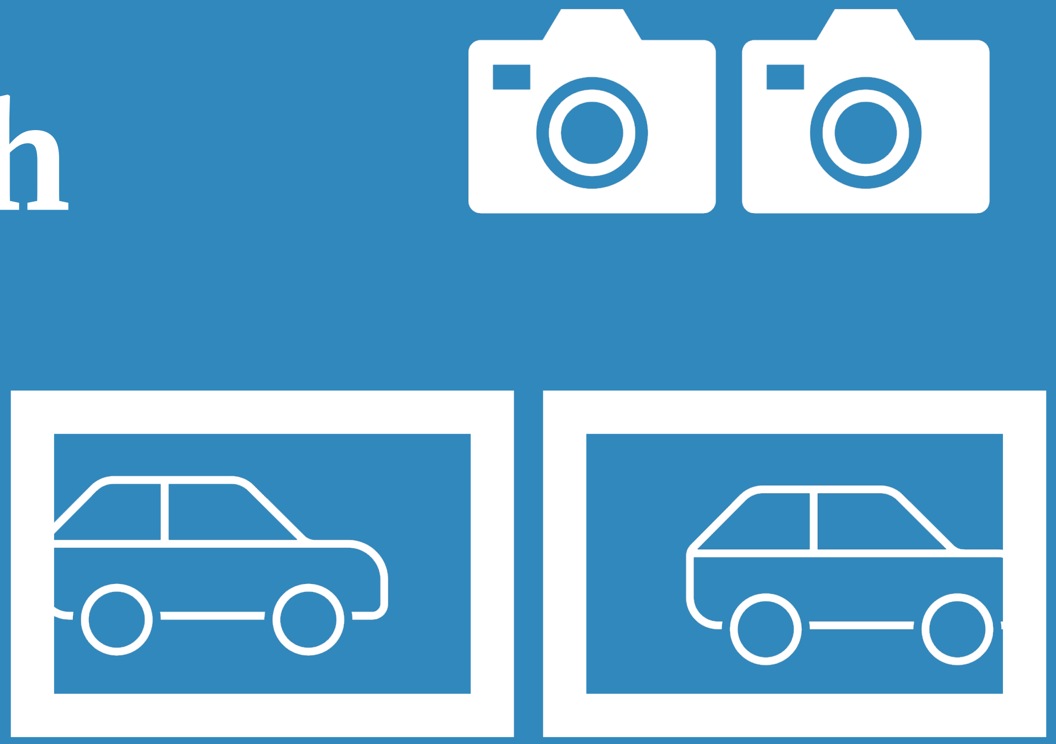
Let's try to estimate depth from a single image using AdaBin
A Unet-like architecture with adaptive bin of depths

- You can download the pretrained models "AdaBins_nyu.pt" and "AdaBins_kitti.pt"
- You can download the predicted depths in 16-bit format for NYU-Depth-v2 official test set and KITTI Eigen split test set

https://github.com/andreadalcin/DNN3D

Estimating depth from stereo

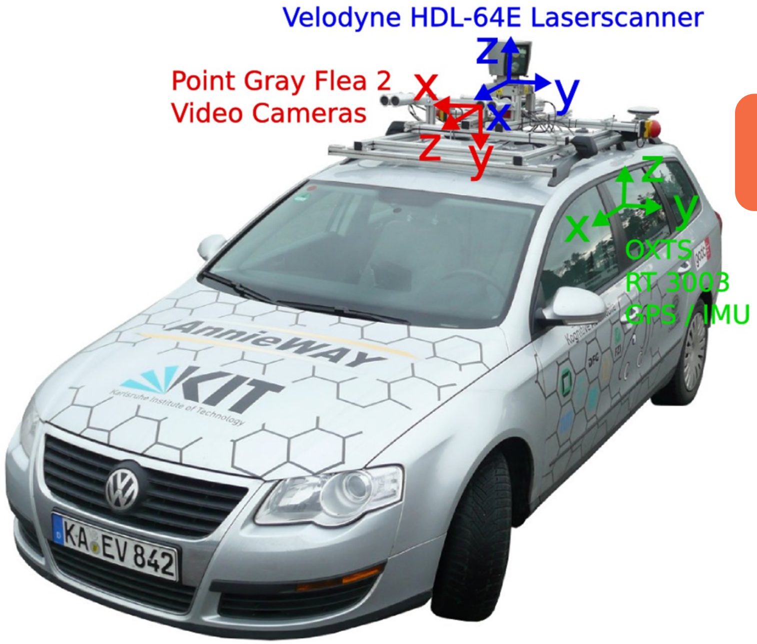
Geometric supervision



Acquiring target 3D data is difficult

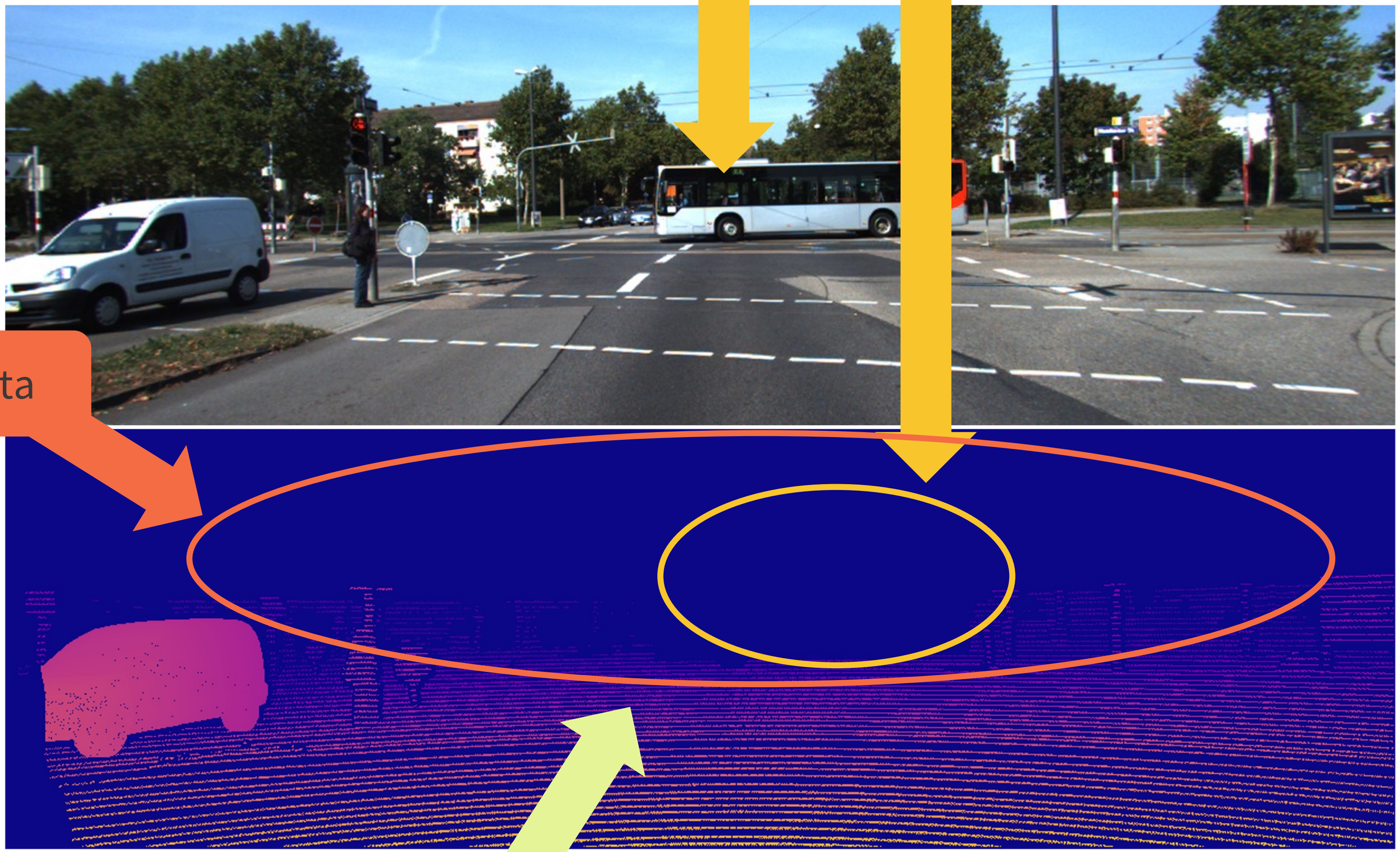
Don’t work outside

Costly equipement



Missing moving objects

No data



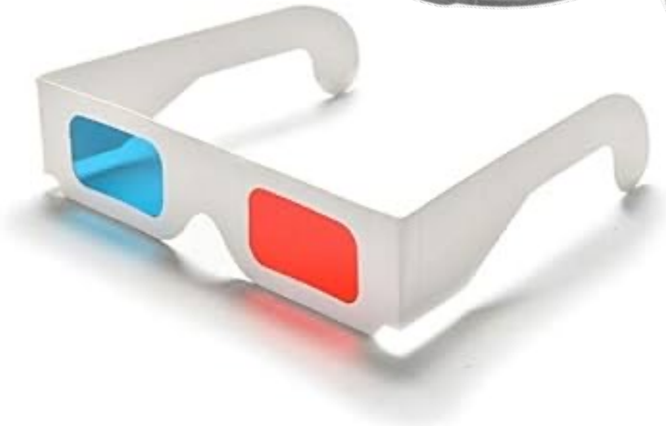
Sparse measurements

Credits Clément Godard

Stereoscopy

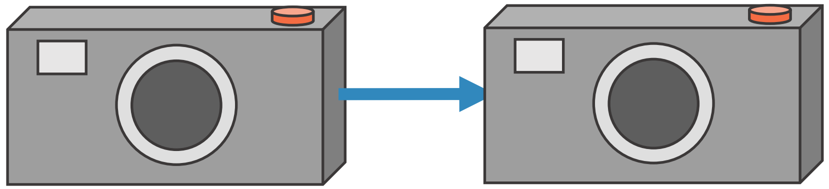
Around 1830, the stereoscope. A couple of two-dimensional images captured from a slightly different perspective, could be recombined by the brain to provide a three-dimensional image.

Special stereoscopic cameras were developed to take the left and right images simultaneously, with two lenses separated by around the same distance as human eyes.

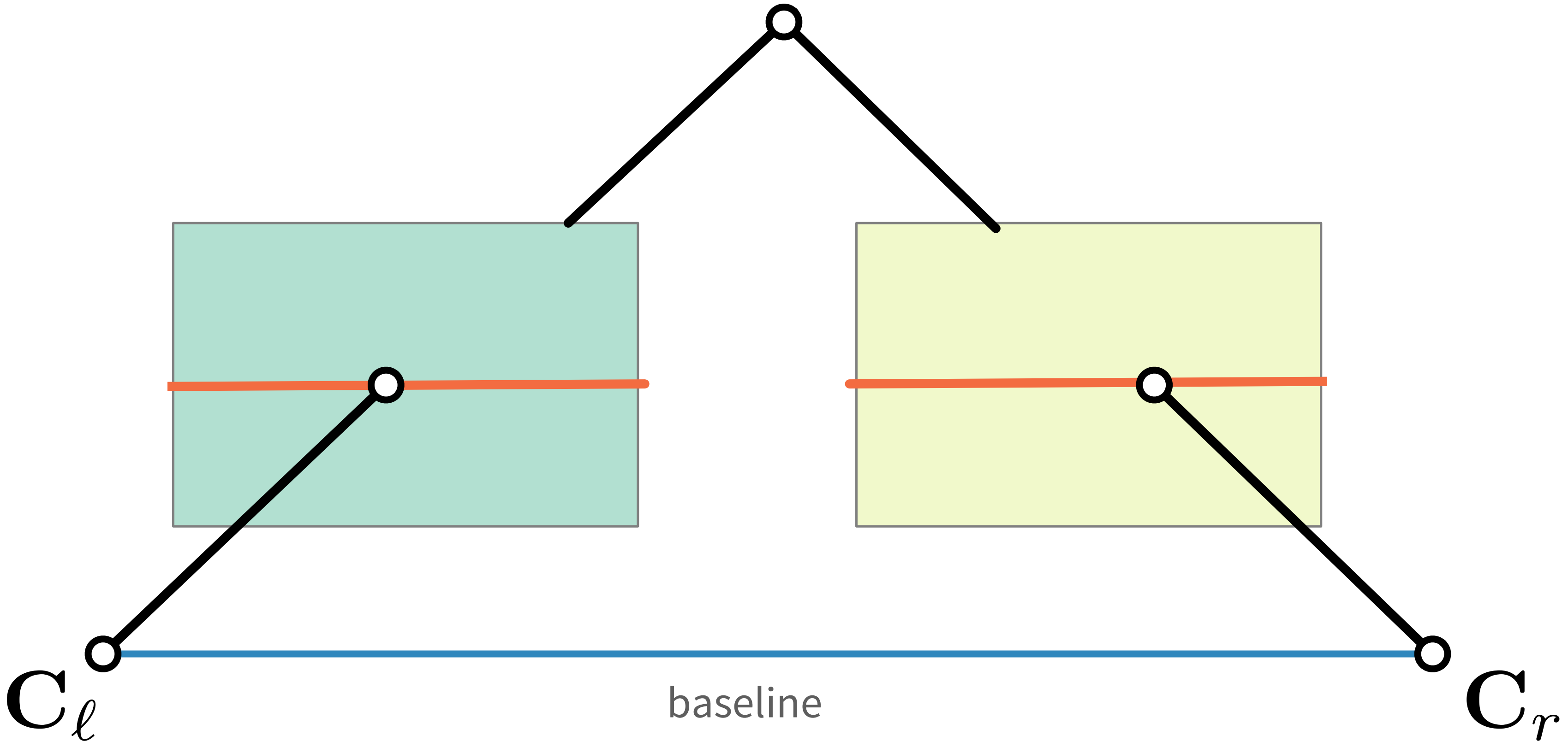


Motion is a strong cue to make depth estimation not (so) ambiguous!

Calibrated stereo pair



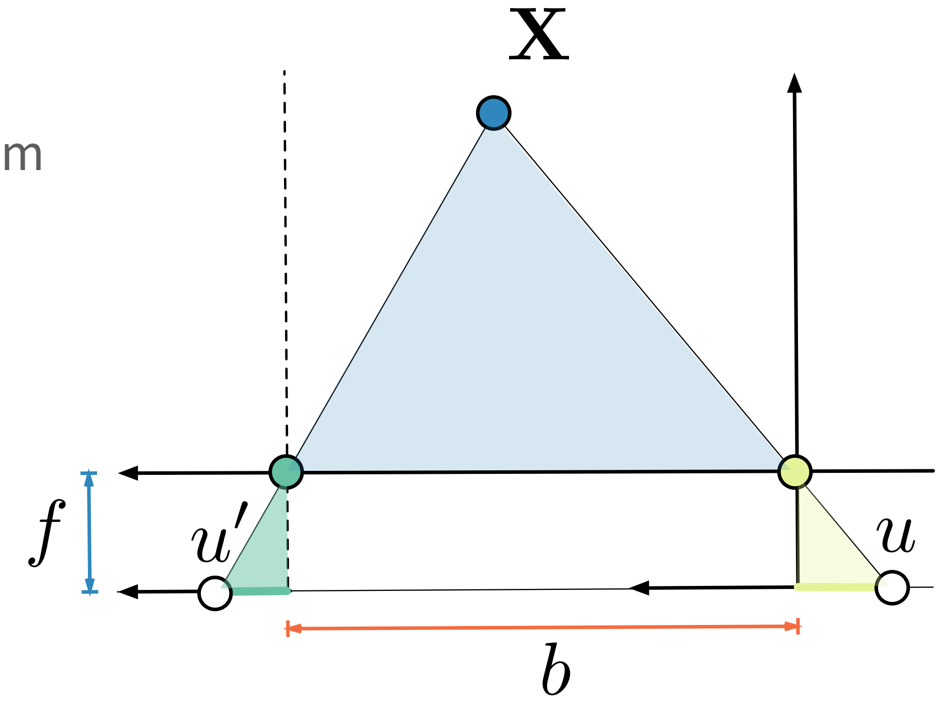
The baseline is parallel to both image planes is known.



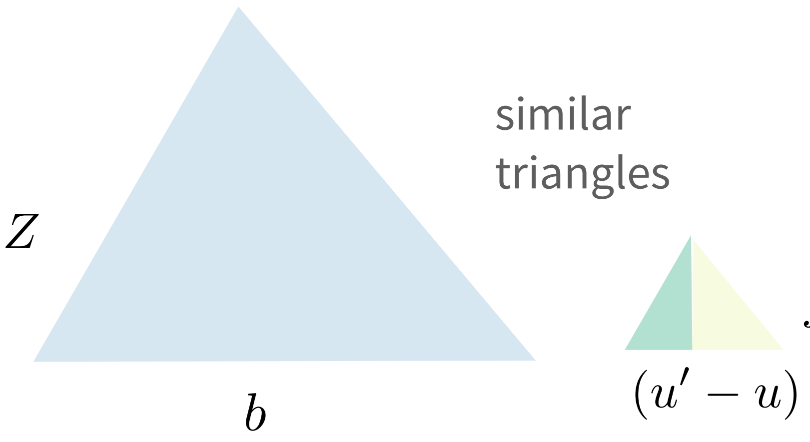
Calibrated stereo pair

When the camera are calibrated (e.g., we know the focal length f and the baseline b), it is possible to deduce the coordinates Z, from binocular disparity $(u' - u)$:

$$\begin{cases} \dfrac{f}{Z} = \dfrac{-u}{X} \\ \dfrac{f}{Z} = \dfrac{-u'}{X - b} \end{cases}$$



from which we obtain $Z = \frac{bf}{u' - u}$.



Note that when b is unknown, 3D reconstruction is possible only up to a **scaling factor**.

Binocluar disparity: the difference in image location of an object seen by the left and right cameras.

The key observation is that close objects have a larger disparity than further ones

From disparity it's possible to recover the depth.

http://vision.middlebury.edu/stereo/data/

Binocluar disparity: the difference in image location of an object seen by the left and right cameras.

The key observation is that close objects have a larger disparity than further ones.

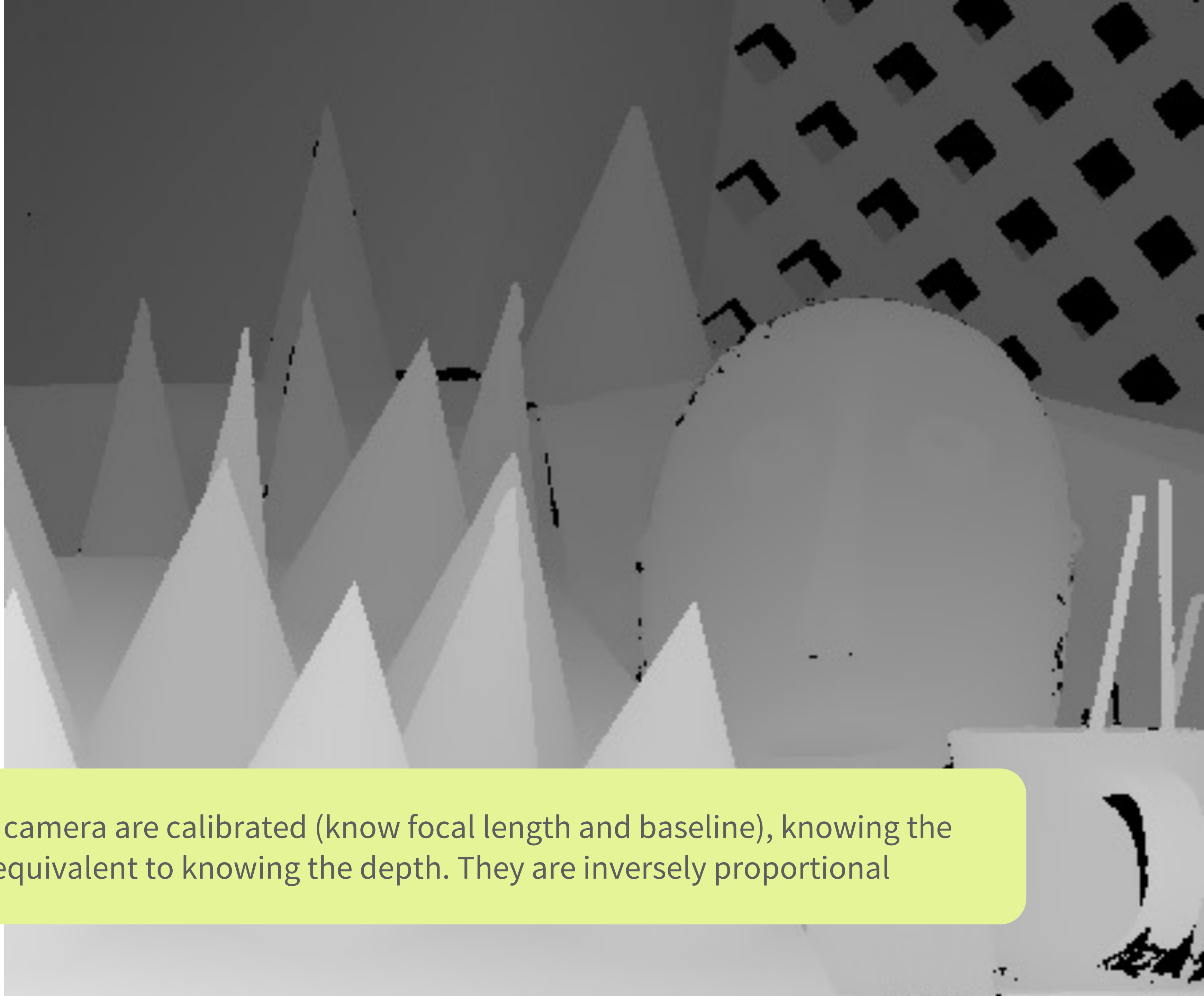
From disparity it's possible to recover the depth.

http://vision.middlebury.edu/stereo/data/

Disparity Map

Estimate **at each image point** $\boldsymbol{x}$, the depth of the scene point X as inversely proportional to the displacement between $\boldsymbol{u}$ and $\boldsymbol{u}'$

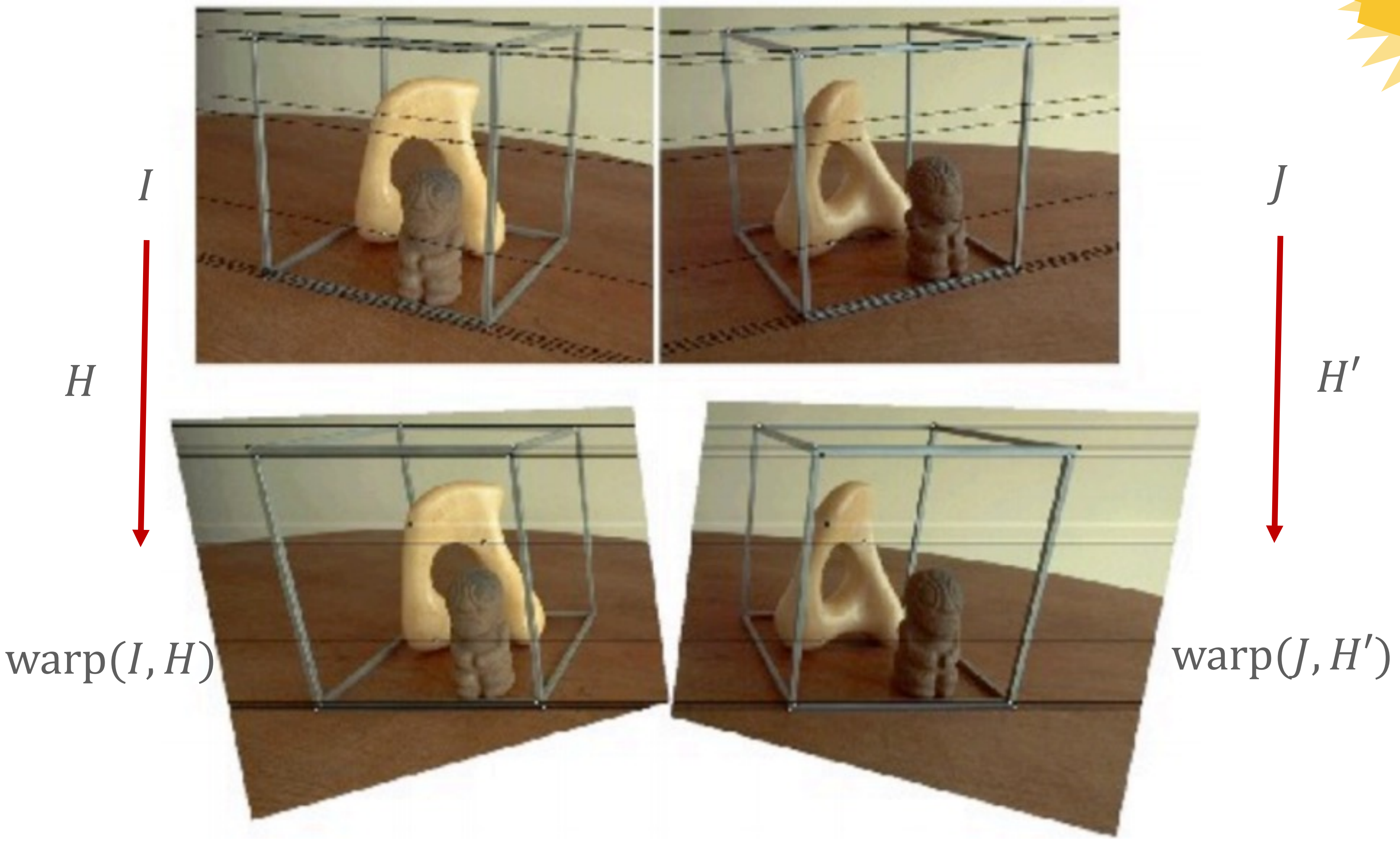
When the cameras are parallel, then the search is much more convenient, as it has to be performed row-wise only



When the stereo camera are calibrated (know focal length and baseline), knowing the disparity is equivalent to knowing the depth. They are inversely proportional

What if camera are not parallel? No problem!

Geometric wisdom

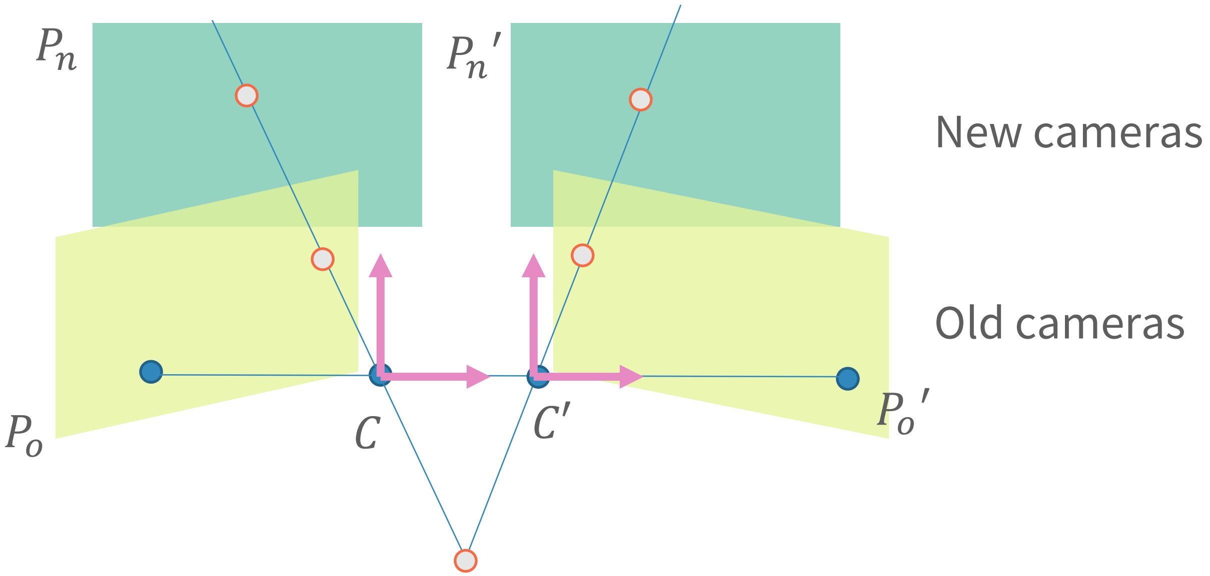


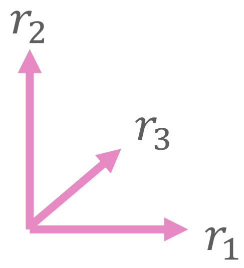
Stereo rectification

Let's write the new cameras in term of their centers of projection:

$$P_n = K[R \mid -RC], P'_n = K[R \mid -RC']$$

The rotation is the same for the new cameras: $R = \begin{bmatrix} r_1^\top \\ r_2^\top \\ r_3^\top \end{bmatrix}$





$$r_1 = \frac{C - C'}{||C - C'||}$$

$$r_2 = \frac{k \times r_1}{||k \times r_1||}$$

$$r_3 = r_1 \times r_2$$

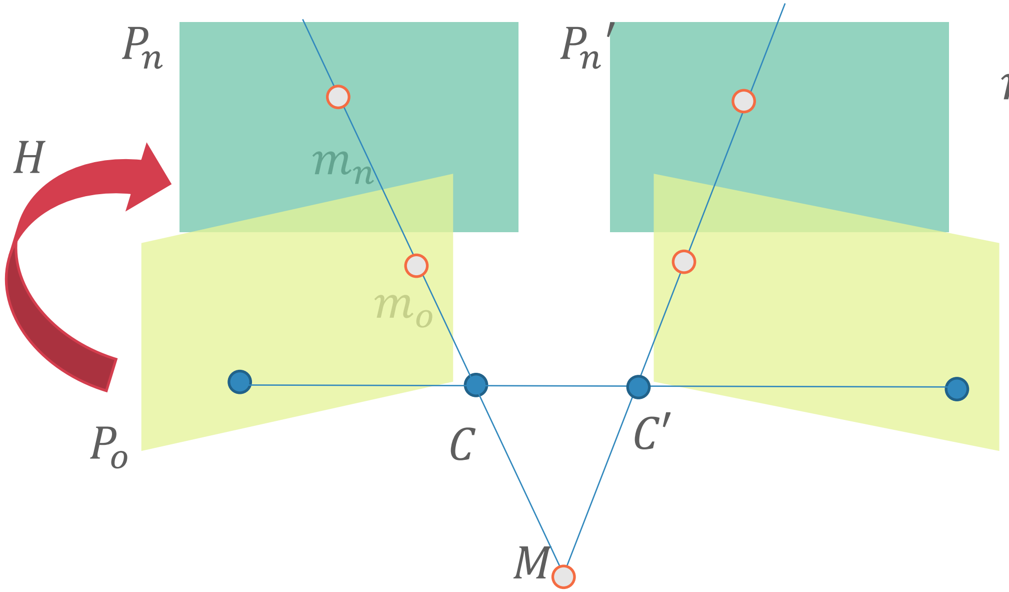
where k is r_{3o}

Rectification

After rectification images are parallel to the baseline.
The idea is to define two new projection matrices $P_n, P_n{}'$ obtained by rotating the cameras and keeping fixed the centers of projection.

Every point M is mapped to $m_o \cong P_o M, m_n \cong P_n M$.

$$\begin{cases} M = C + \lambda [P_{o\,1:3}^{-1} m_o | 0]^\top \\ M = C + \lambda [P_{n\,1:3}^{-1} m_n | 0]^\top \end{cases}$$

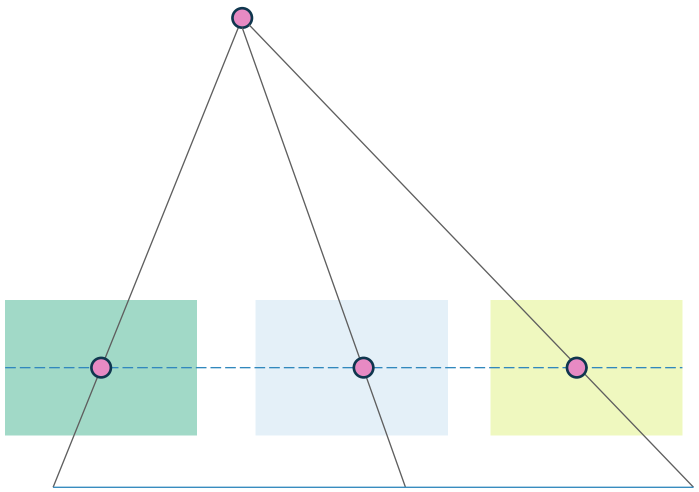


$$m_n = P_{n1:3}\, P_{o\,1:3}^{-1} m_o$$

This is a 3x3 invertible matix: an homography that depends on the camera parameters

Image reconstruction as supervision

The trick is to pose the problem as an image reconstruction one.



Pretext task: **View synthesis**

- Given an image
- Given the 3D scene (but in our case we want to estimate this!)
- Given the displacement of the cameras (in general the relative pose)

Synthetize a novel image form the point of view defined by the relative pose

In practice, the network learn just to move each pixel by the right horizontal displacement by looking at several left-right pairs.
Hence the network has an internal understanding of disparity and hence of depth.

Image reconstruction as supervision

The training set is given by pair of RGB images $TR = \{(I_L, I_R)\}$, **the depth is no required**!

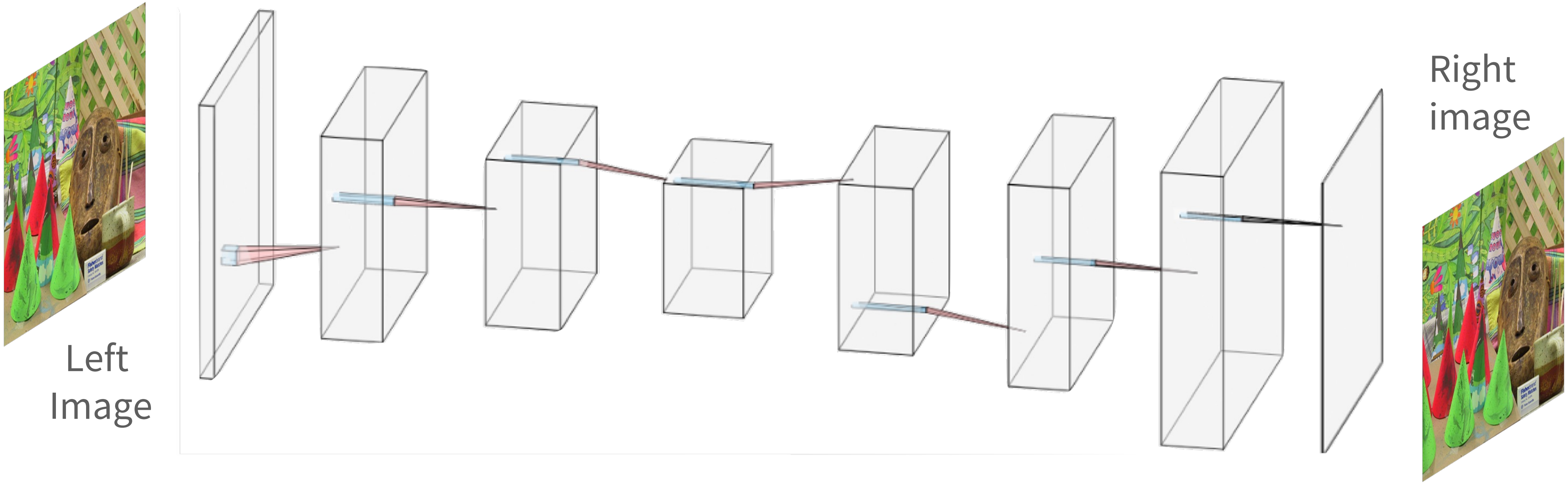
The loss is simply image reconstruction. **It's a self-supervised problem.**

Left
Image

Right
image

Image reconstruction as supervision

Let's start again with a naïve encoder-decoder model



It works! Hovewer depth perception is latend and we need a way to extract it.

We need an **interpretable** internal representation. Since we are working with a calibrated stereo rig, the obvous choice is disparity.

Image reconstruction as supervision: Deep 3D

Introduce a *differentiable* way to take I_L and render a novel view close to I_R.

I_L I_R

Xie, Junyuan, Ross Girshick, and Ali Farhadi. "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks." ECCV, 2016.

Image reconstruction as supervision: Deep 3D

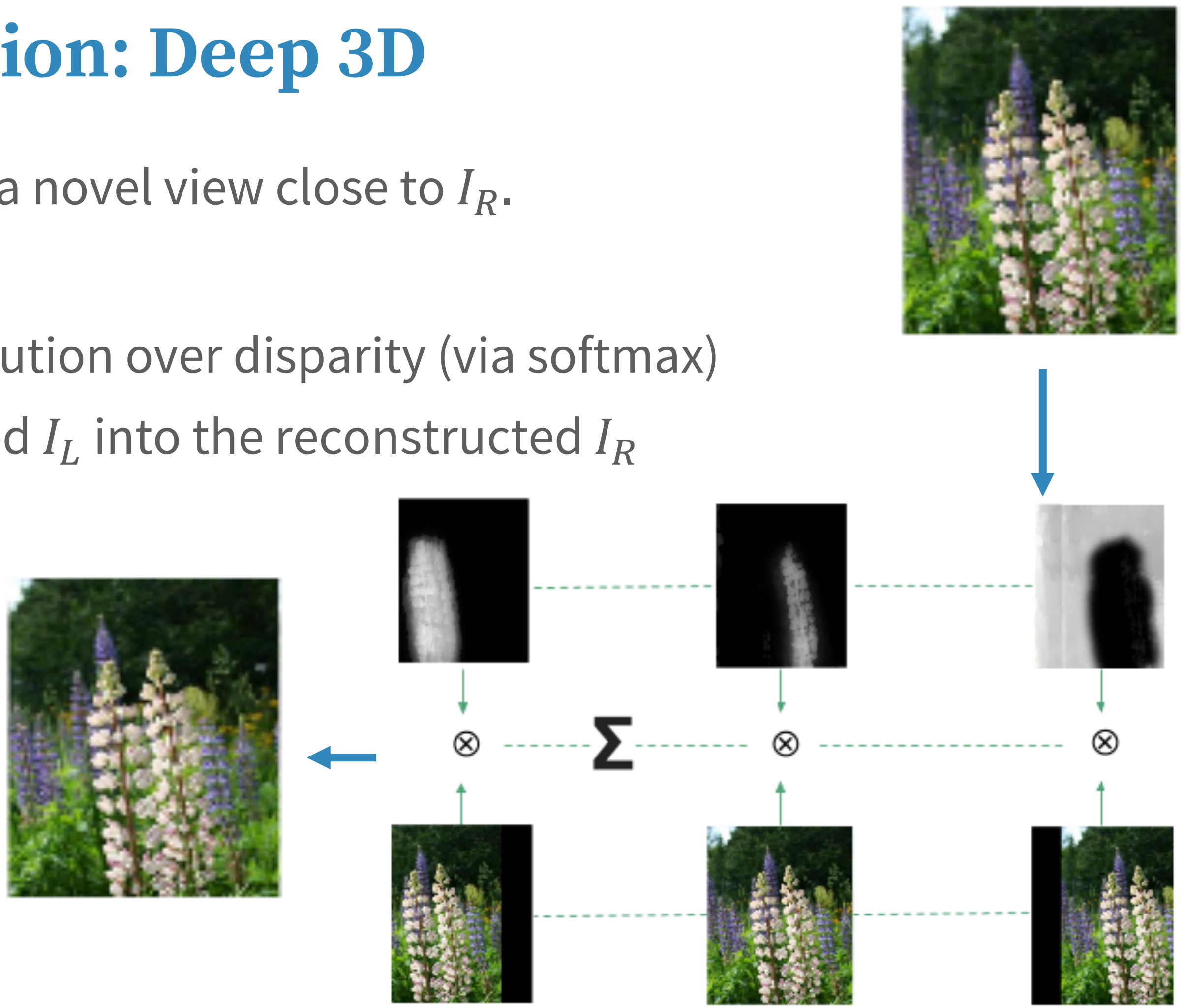
Introduce a *differentiable* way to take I_L and render a novel view close to I_R.

1. Each pixel predicts a discrete probability distribution over disparity (via softmax)
2. Probabilities are used as weights to blend shifted I_L into the reconstructed I_R

✓ Work better than predicting disparity directly

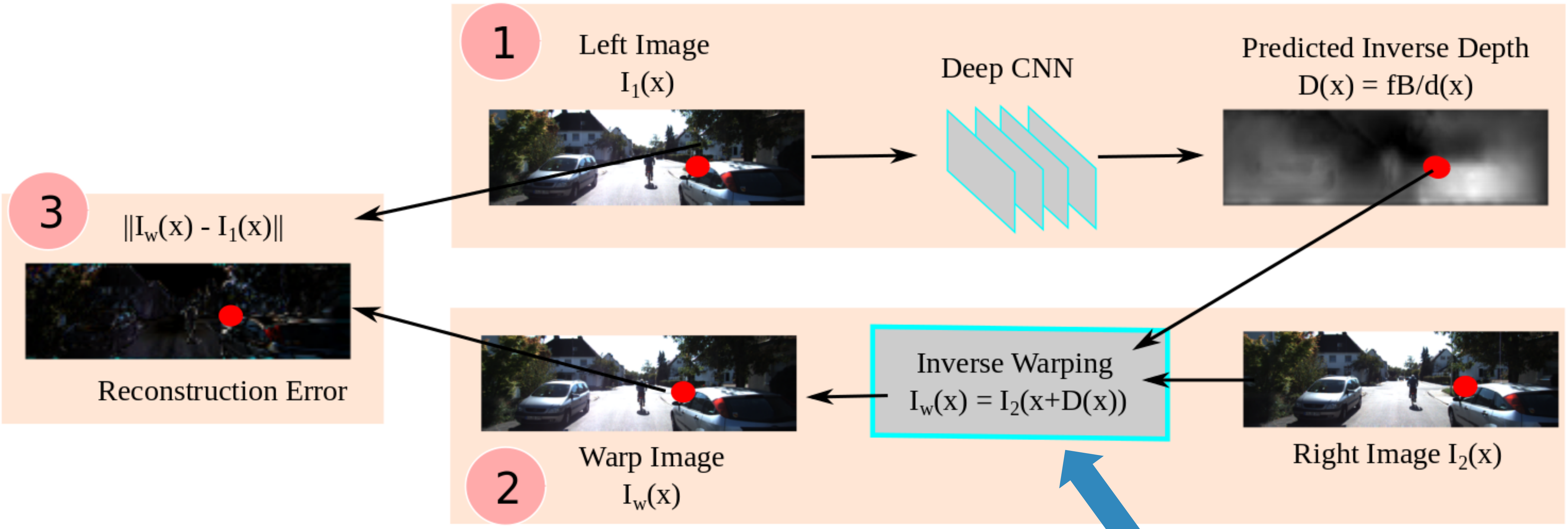
✗ Memory consuming: for large image you must represent all the disparities

✗ No single value disparity predicted: noisy result



Xie, Junyuan, Ross Girshick, and Ali Farhadi. "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks." ECCV, 2016.

Image reconstruction as supervision



1. Predict the depthmap of I_L
2. Use the inverse depth as a disparity to wrap I_R
3. Minimize the reconstruction error between the warped image and I_L

✗ Warping is not differentiable; the authors propose several ad-hoc strategy to

✗ Poor quality

You can make this step differentiable to make it easier to optimize:
- Backward mapping
- Bilinear interpolaiton

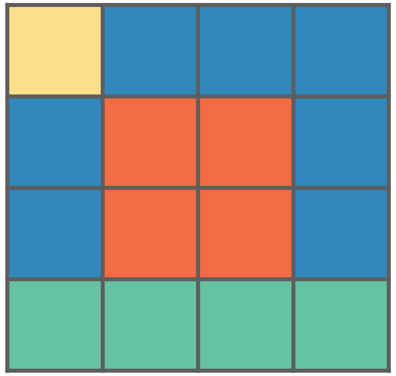
Garg, Ravi, et al. "Unsupervised cnn for single view depth estimation: Geometry to the rescue." ECCV 2016.

Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

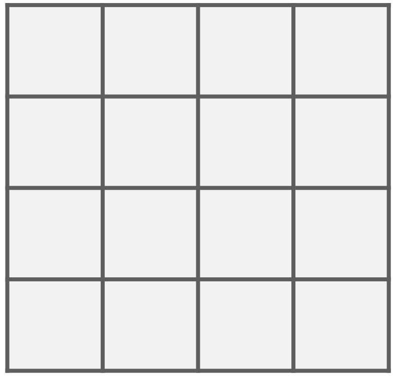
Source



Disparity

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

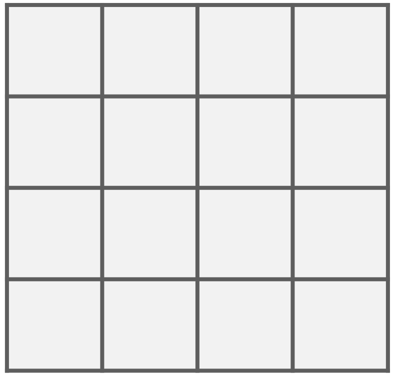
Target



Backward mapping

Where do target pixel comes from?

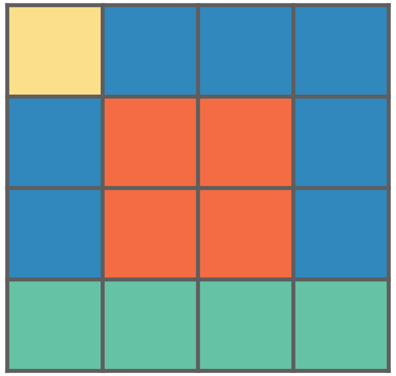
Target



Disparity

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

Source

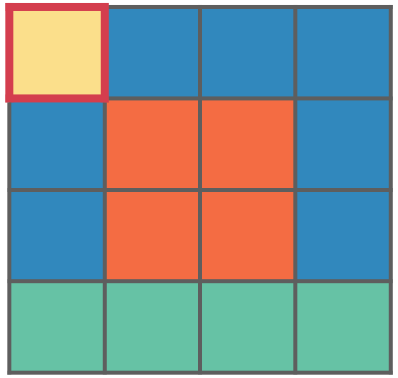


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

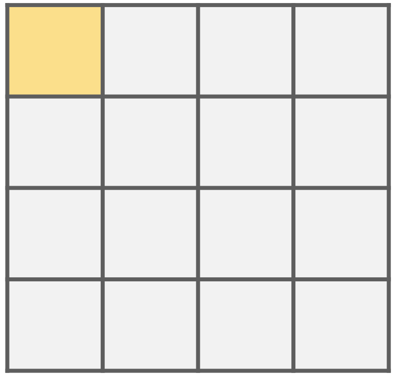
Source



Disparity

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

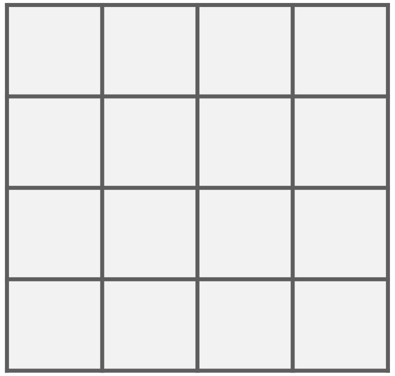
Target



Backward mapping

Where do target pixel comes from?

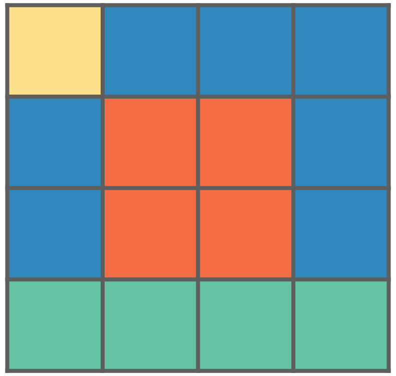
Target



Disparity

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

Source

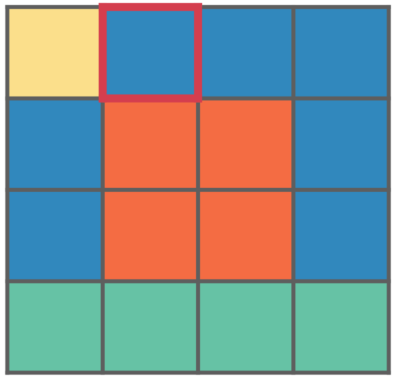


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

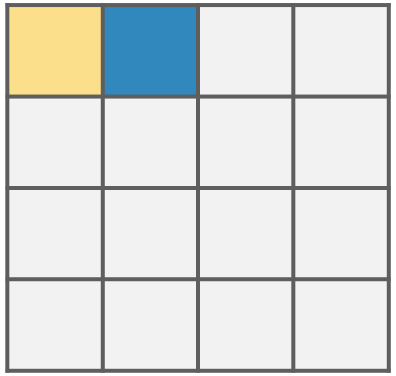
Source



Disparity

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

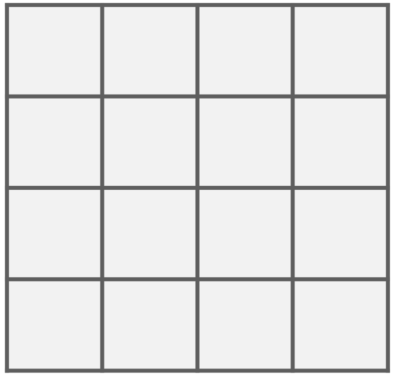
Target



Backward mapping

Where do target pixel comes from?

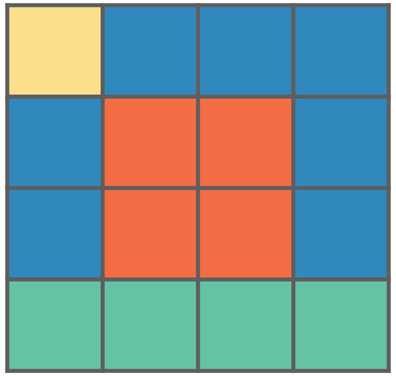
Target



Disparity

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

Source

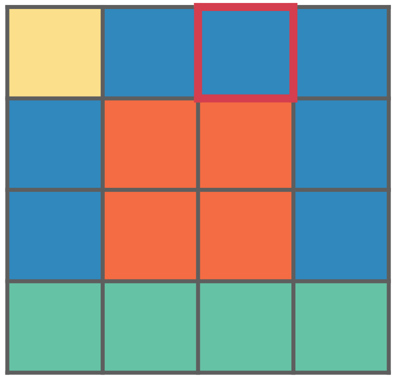


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

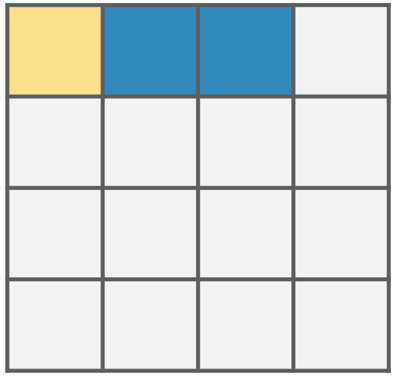
Source



Disparity

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

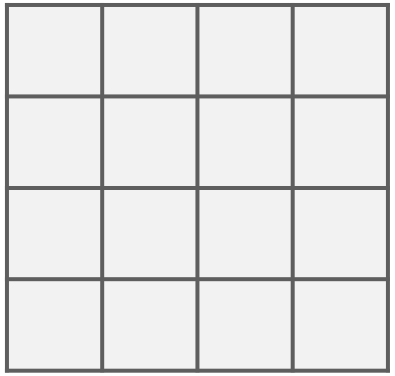
Target



Backward mapping

Where do target pixel comes from?

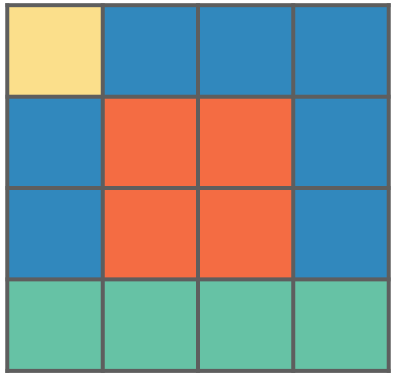
Target



Disparity

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

Source

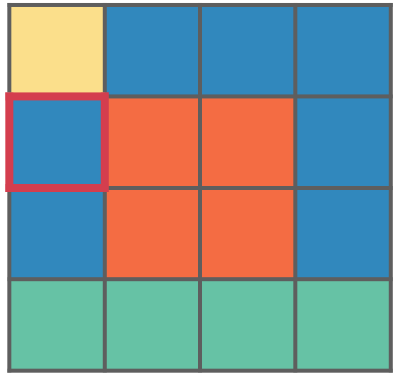


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

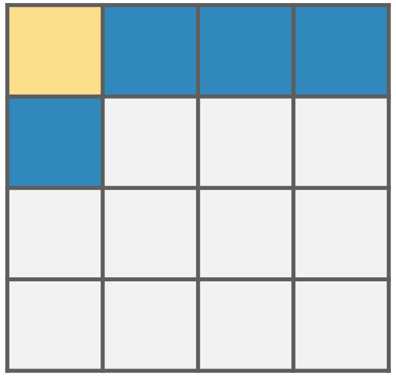
Source



Disparity

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

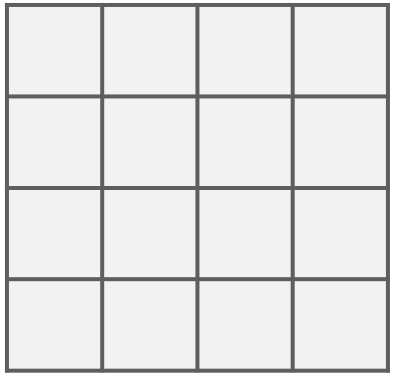
Target



Backward mapping

Where do target pixel comes from?

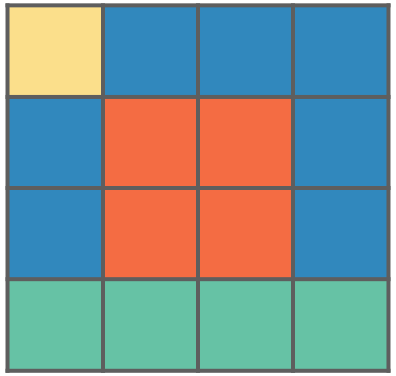
Target



Disparity

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

Source

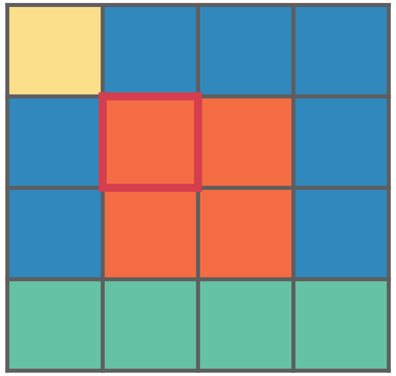


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

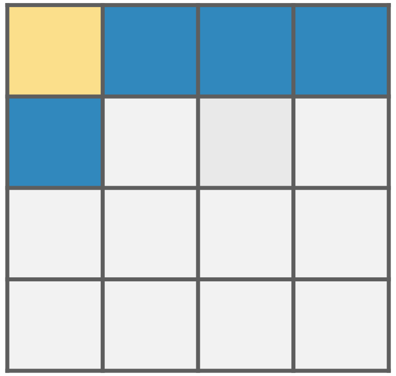
Source



Disparity

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

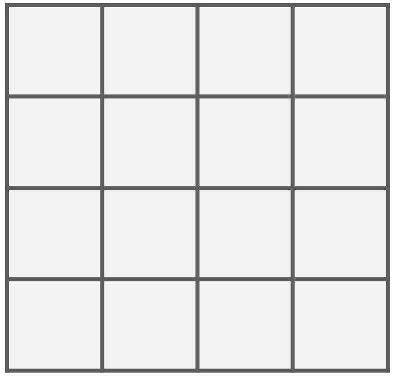
Target



Backward mapping

Where do target pixel comes from?

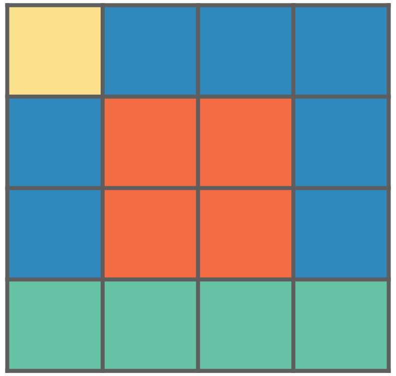
Target



Disparity

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

Source

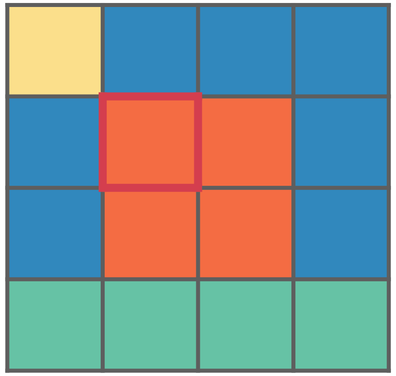


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

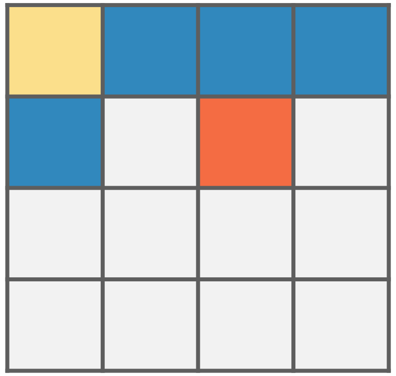
Source



Disparity

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

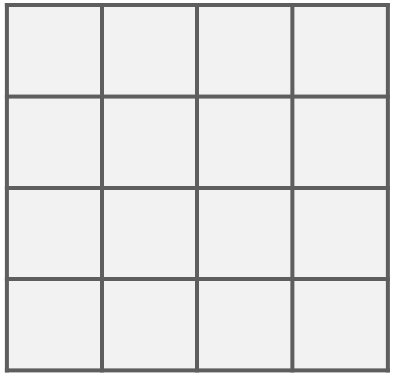
Target



Backward mapping

Where do target pixel comes from?

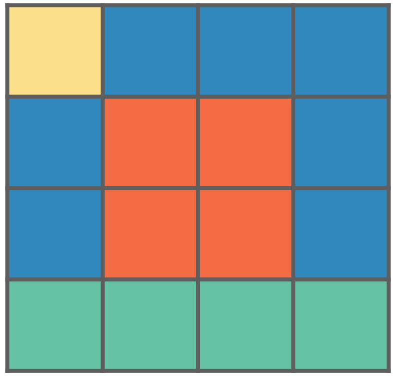
Target



Disparity

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

Source

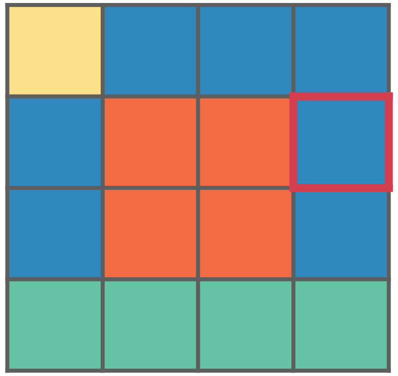


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

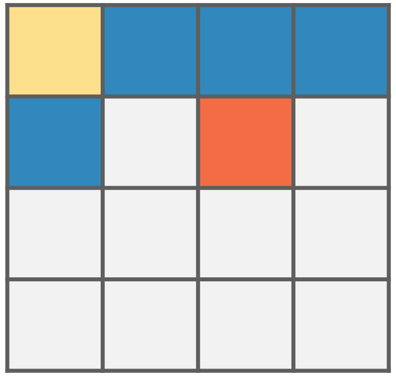
Source



Disparity

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

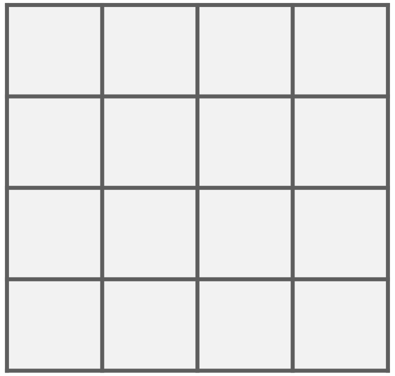
Target



Backward mapping

Where do target pixel comes from?

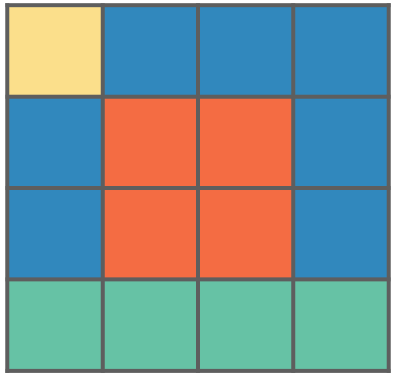
Target



Disparity

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

Source

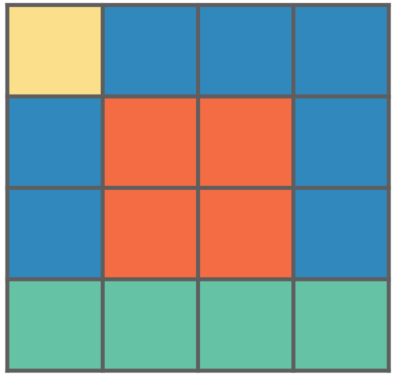


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

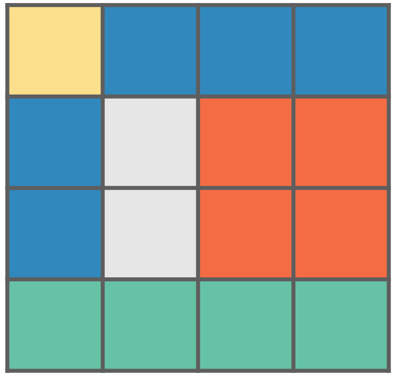
Source



Disparity

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

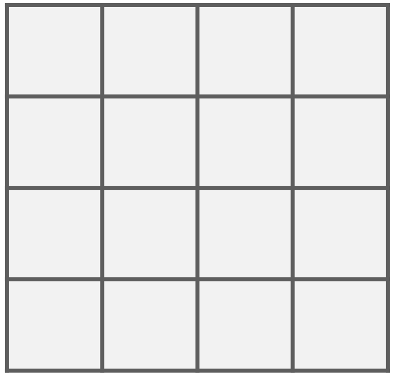
Target



Backward mapping

Where do target pixel comes from?

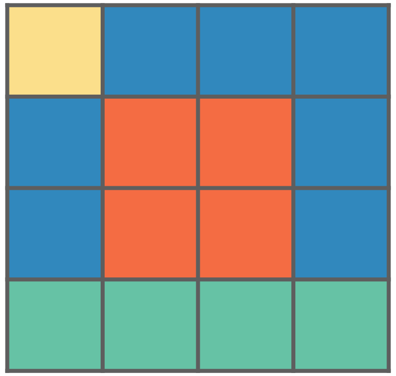
Target



Disparity

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

Source

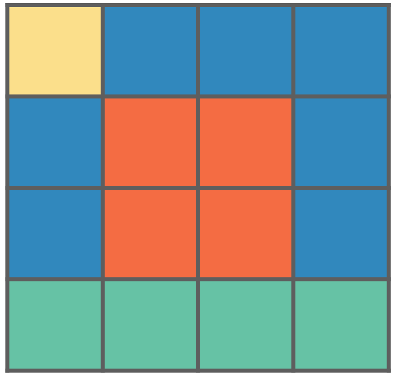


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

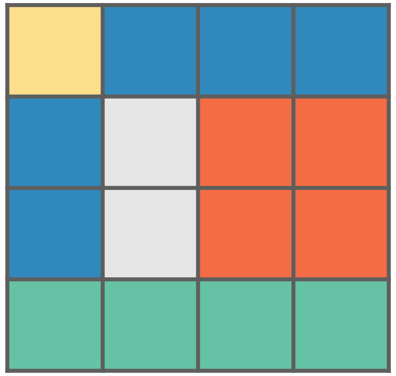
Source



Disparity

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

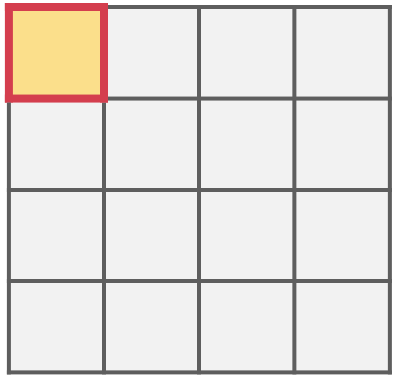
Target



Backward mapping

Where do target pixel comes from?

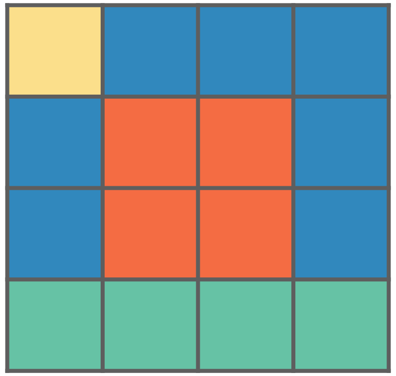
Target



Disparity

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

Source

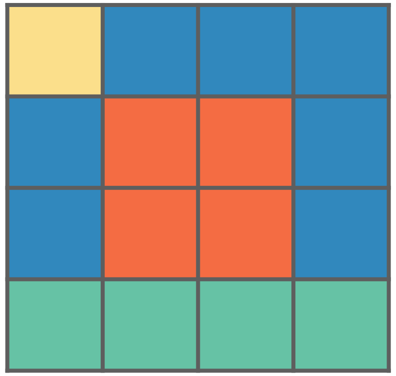


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

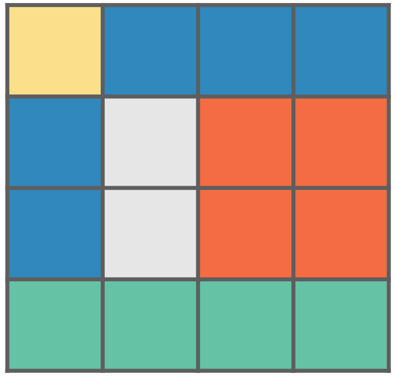
Source



Disparity

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

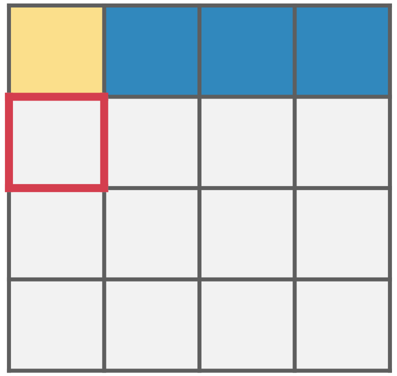
Target



Backward mapping

Where do target pixel comes from?

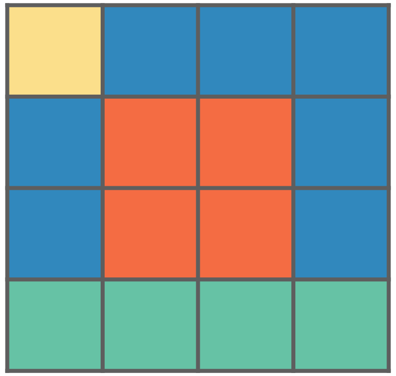
Target



Disparity

0	0	0	0
$\boxed{1}$	1	1	0
1	1	1	0
0	0	0	0

Source

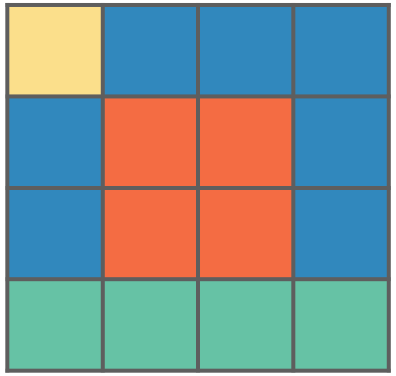


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

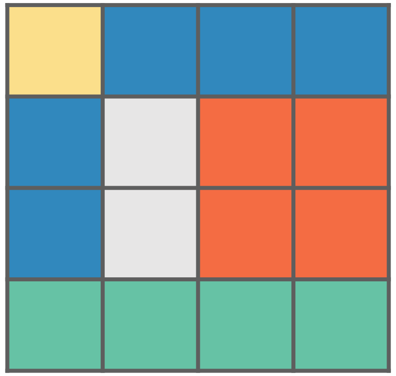
Source



Disparity

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

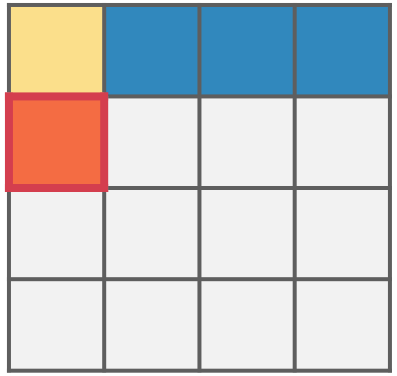
Target



Backward mapping

Where do target pixel comes from?

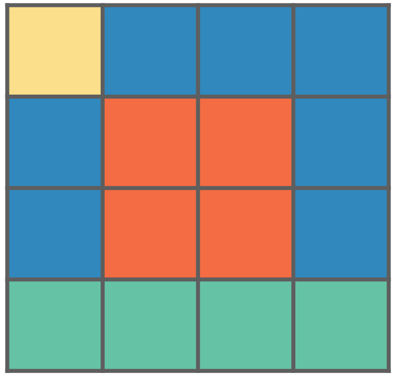
Target



Disparity

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

Source

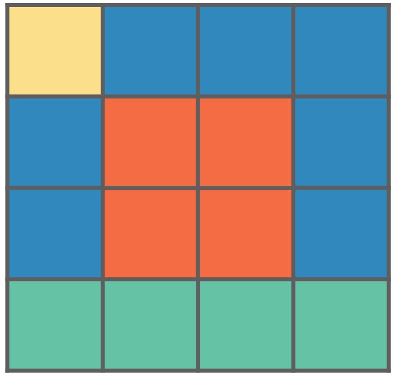


Making the warping differentiable – idea

Forward mapping

Where so source pixel go?

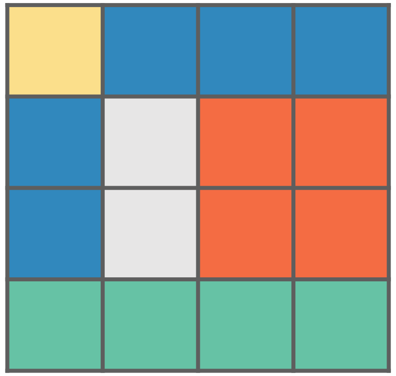
Source



Disparity T-S

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

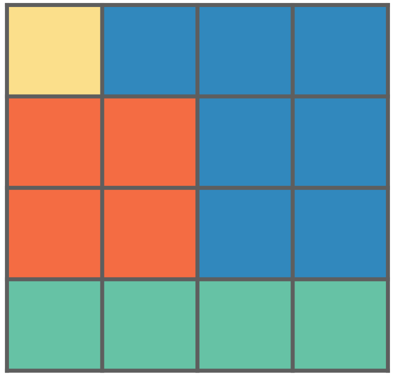
Target



Backward mapping

Where do target pixel comes from?

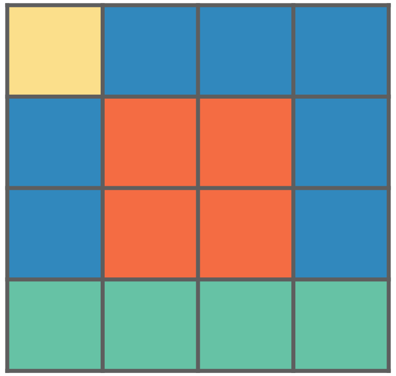
Target



Disparity S-T

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

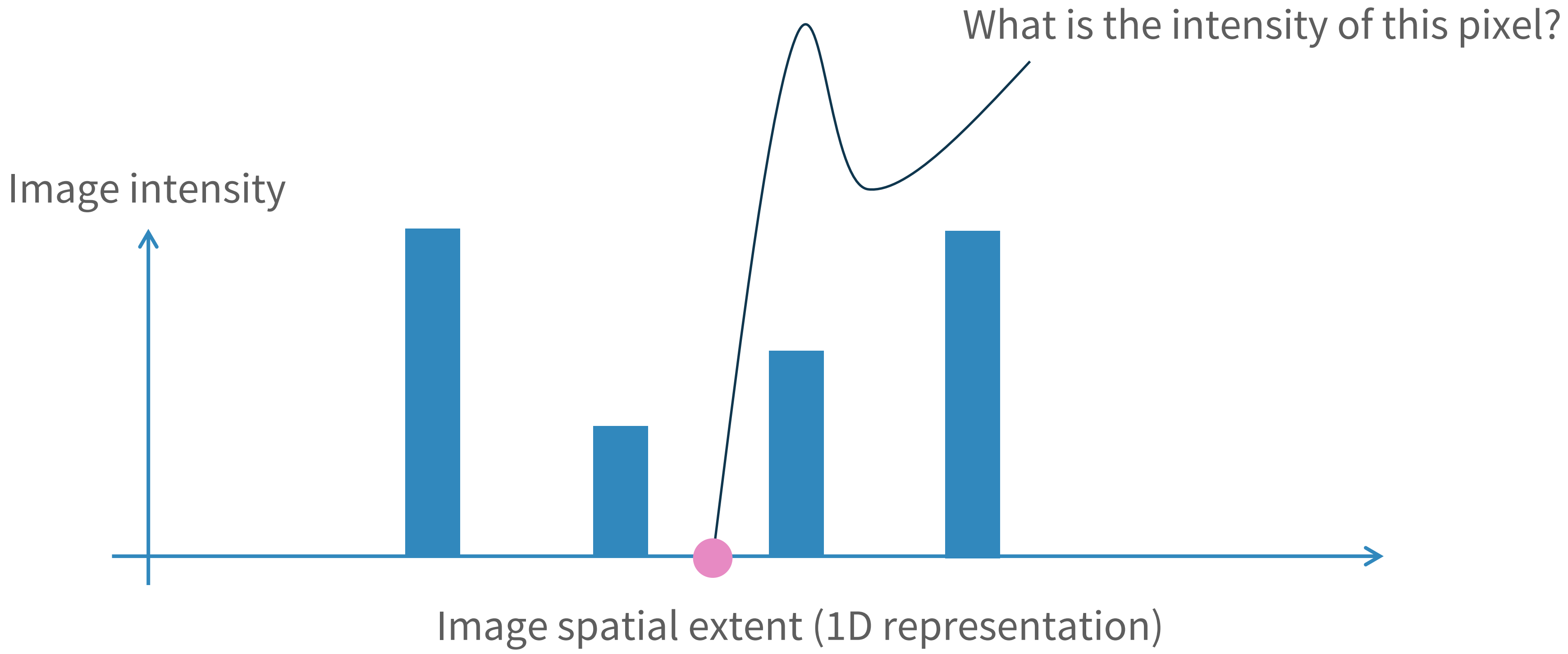
Source



Making the warping differentiable – idea

In general, disparity values can be in floating precision

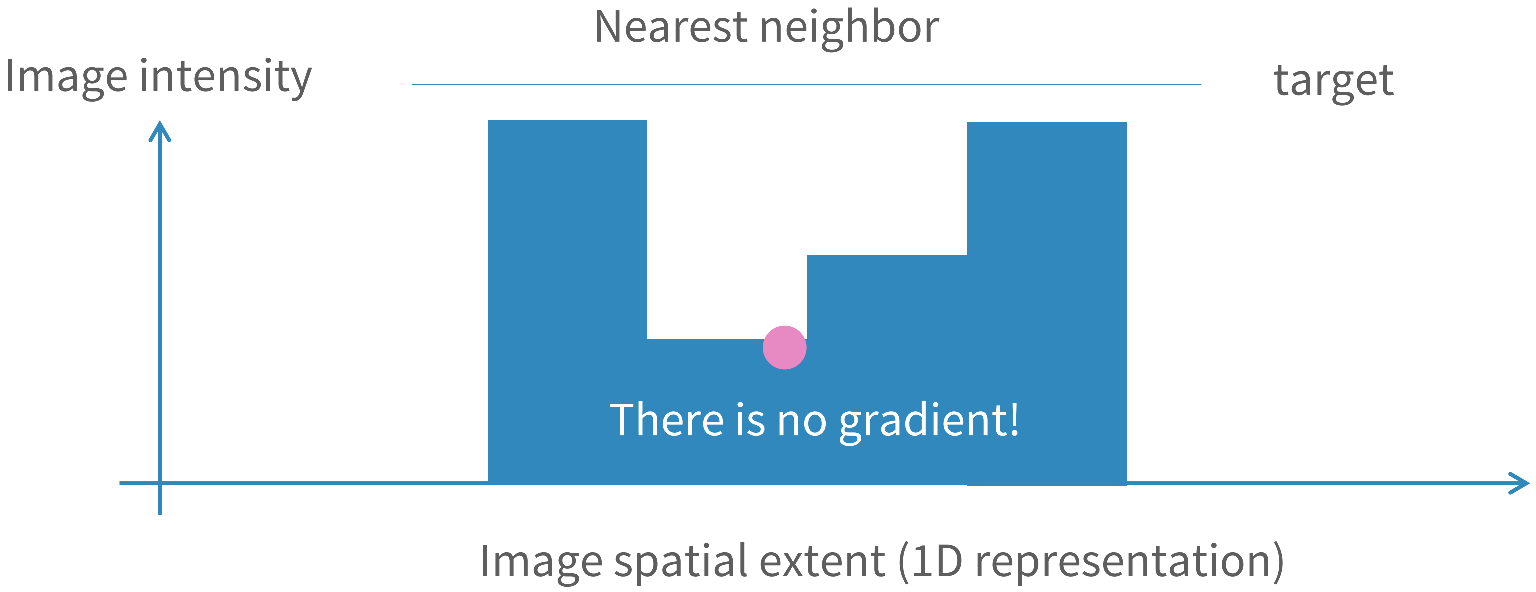
0	0	0	0
1.1	2.1	0	0
2.9	1.4	0	0
0	0	0	0



Making the warping differentiable – idea

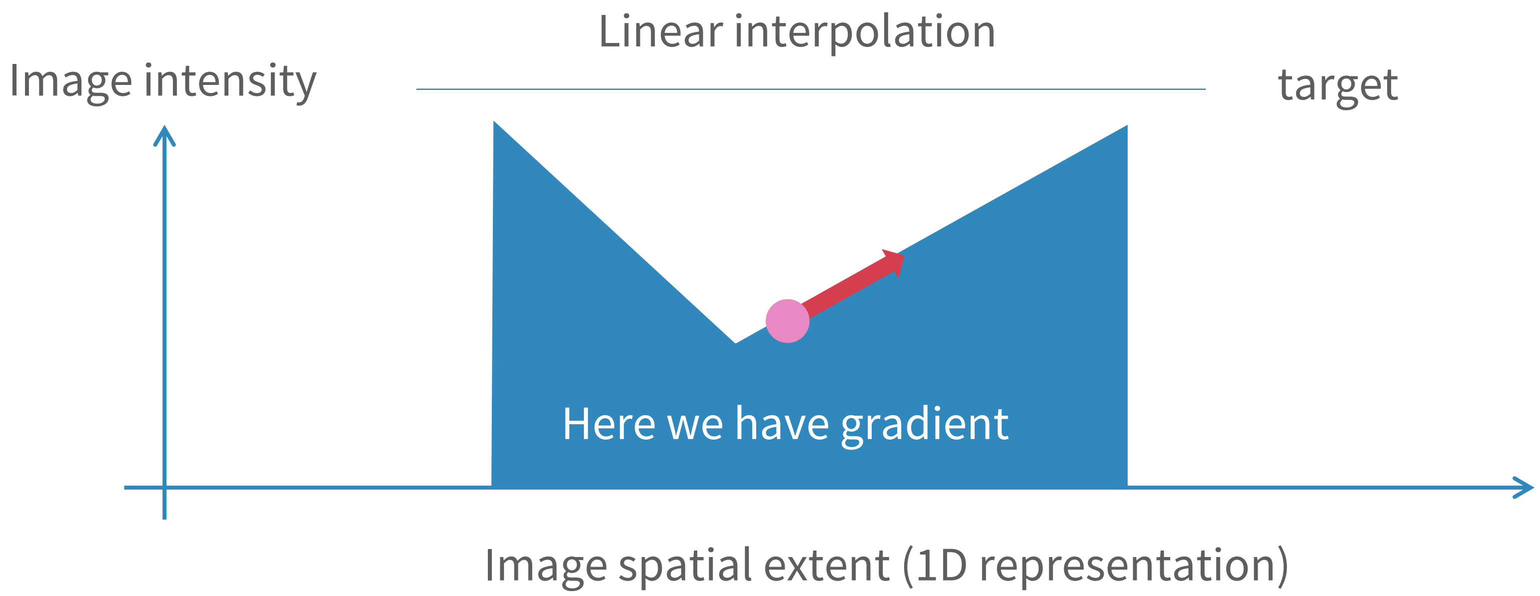
In general, disparity values can be in floating precision

0	0	0	0
1.1	2.1	0	0
2.9	1.4	0	0
0	0	0	0



Making the warping differentiable – idea

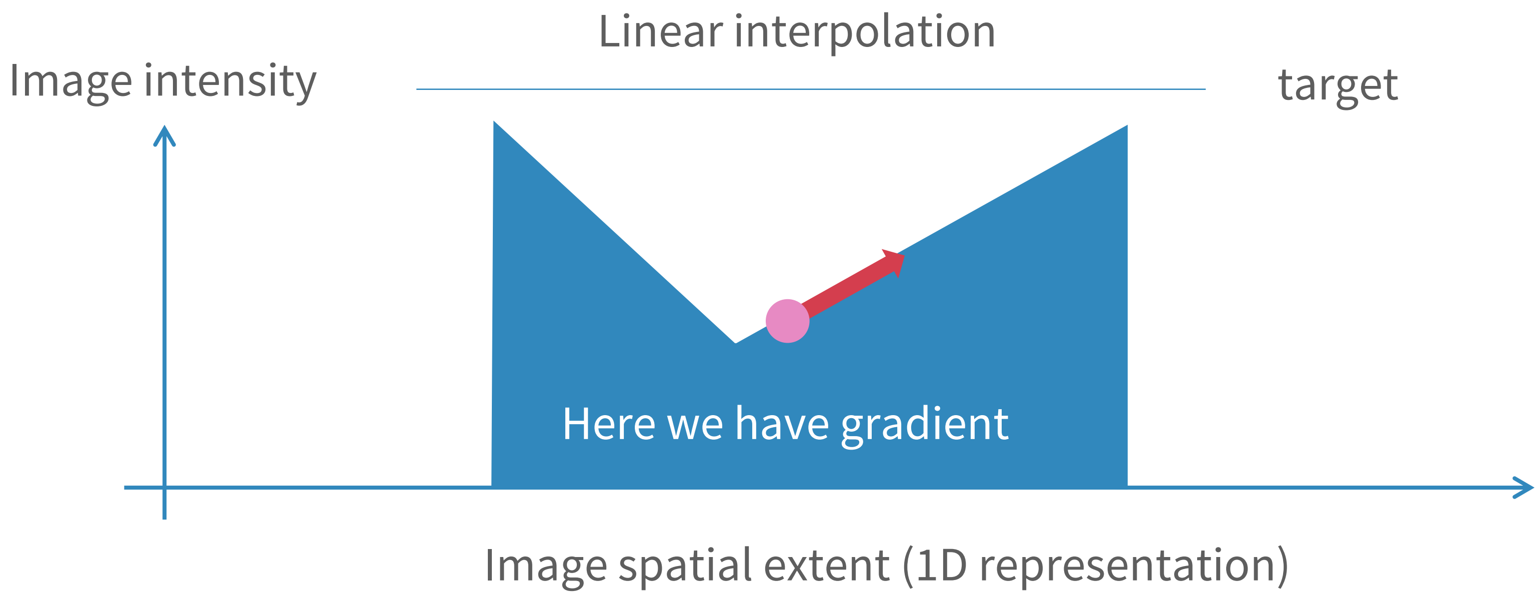
0	0	0	0
1.1	2.1	0	0
2.9	1.4	0	0
0	0	0	0

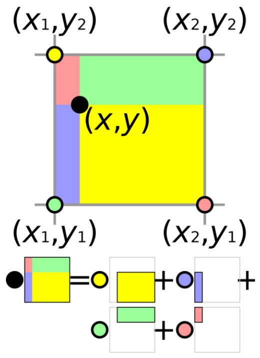


Since we are working on 2D we can use bilinear interpolation

Making the warping differentiable – idea

0	0	0	0
1.1	2.1	0	0
2.9	1.4	0	0
0	0	0	0





Since we are working on 2D spatial extent, let's use bilinear interpolation.

Jarderberg, Max et al. "Spatial transformer networks". Neruisp 2015

Image reconstruction as a supervision: Vanilla Monodepth

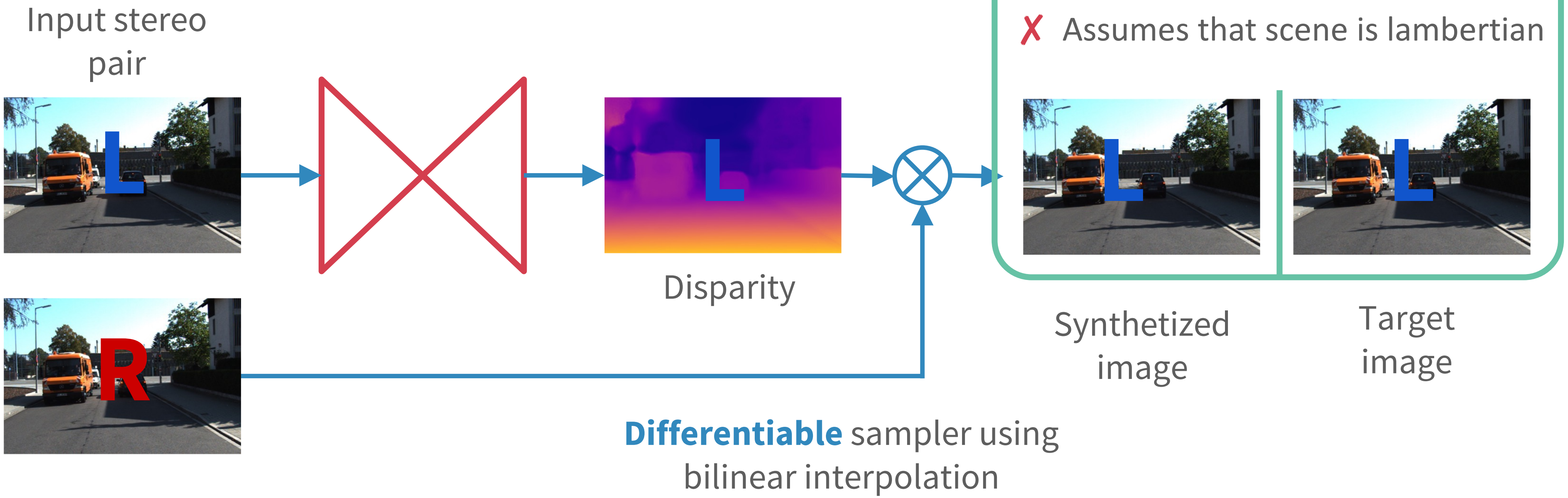
- Estimate disparity
- Use differentiable bilinear interpolation to render I_R from I_L

Structured similarity index measure

Image reconstruction Loss

$$\alpha \frac{1 - SSIM(I_R, I_L)}{2} + (1-\alpha)|I_R - I_L|$$

✗ Assumes that scene is lambertian



Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

Input

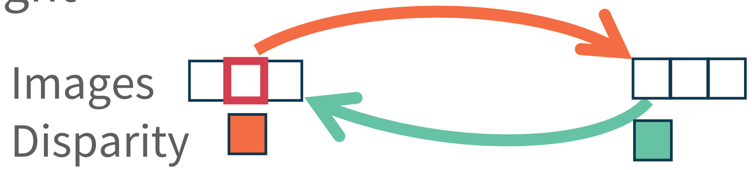
Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

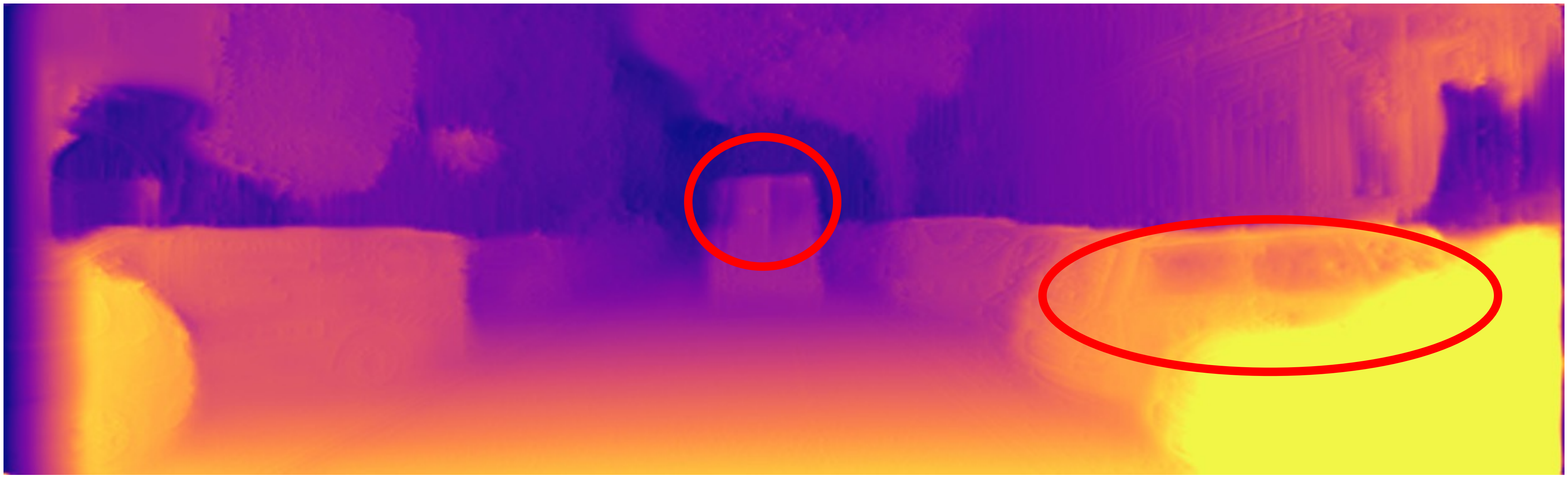
Vanilla monodepth

Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

Monodepth

By enforcing that the left-view disparity map be equal to the projected right - view disparity map



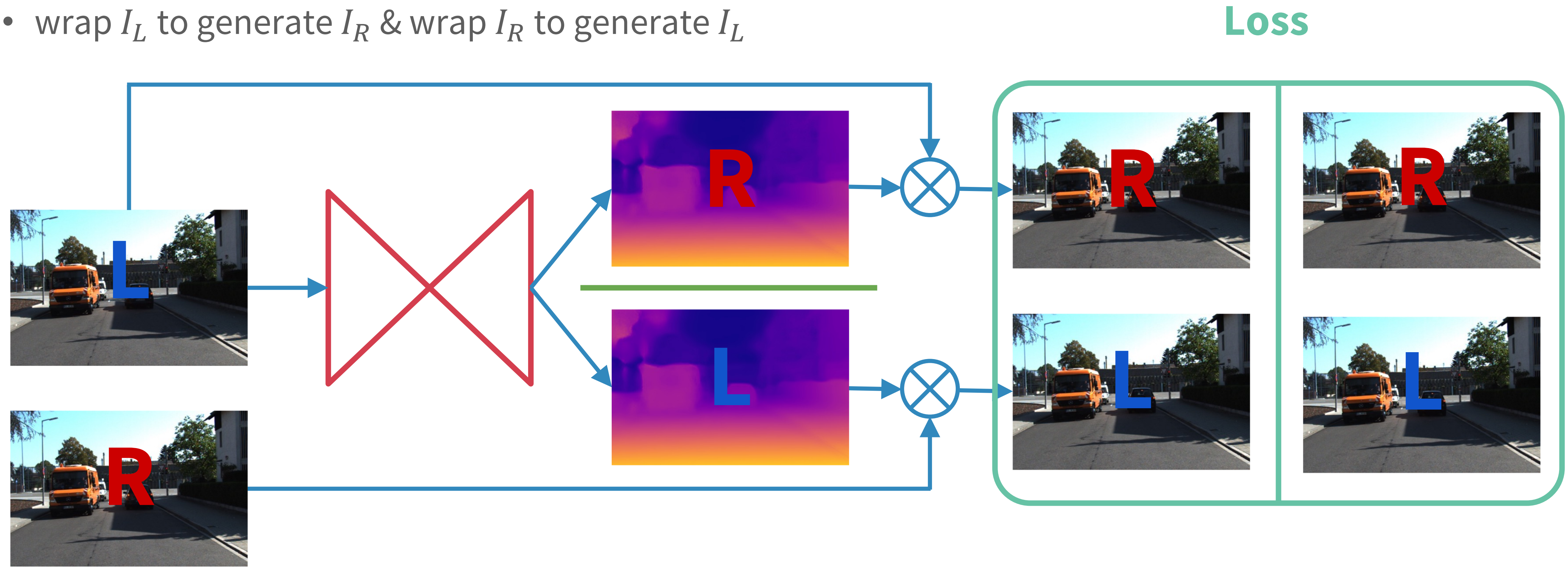


Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

Image reconstruction as a supervision: Monodepth

Operate on both images:

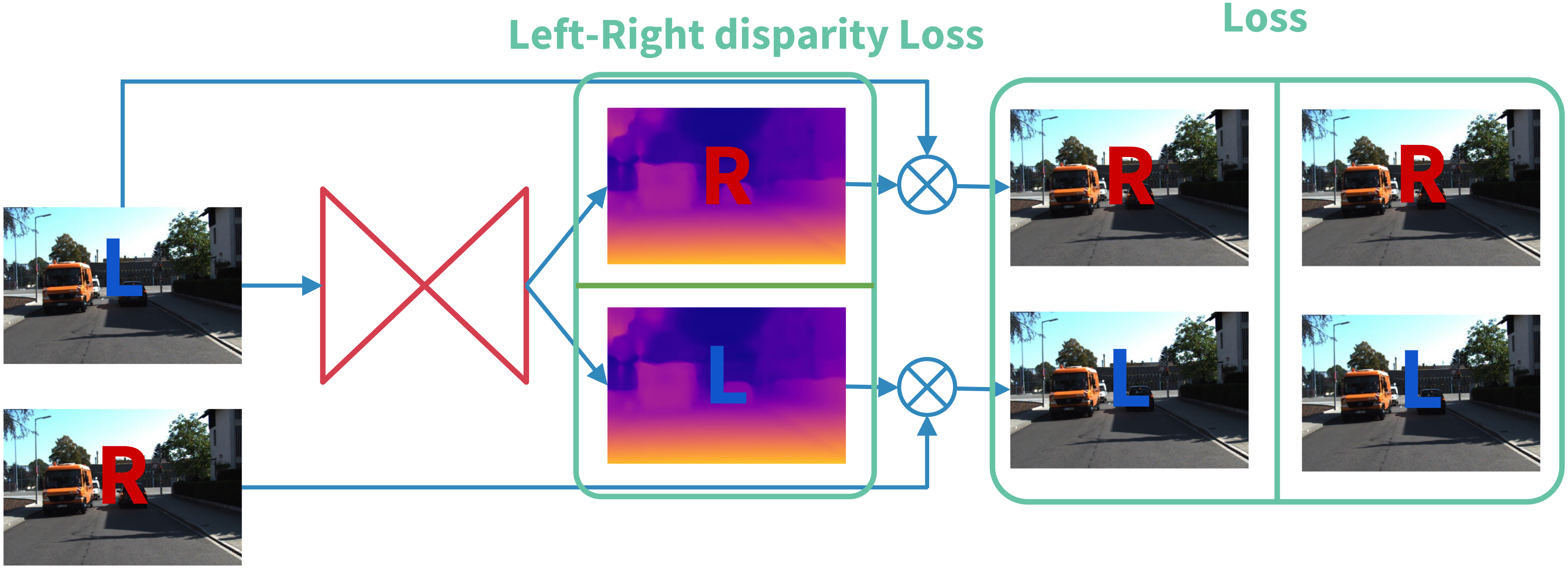
- wrap I_L to generate I_R & wrap I_R to generate I_L



Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

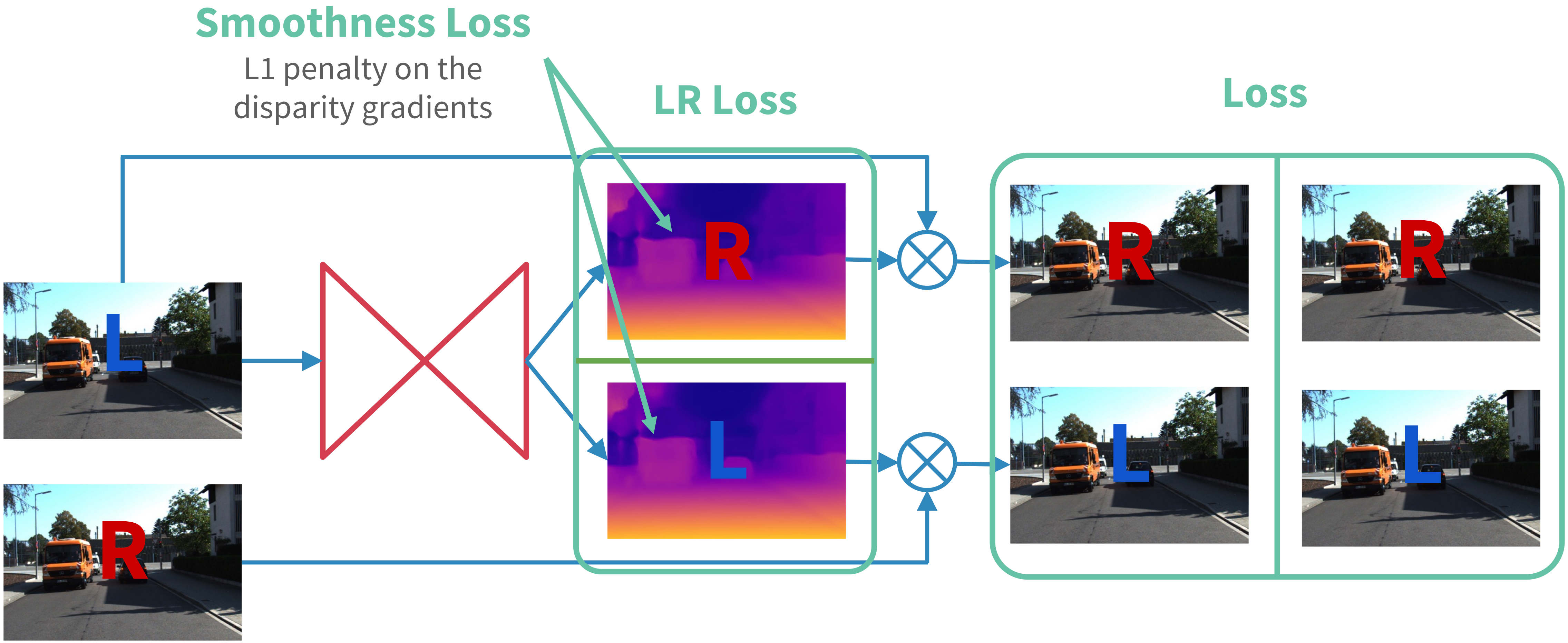
Image reconstruction as a supervision: Monodepth

Enforce consistency between left and right disparities



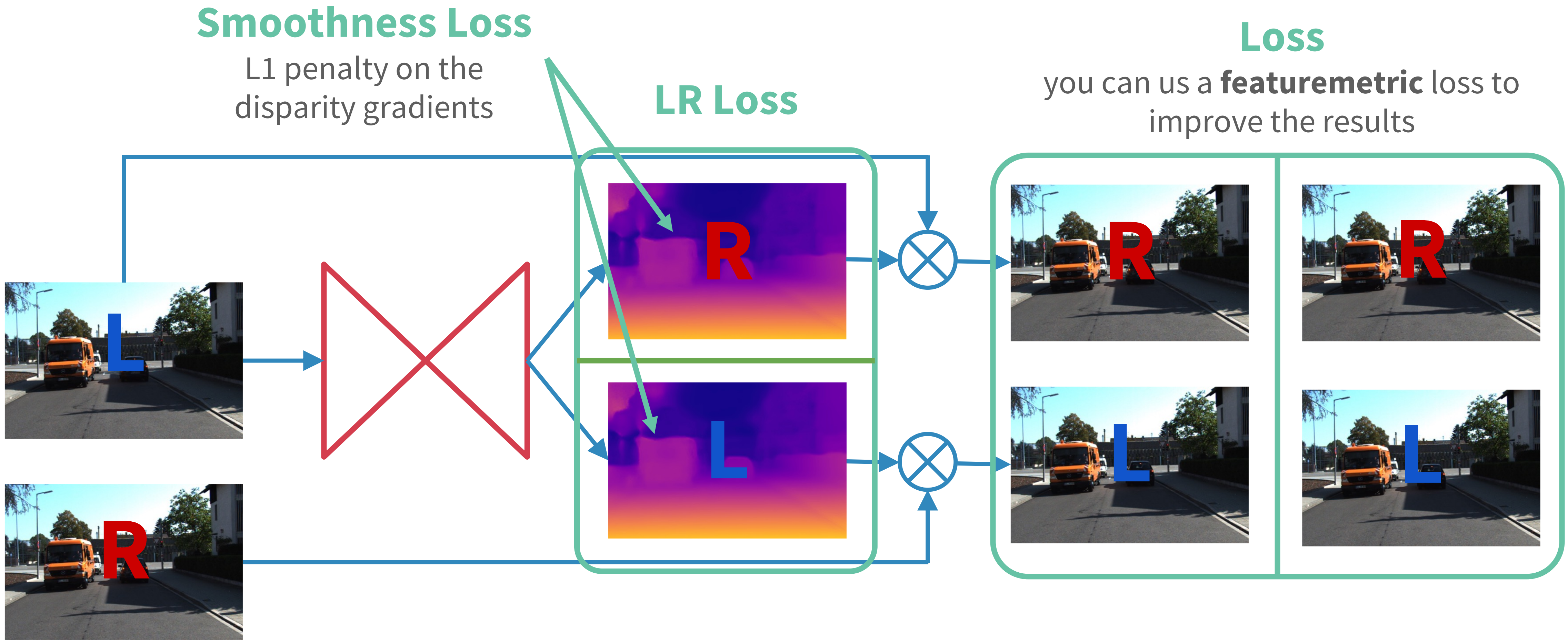
Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

Image reconstruction as a supervision: Monodepth



Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

Image reconstruction as a supervision: Monodepth

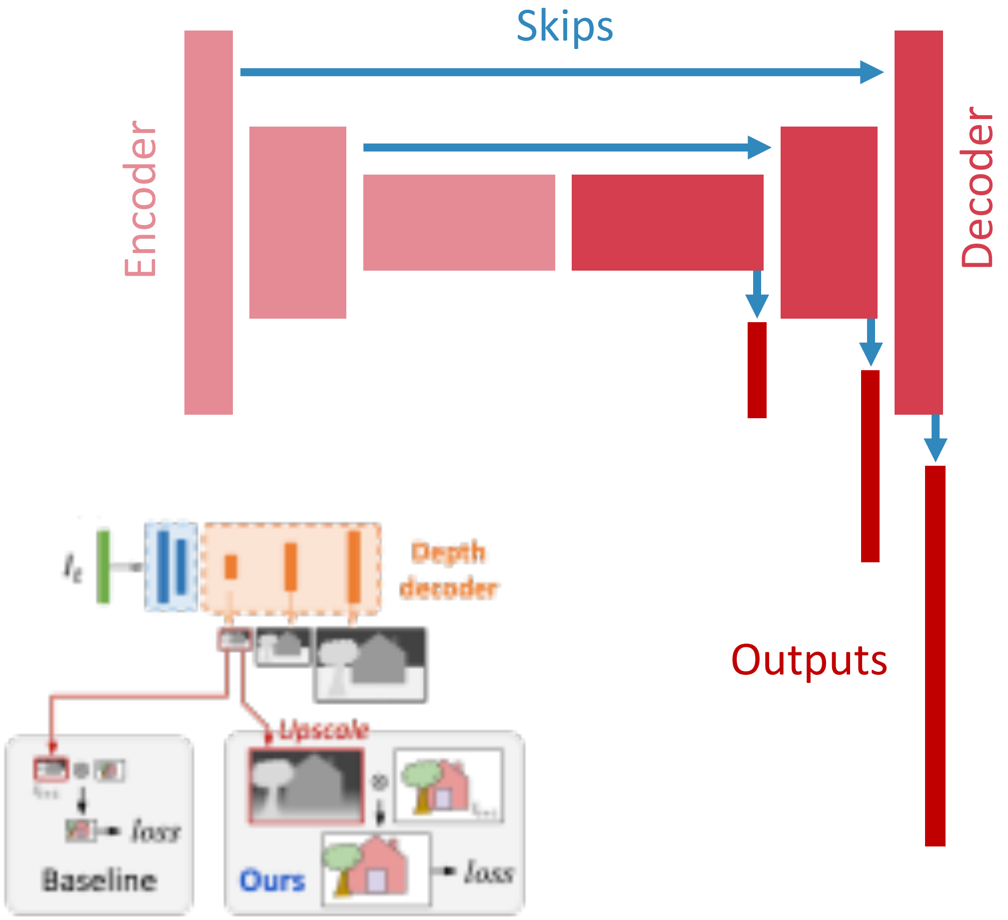


Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. 2017

Image reconstruction as a supervision: Monodepth

U-net architecture

- Fully convolutional
- Skip connections
- Fast ~30fps on a Titan X
- Multiscale generation and loss:
  - Reconstruct loss at each stage
  - Upsample the depth and then reconstruction losses at high res, reducing copying texture artefacts



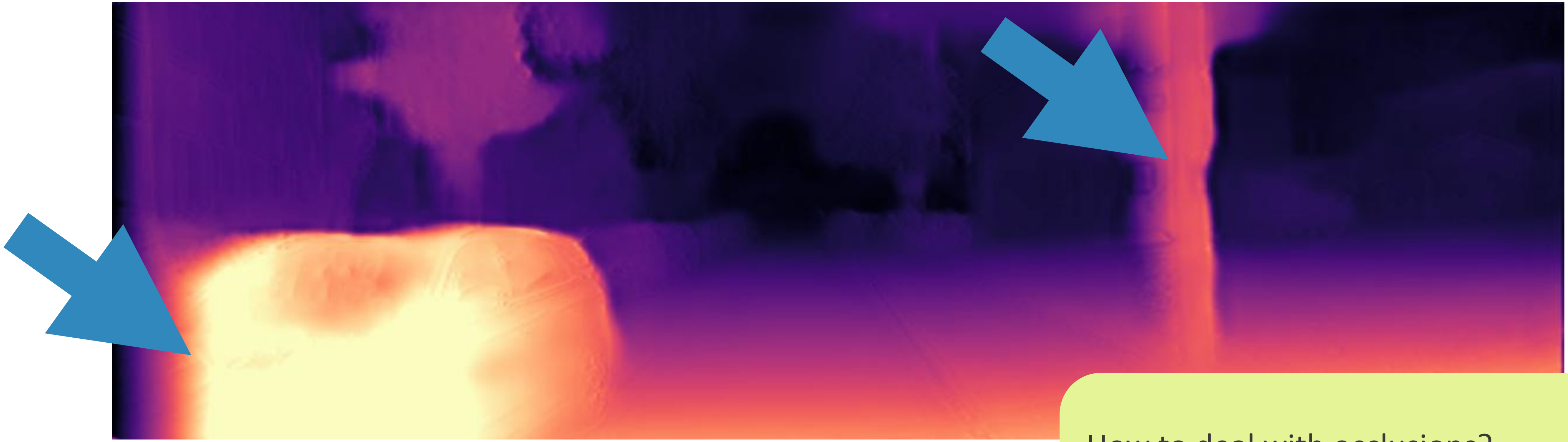
Godard, Clément, et al. "Digging into self-supervised monocular depth estimation." Proceedings of the IEEE/CVF international conference on computer vision. 2019

Image reconstruction as a supervision: Monodepth

Challenges: occlusions

Challenges: occlusions

Challenges: occlusions



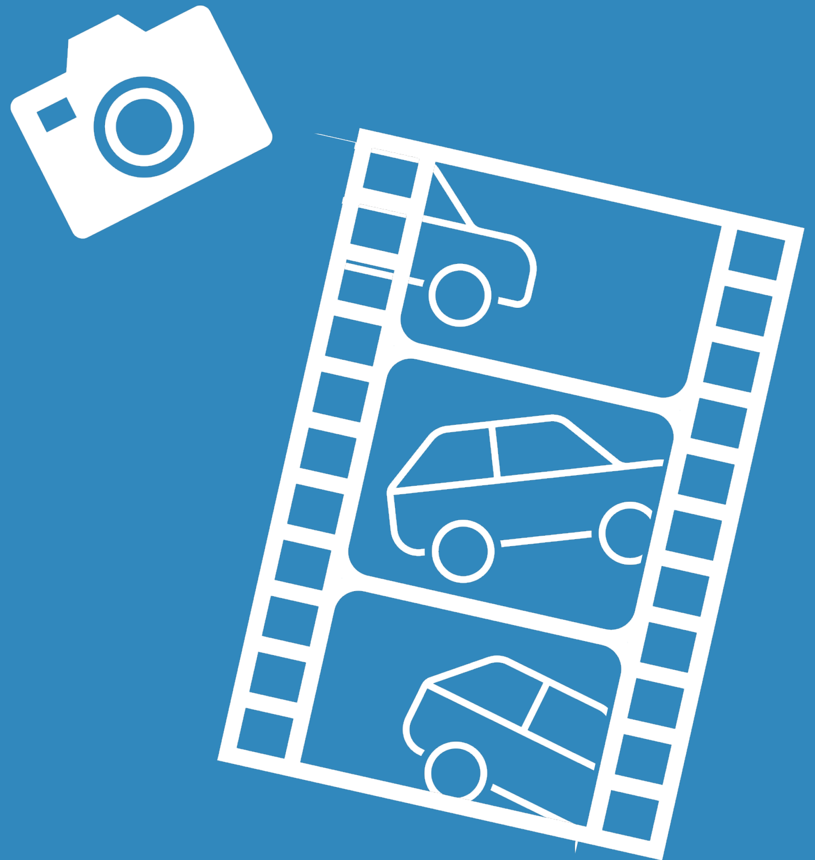
How to deal with occlusions?

- Postprocessing
- Predict occlusion mask
- Use more than two views

Recap

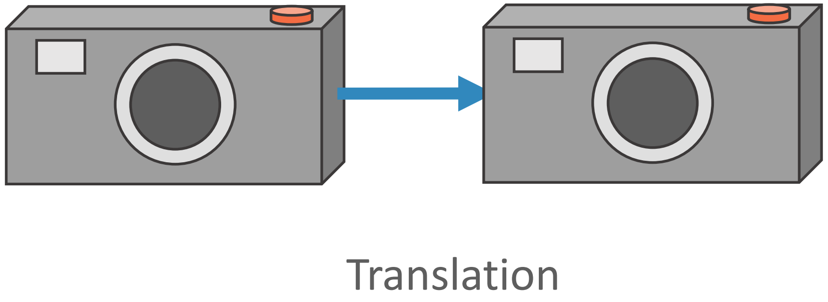
- UNet-like architectures.
- We framed the depth prediction problem as an image reconstruction one.
- Differentiable parametric image generation is easily achieved via bilinear sampling.
- Good results are achieved using multiscale, robust photometric losses, and

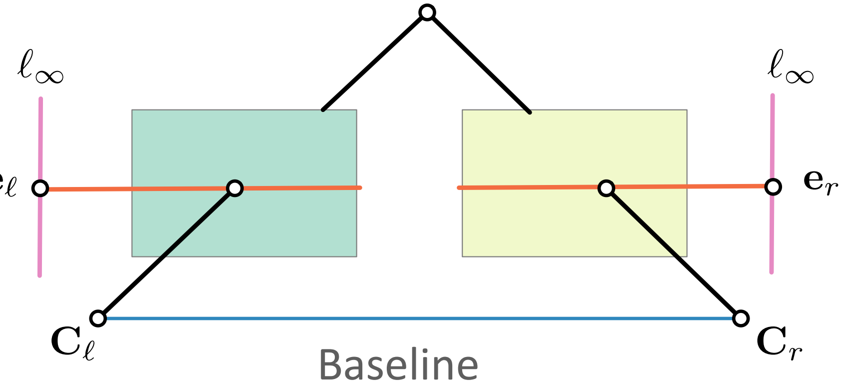
Monocular supervision



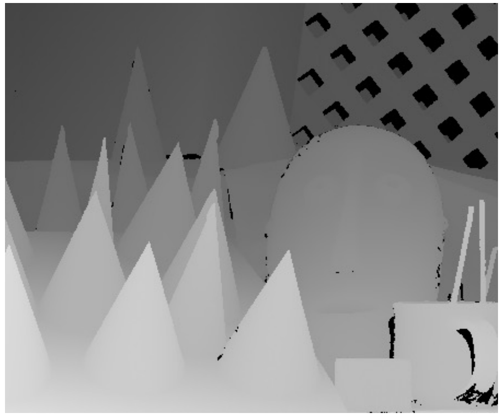
What happen when the camera is moving?

In stereo configuration, cameras that does not rotate and moves by pure translation parallel to the image plane. This results in displacement along horizontal lines

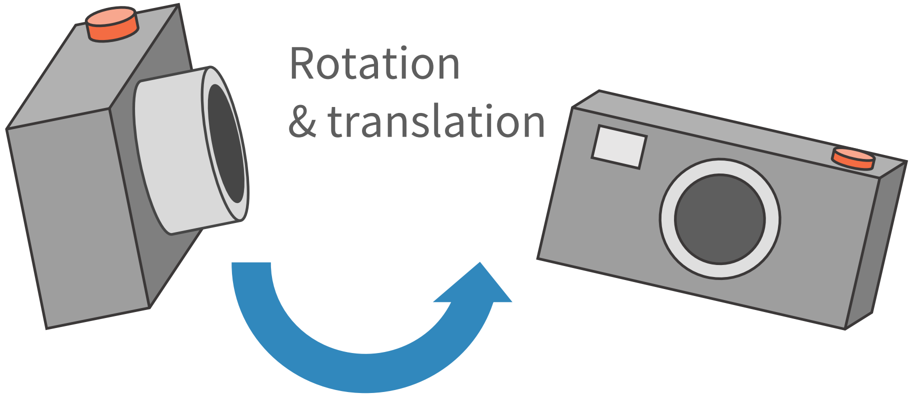




Disparity

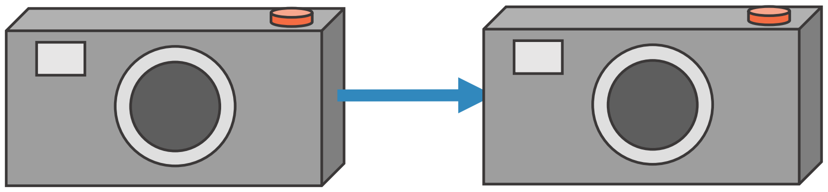


What happen if we have a more general motion?

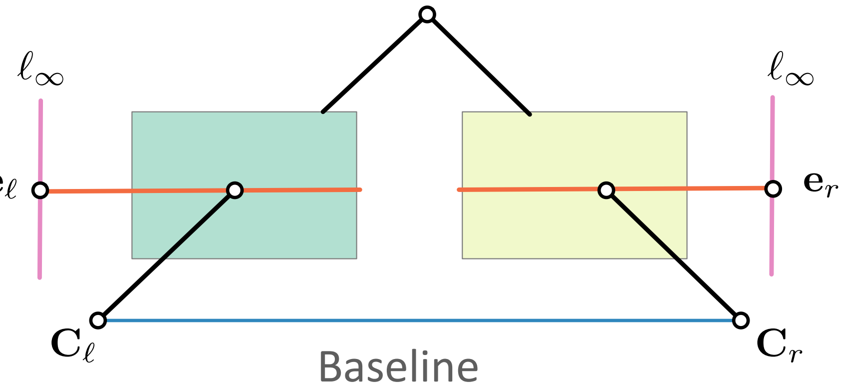


What happen when the camera is moving?

In stereo configuration, cameras that does not rotate and moves by pure translation parallel to the image plane. This results in displacement along horizontal lines

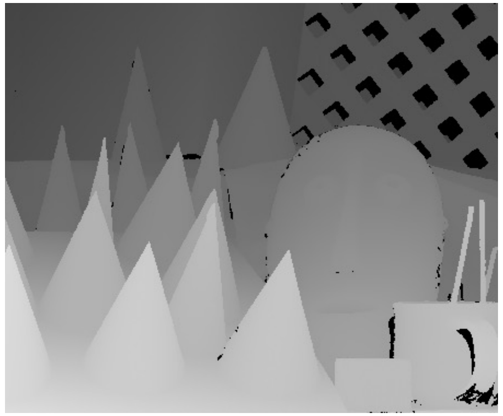


Translation

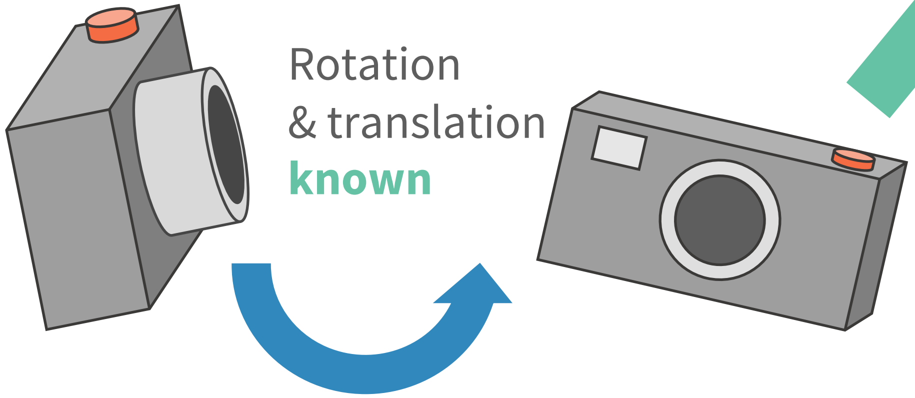


Baseline

Disparity



What happen if we have a more general motion

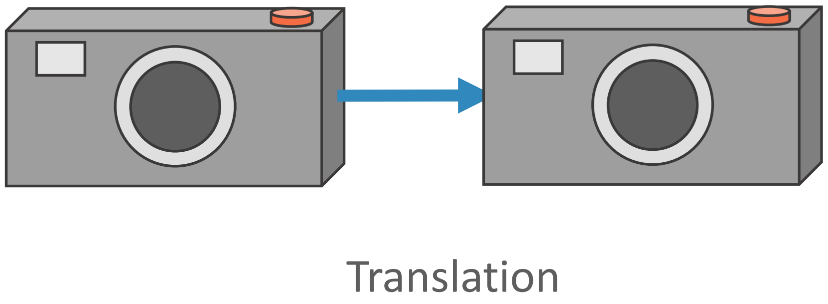


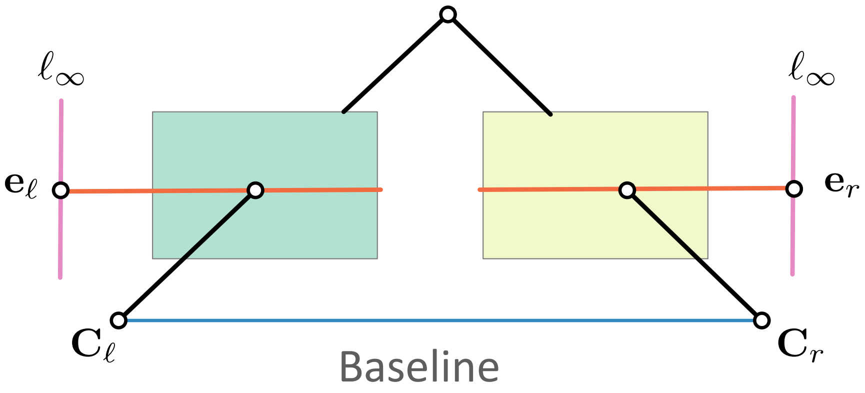
Rotation
& translation
known

If the camera are calibrated and if their relative pose is known, we can **rectify** the camera to obtain a stereo depth

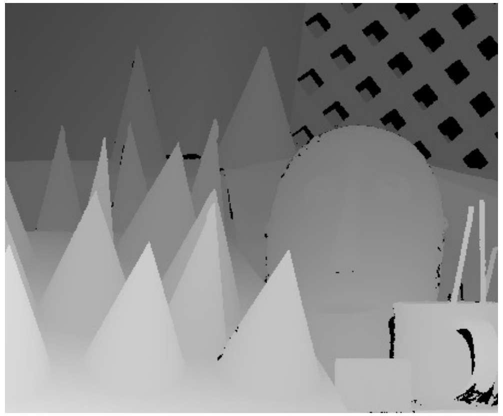
What happen when the camera is moving?

In stereo configuration, cameras that does not rotate and moves by pure translation parallel to the image plane. This results in displacement along horizontal lines

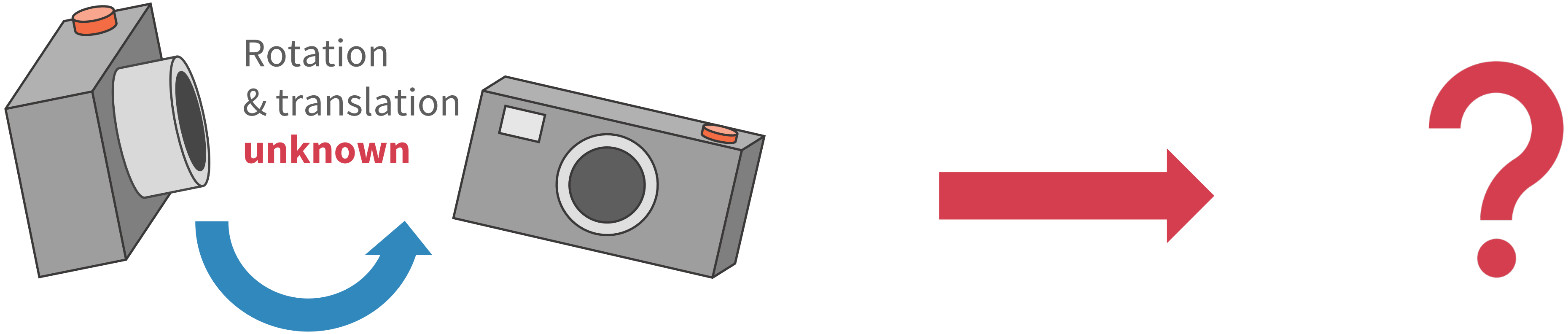




Disparity

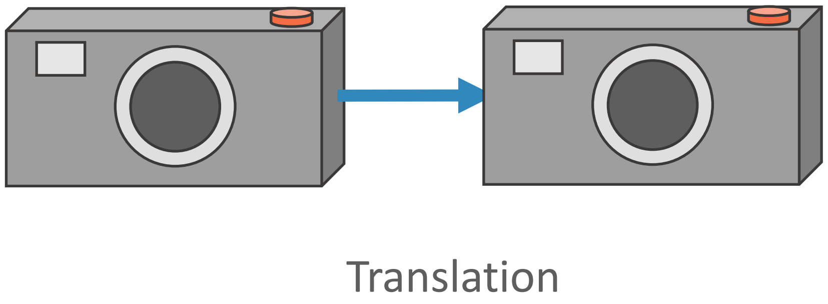


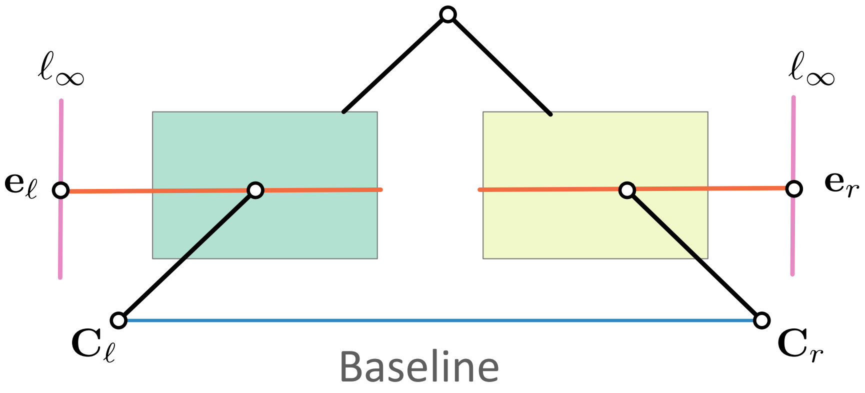
What happen if we have a more general motion?



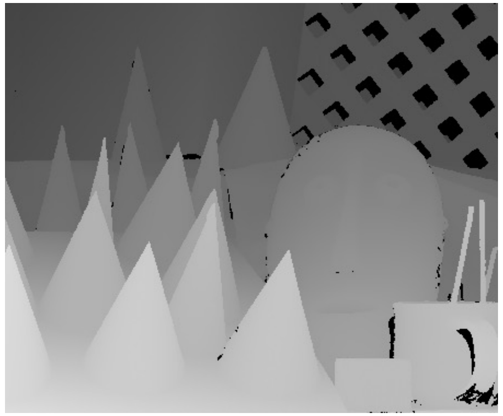
What happen when the camera is moving?

In stereo configuration, cameras that does not rotate and moves by pure translation parallel to the image plane. This results in displacement along horizontal lines

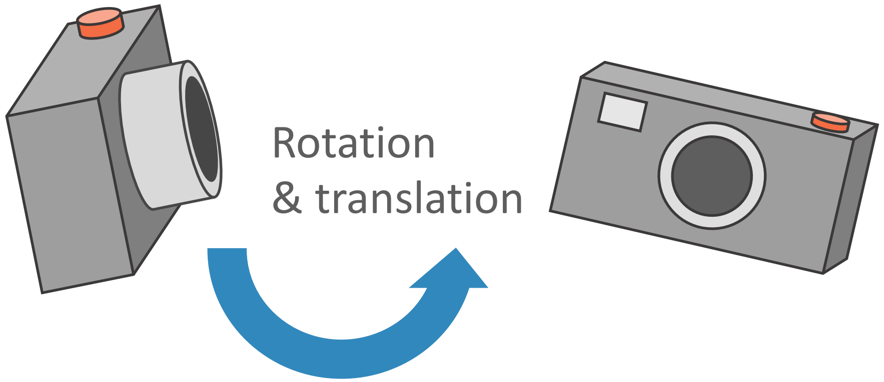


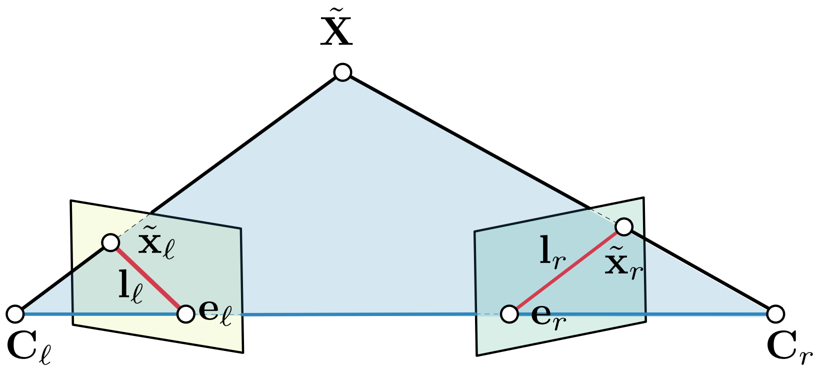


Disparity



What happen if we have a more general motion?

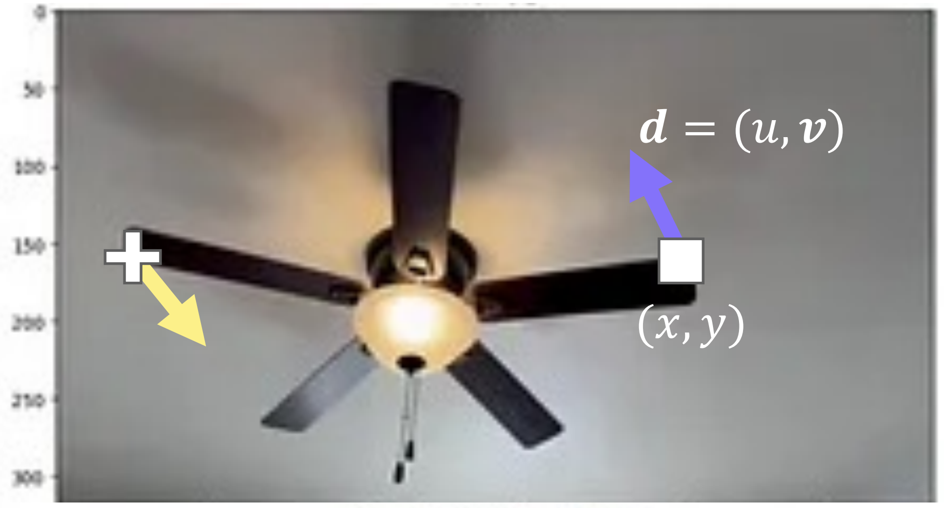




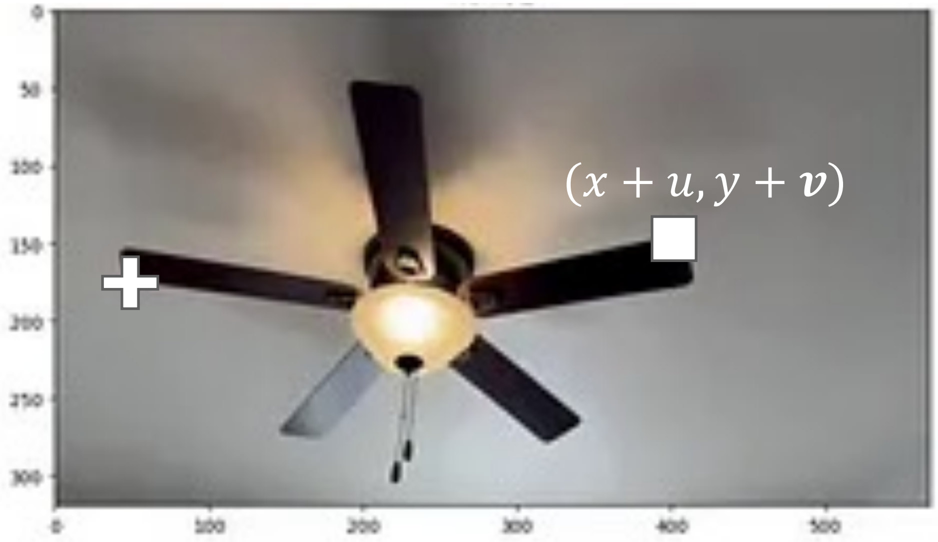
Optical flow

Optical flow

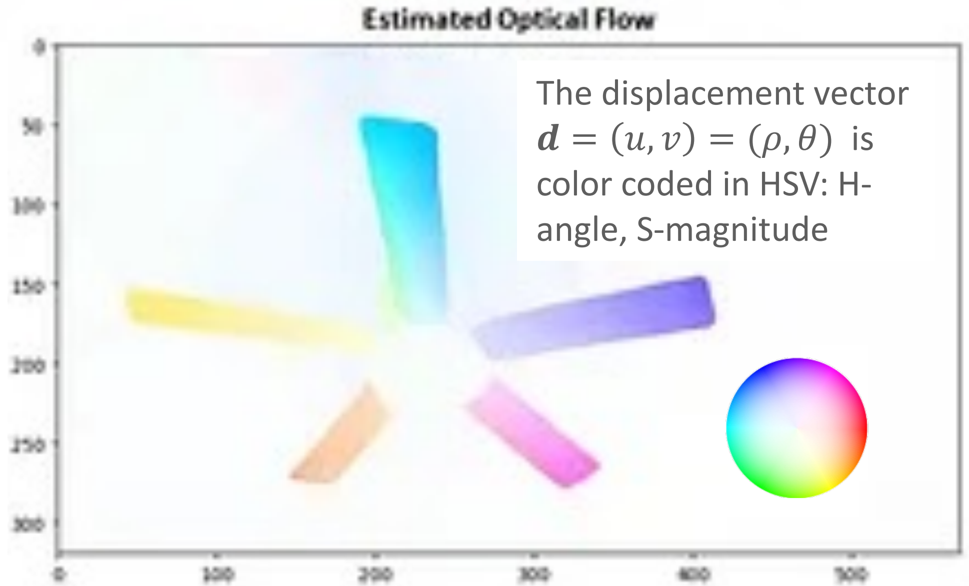
$I(:,:,t)$



$I(:,:,t+dt)$



Images from Isaac Berrios



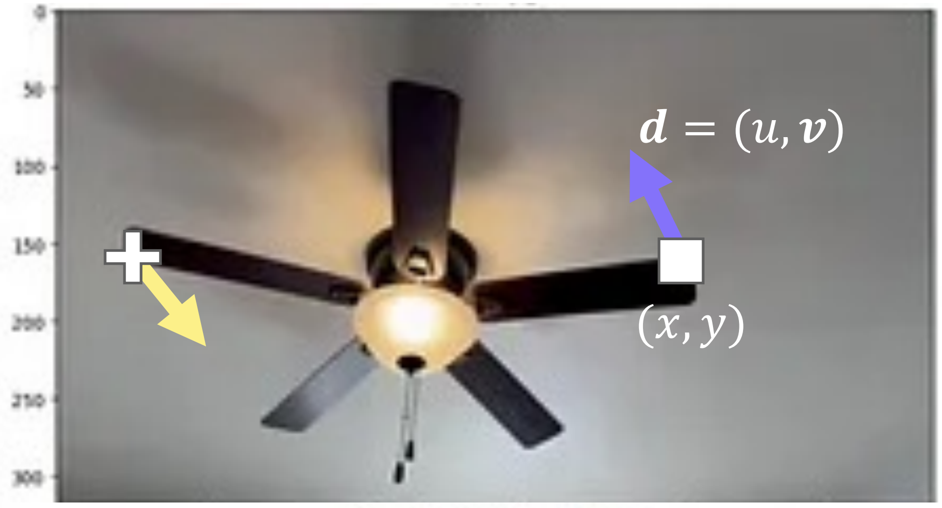
Optical flow is computed enforcing the brightness consistency assumption:
$I(x, y, t) = I(x+u, y+v, t+dt)$

This is not always satisfied due to:

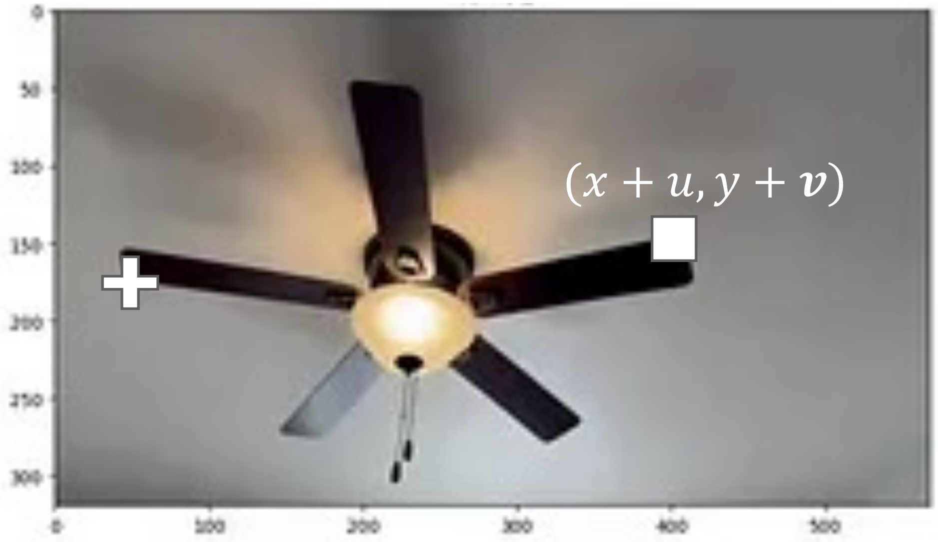
- Occlusions
- Non Lambertian objects
- Perspective effects

Optical flow

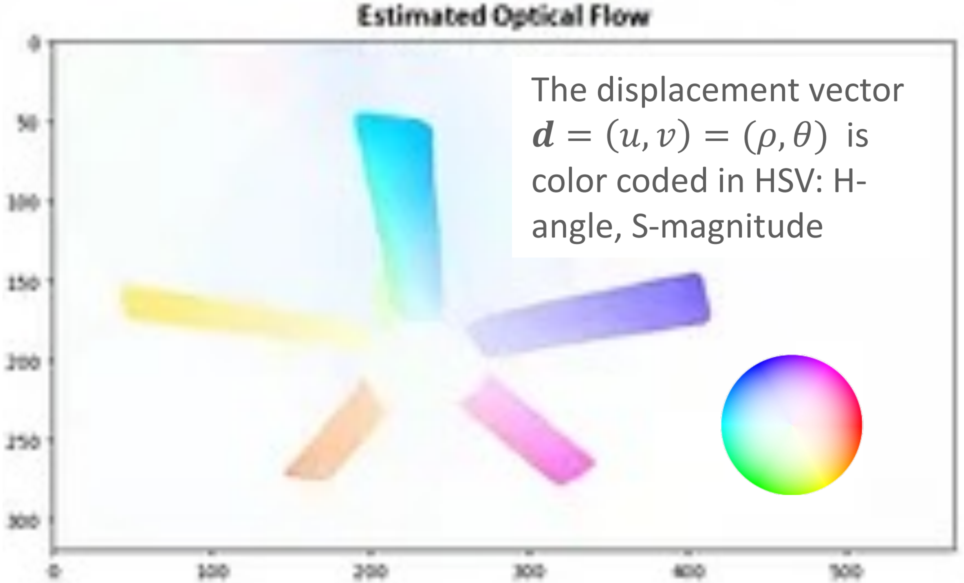
$I(:,:,t)$



$I(:,:,t+dt)$



Images from Isaac Berrios



The displacement vector $\boldsymbol{d} = (u, v) = (\rho, \theta)$ is color coded in HSV: H-angle, S-magnitude

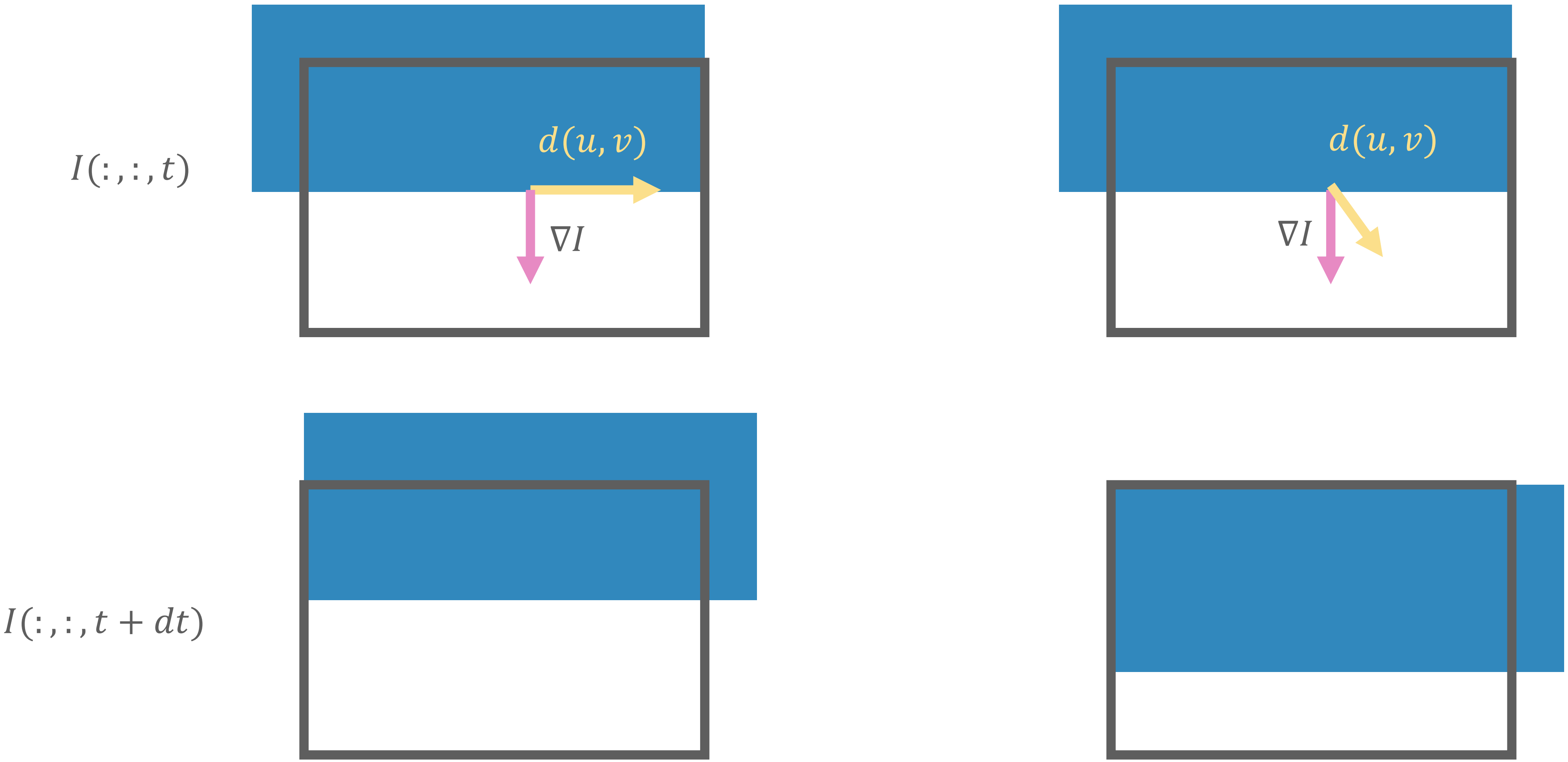
Optical flow is computed enforcing the brightness consistency assumption:
$I(x, y, t) = I(x+u, y+v, t+dt)$

First order expansion:

$$\nabla I(x, y, t)^\top \begin{bmatrix} u \\ v \end{bmatrix} + \partial_t I(x, y, t) = 0$$

This is the projection of d along the spatial gradient. The motion can be measured only along the brightness gradient (aperture problem)

Aperture problem



Computing the optical flow – sketch of the ideas

The brightness consistency provide a single equation in two unknown (u, v).

Traditional approach:
Tomasi and Kanade algorithm assume that the optical flow is constant in a small $n \times n$ window in order to accumulate enough constraints, hence they solve a overconstrained linear system

Deep learning approach:

- Supervised vanilla
  - GT data comes from 3D scenes or synthetic 3D dataset
  - Direct prediction using an Encoder-Decoder
  - Usually, multiple encoder and decoder are stacked to have a corse to fine refinement

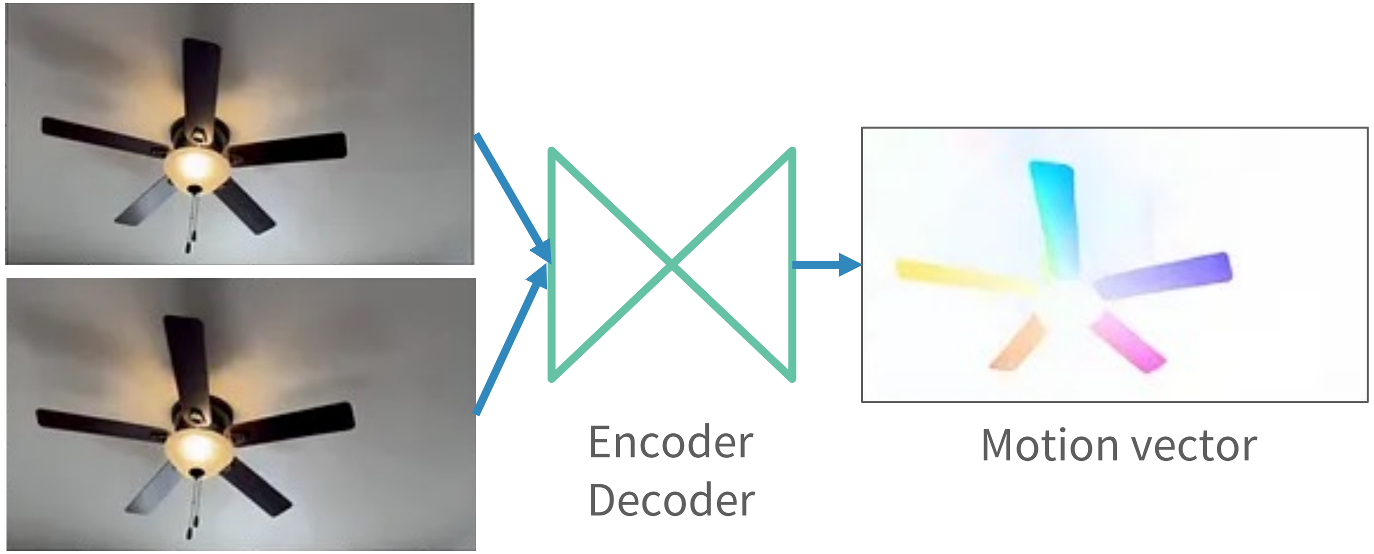


Image pairs

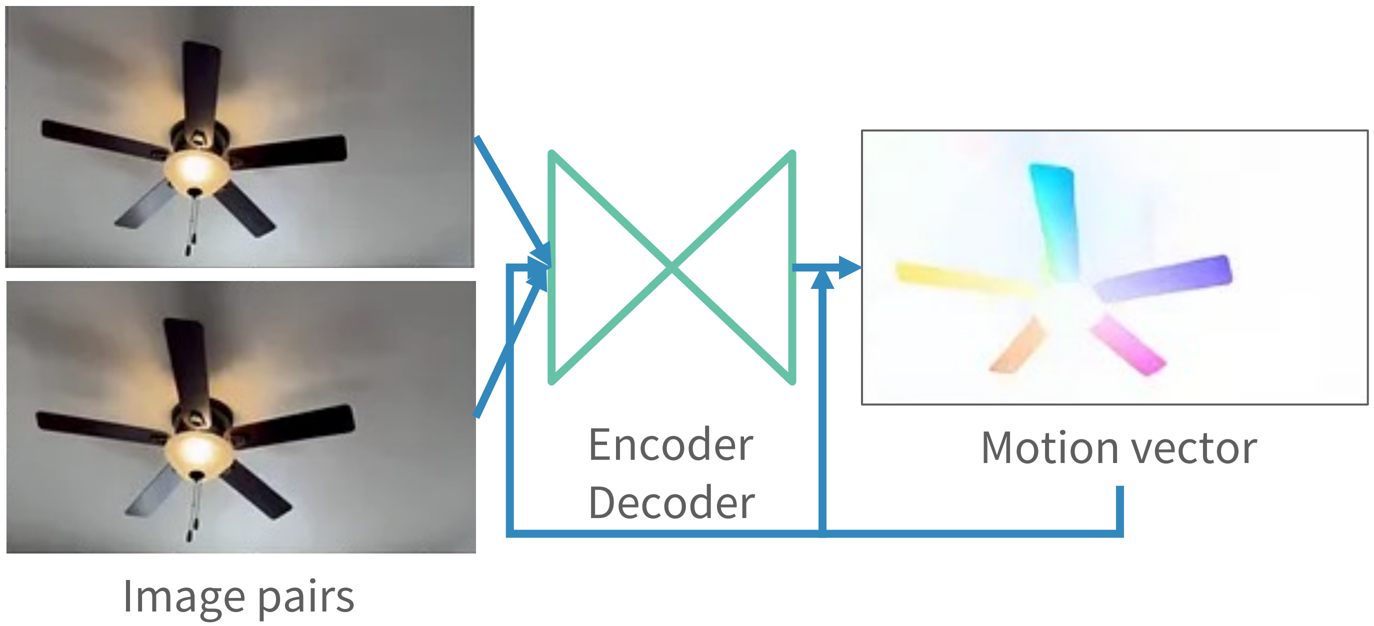
Computing the optical flow – sketch of the ideas

The brightness consistency provide a single equation in two unknown (u, v).

Traditional approach:
Tomasi and Kanade algorithm assume that the optical flow is constant in a small $n \times n$ window in order to accumulate enough constraints, hence they solve a overconstrained linear system

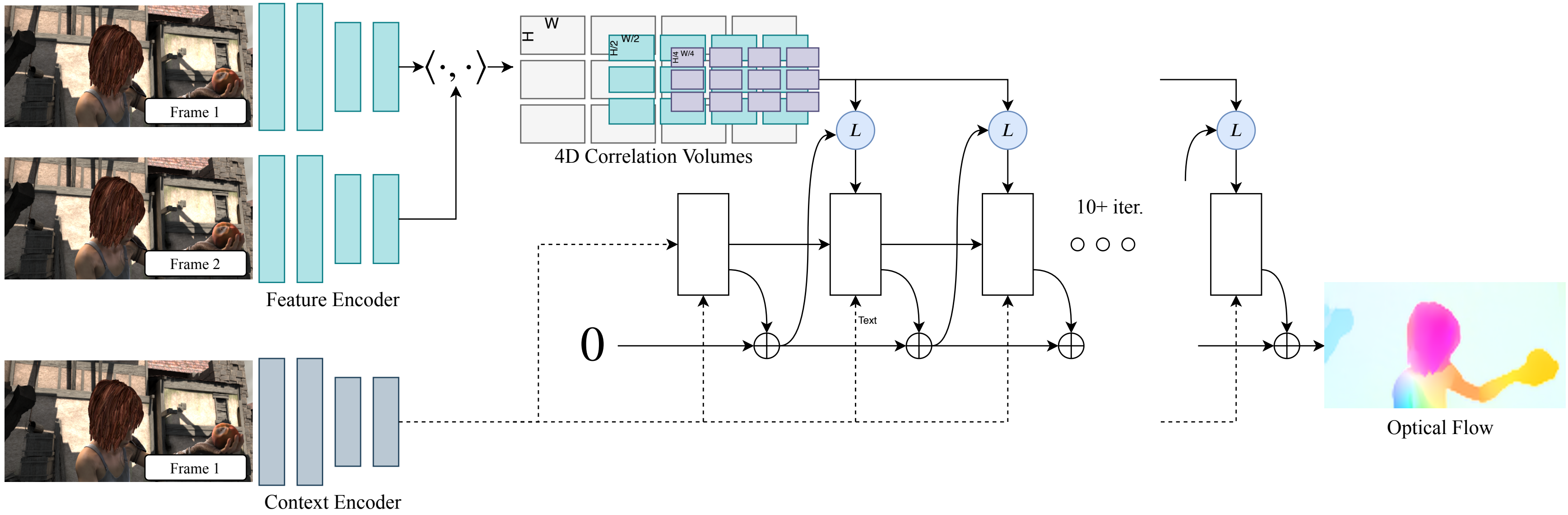
Deep learning approach:

- Supervised vanilla
- Iterative approaches
  Use a subnetwork to iteratively refine and update the residuals of the optical flow



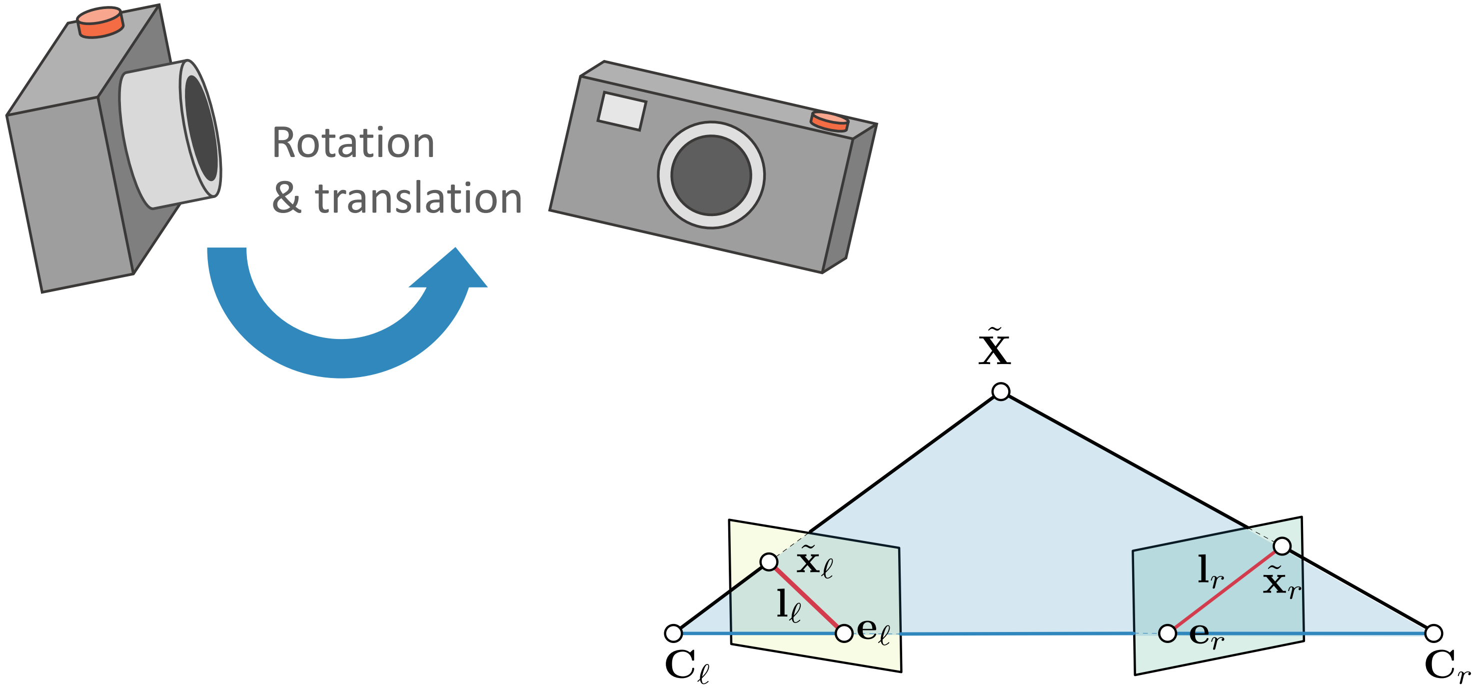
Computing the optical flow – sketch of the ideas

- Feature encoder that extract features from both input images (context encoder extract feature only from the first image)
- Context encoder to maintain high details
- A correlation layer which build a 4D correlation volume + spatial pyramid pooling (to perform correlation at different scales)
- An update operator which recurrently update the optical flow



Teed, Zachary, and Jia Deng. "Raft: Recurrent all-pairs field transforms for optical flow.” ECCV 2020

Let's go back to our problem



Optical flow

Geometric interlude

Epipolar geometry

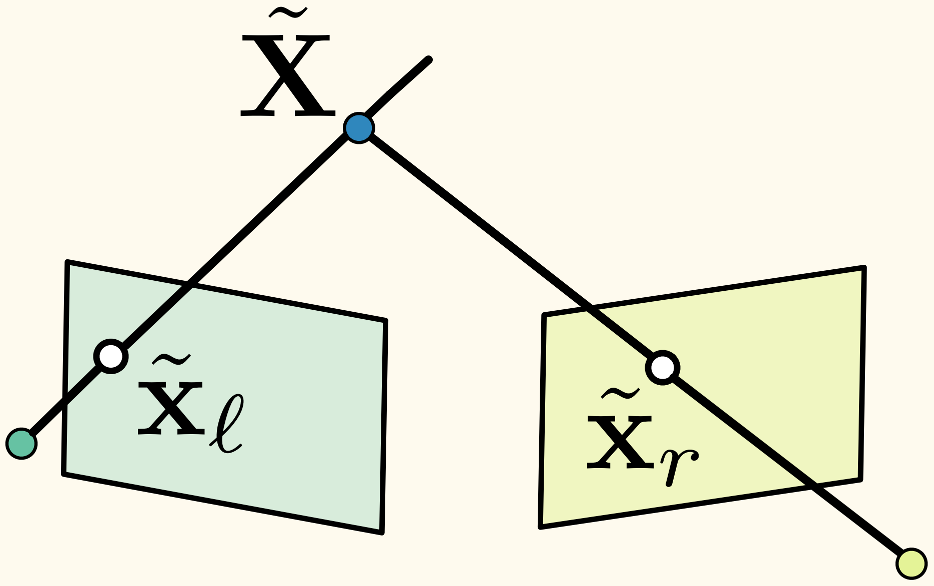
Epipolar geometry

A unoclluded 3D point $\widetilde{\boldsymbol{X}} = (X, Y, Z, 1)^T$ is projected to the left and right image as $\widetilde{\boldsymbol{x}}_\ell = (u_\ell, v_\ell, 1)^T$ and $\widetilde{\boldsymbol{x}}_r = (u_r, v_r, 1)^T$, by

$$\zeta_\ell \widetilde{\boldsymbol{x}}_\ell = P_\ell \widetilde{\boldsymbol{X}}$$
$$\zeta_r \widetilde{\boldsymbol{x}}_r = P_r \widetilde{\boldsymbol{X}}$$

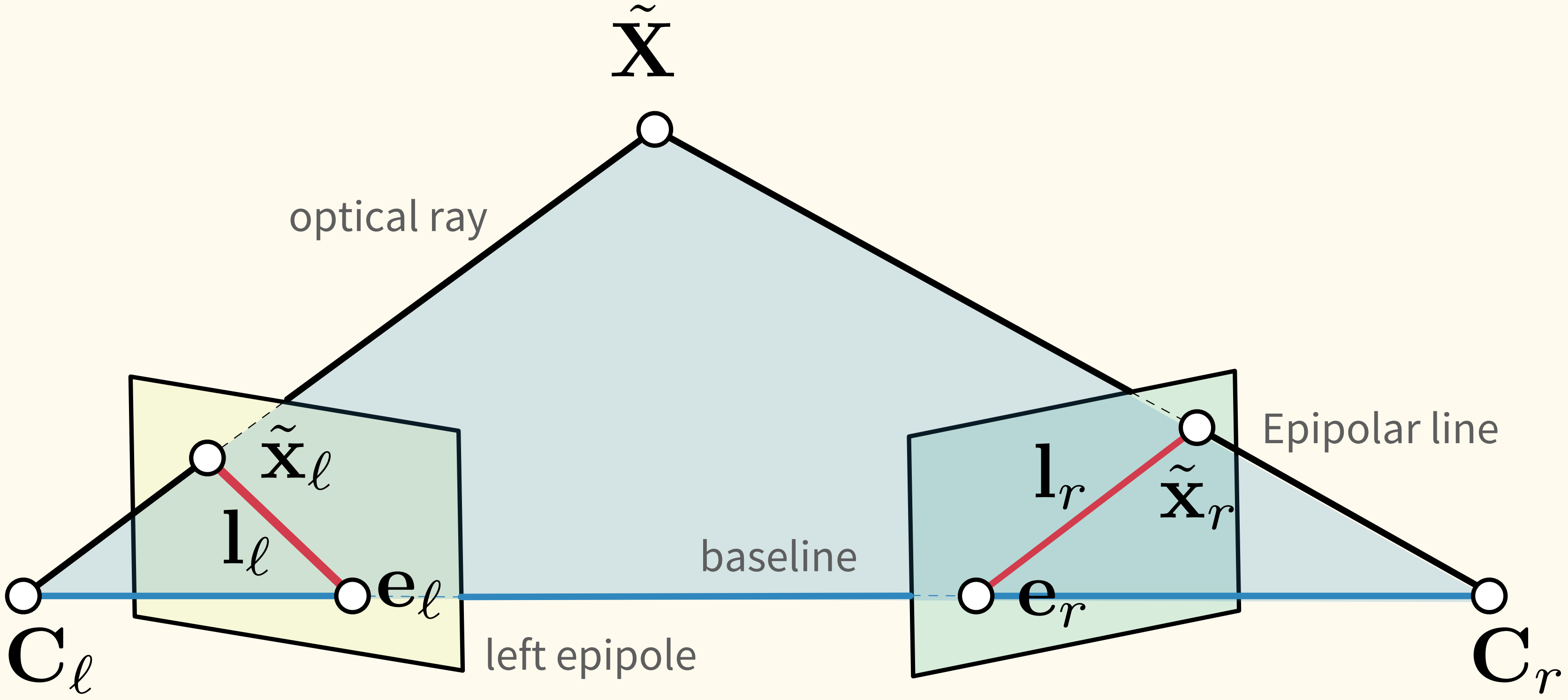
where P_ℓ and P_r denotes the left and the right camera matrix respectively.

Points $\widetilde{\boldsymbol{x}}_\ell \leftrightarrow \widetilde{\boldsymbol{x}}_r$ are called corresponding points.



Epipolar geometry

- Baseline: the line passing through the camera centers
- Epipolar plane: the plane containg $\boldsymbol{X}$ and the baseline
- Epipoles: the intersection points $\boldsymbol{e}_\ell$ and $\boldsymbol{e}_r$ of the image planes and the baseline
- Epipolar lines: lines l_ℓ, l_r intersection of the epipolar plane and the image plane



Epipolar geometry

Given a point $\widetilde{\boldsymbol{x}}_\ell$, one can determine the epipolar line in the right image on which the corresponding point $\widetilde{\boldsymbol{x}}_r$, must lie.

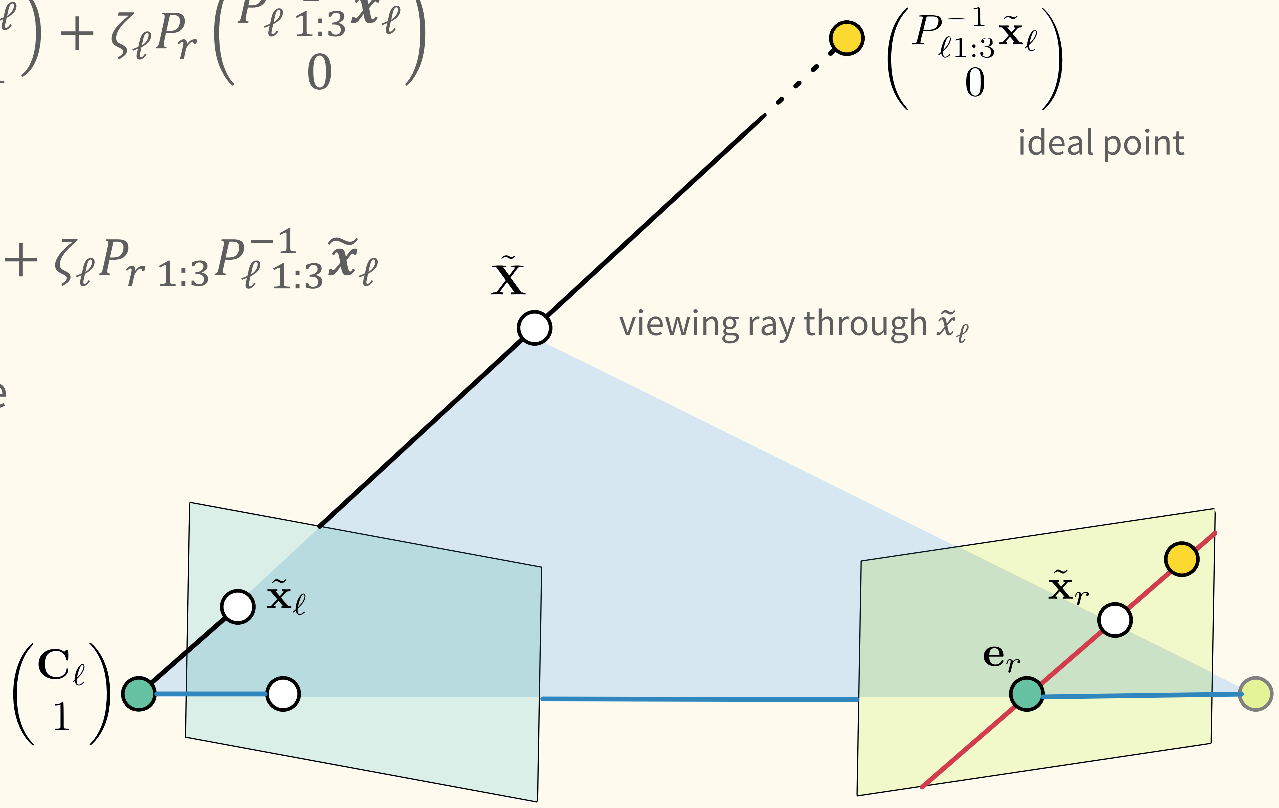
The equation of the epipolar line can be derived geometrically, as the projection of the optical ray of $\widetilde{\boldsymbol{x}}_\ell$ onto the right image plane:

$$\zeta_r \widetilde{\boldsymbol{x}}_r = P_r \begin{pmatrix} C_\ell \\ 1 \end{pmatrix} + \zeta_\ell P_r \begin{pmatrix} P_{\ell\,1:3}^{-1} \widetilde{\boldsymbol{x}}_\ell \\ 0 \end{pmatrix}$$

$$\zeta_r \widetilde{\boldsymbol{x}}_r = \boldsymbol{e}_r + \zeta_\ell P_{r\,1:3} P_{\ell\,1:3}^{-1} \widetilde{\boldsymbol{x}}_\ell$$

This is the equation of a line $\boldsymbol{l}_r$ through the right epipole and the image point $P_{r\,1:3} P_{\ell\,1:3}^{-1} \widetilde{\boldsymbol{x}}_\ell$ which represents the projection onto the right image plane of the point at infinity of the optical ray.

The left epipolar line can be derived similarly.



Epipolar geometry

The line $\boldsymbol{l}_r$ joining $\boldsymbol{e}_r$ and $P_{r\,1:3}P_{\ell\,1:3}^{-1}\widetilde{\boldsymbol{x}}_\ell$ can be represented in terms of the cross product

$$\boldsymbol{l}_r \equiv \boldsymbol{e}_r \times\ P_{r\,1:3}P_{\ell\,1:3}^{-1}\widetilde{\boldsymbol{x}}_\ell$$

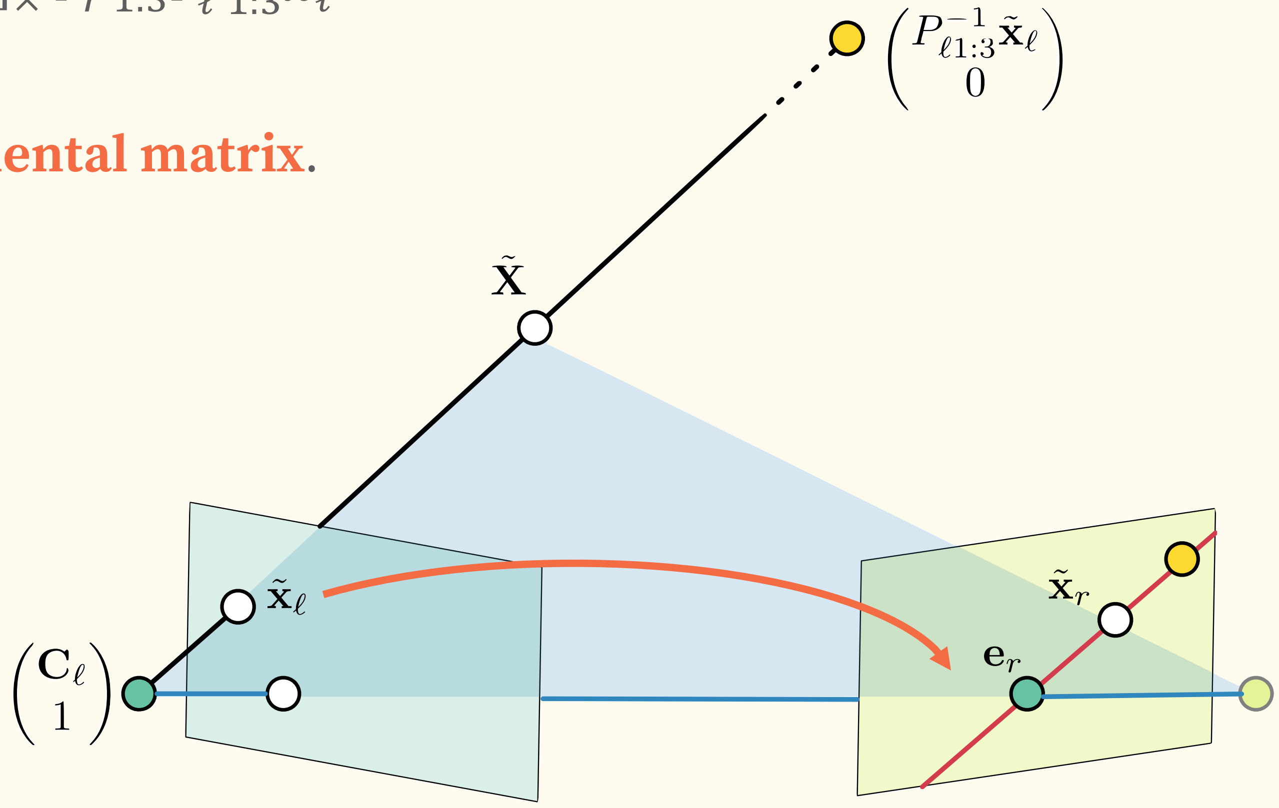
$$\boldsymbol{l}_r \equiv [\boldsymbol{e}_r]_\times\, P_{r\,1:3}P_{\ell\,1:3}^{-1}\widetilde{\boldsymbol{x}}_\ell$$

The matrix $F = [\boldsymbol{e}_r]_\times\, P_{r\,1:3}P_{\ell\,1:3}^{-1}$ is called **fundamental matrix**.

The epipolar line for a point $\boldsymbol{x}$ is $\boldsymbol{l}_r = F\boldsymbol{x}$.
The incidence relation $\widetilde{\boldsymbol{x}}_r \in \boldsymbol{l}_r$ implies $\widetilde{\boldsymbol{x}}_r^T\boldsymbol{l}_r = 0$
and corresponding points have to satisfy

$$\boxed{\widetilde{\boldsymbol{x}}_r^T F \widetilde{\boldsymbol{x}}_l = 0}$$

a point-line relation between two views based only on camera matrices



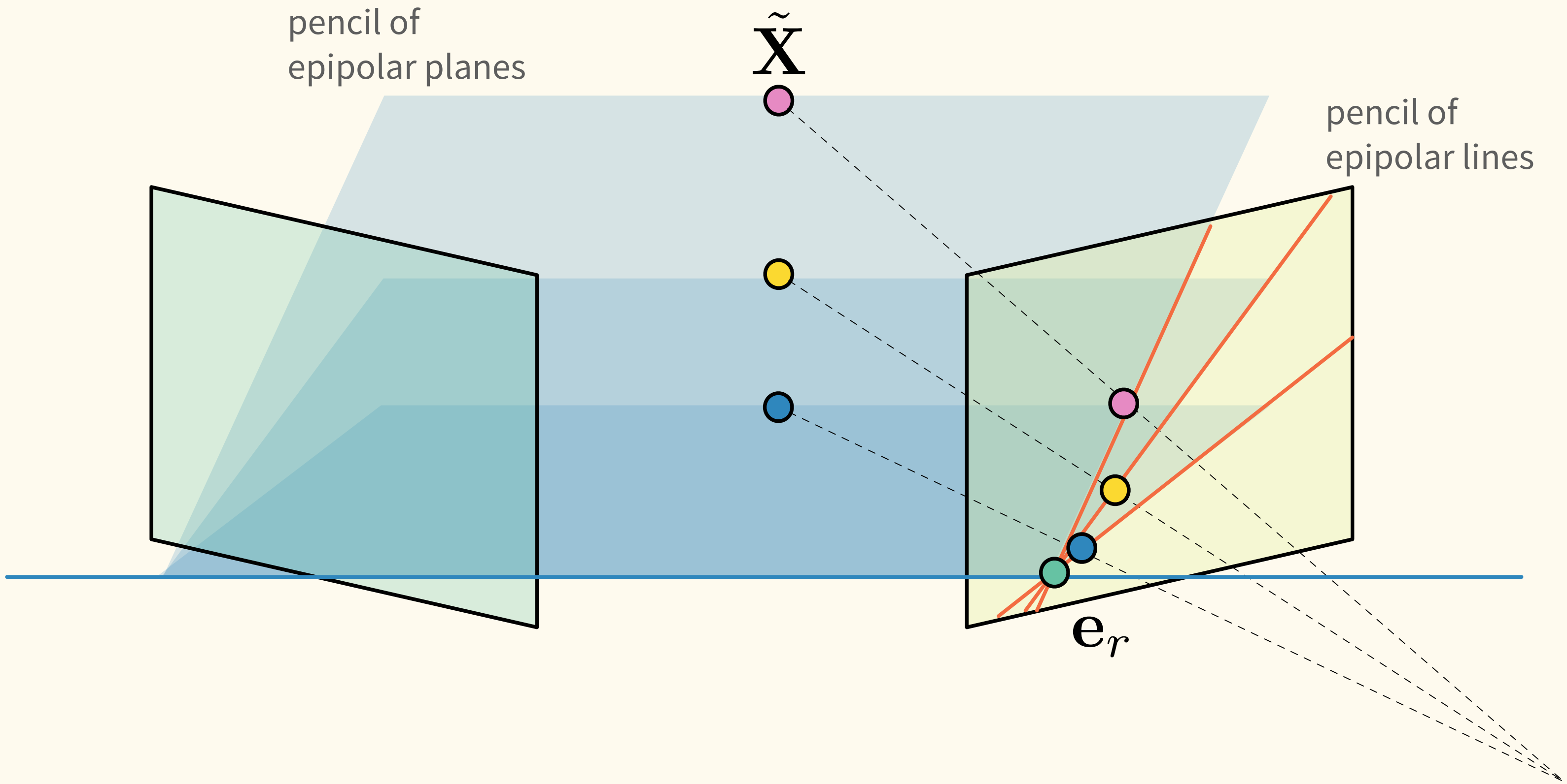
Fundamental matrix

Fundametal matrix:
the fundamental matrix F is the unique 3×3 matrix rank 2 homogeneous matrix which satisfy $\boldsymbol{x}_r^T F \boldsymbol{x}_l = 0$ for all corresponding points $\boldsymbol{x}_r \leftrightarrow \boldsymbol{x}_l$ in the two images

Why rank 2?

Epipolar lines can be seen as the intersection with the image plane of the pencil of planes (epipolar planes) having the baseline as axis.

Consider an epipolar lines $\boldsymbol{l}' = F\boldsymbol{x}_l$, The right epipole $\boldsymbol{e}_r$ lies on this line, so $\boldsymbol{e}_r^T F \boldsymbol{x}_l = 0$ for all $\boldsymbol{x}_l$. This implies that $\boldsymbol{e}_r^T F = 0$. Similarly, one can prove that $F\boldsymbol{e}_\ell = 0$, this gives an intuition of the reason why F is rank deficient.



Fundamental matrix

Fundametal matrix:
the fundamental matrix F is the unique 3×3 matrix rank 2 homogeneous matrix which satisfy $\boldsymbol{x}_r^T F \boldsymbol{x}_l = 0$ for all corresponding points $\boldsymbol{x}_r \leftrightarrow \boldsymbol{x}_l$ in the two images

Why rank 2?

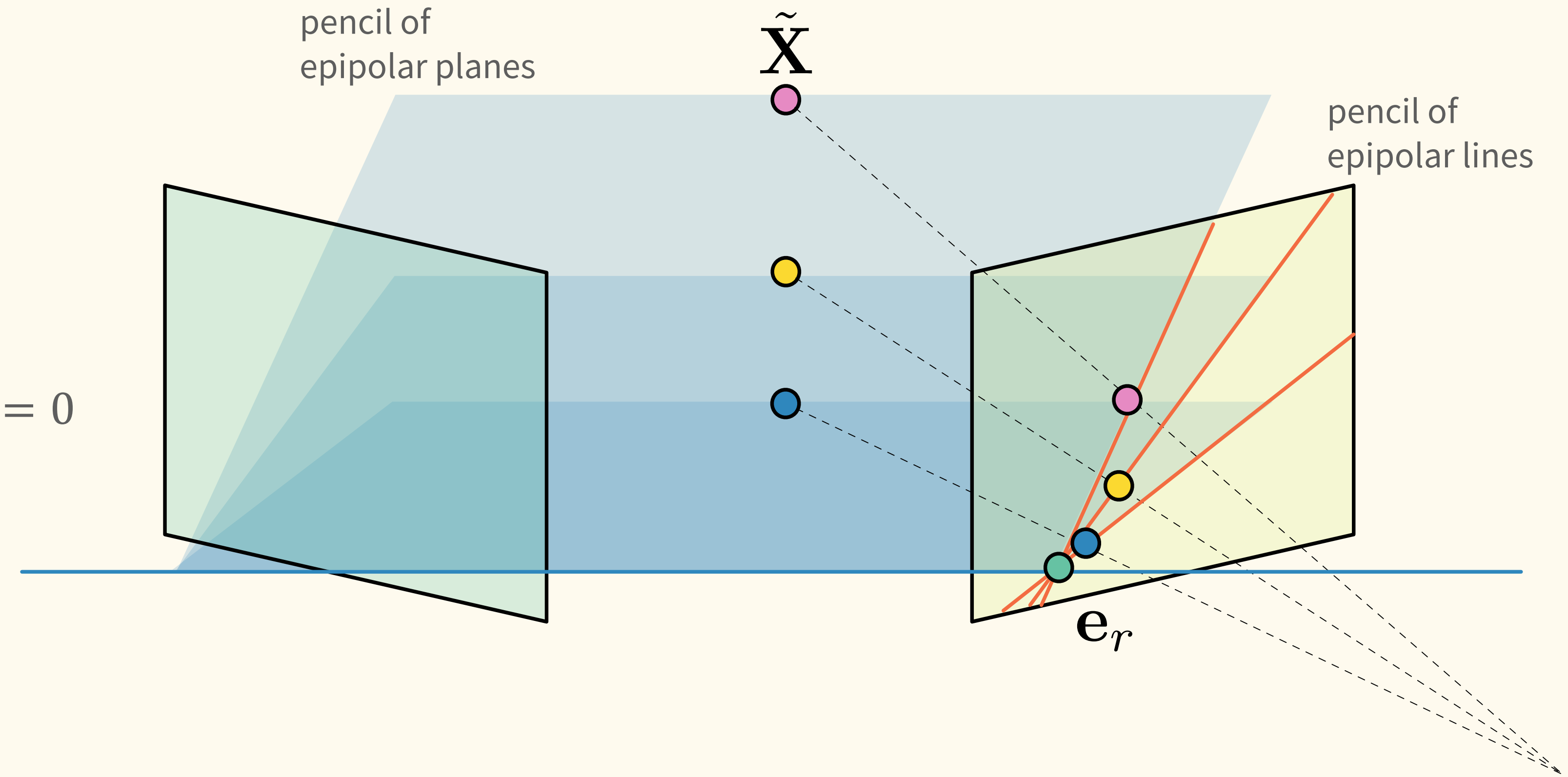
Algebically, we have

$$\det(F) = \det([\boldsymbol{e}_r]_\times P_{r\,1:3} P_{\ell\,1:3}^{-1})$$

that from the Binet theorem can be factorized as

$$\det(F) = \det([\boldsymbol{e}_r]_\times) \det\left(P_{r\,1:3} P_{\ell\,1:3}^{-1}\right) = 0$$

since $\det([\boldsymbol{e}_r]_\times) = 0$.



Fundamental matrix

Fundametal matrix:
the fundamental matrix F is the unique 3×3 matrix rank 2 homogeneous matrix which satisfy $\boldsymbol{x}_r^T F \boldsymbol{x}_l = 0$ for all corresponding points $\boldsymbol{x}_r \leftrightarrow \boldsymbol{x}_l$ in the two images

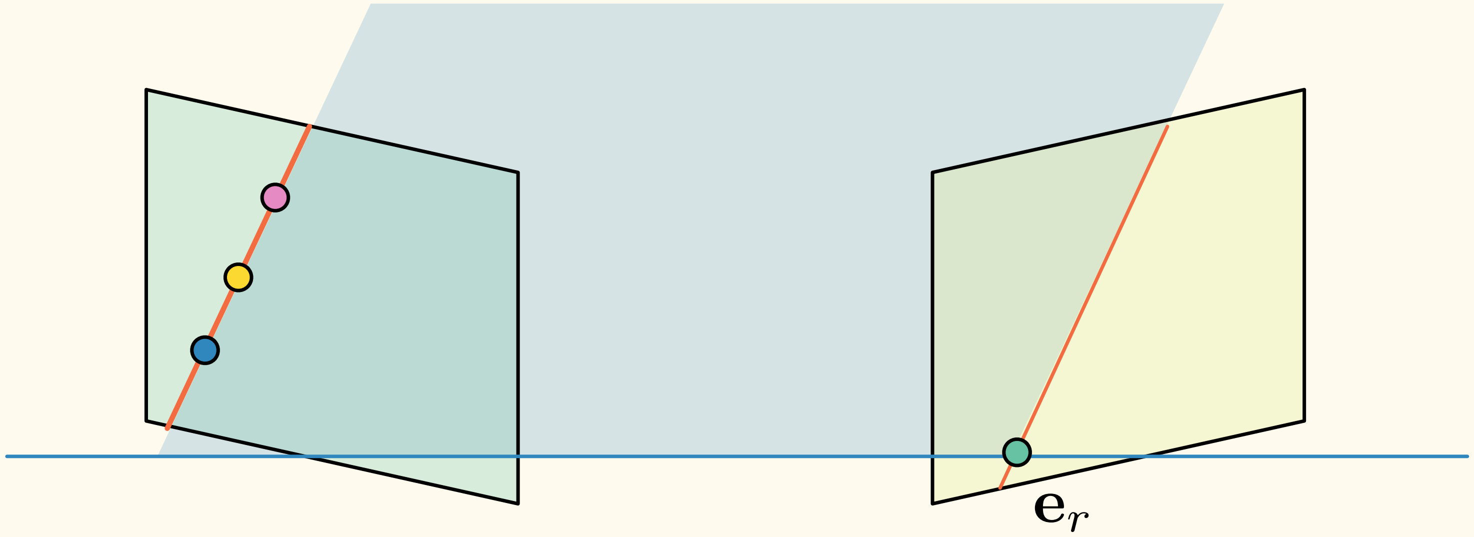
Why rank 2?

F is a projective map that associate a point in the first image to a line

$$F \colon \boldsymbol{x}_\ell \mapsto \boldsymbol{l}_r$$

If l_ℓ and l_r are corresponding epipolar lines, then any point $\boldsymbol{x}_\ell \in \boldsymbol{l}_\ell$ is mapped to the same line $\boldsymbol{l}_r$.

This means there is no inverse mapping and F is not of full rank.



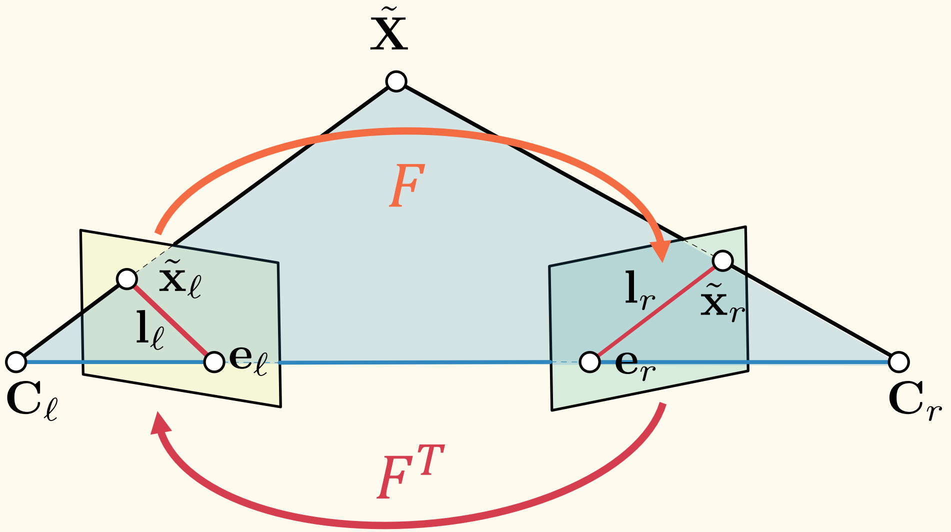
Fundamental matrix

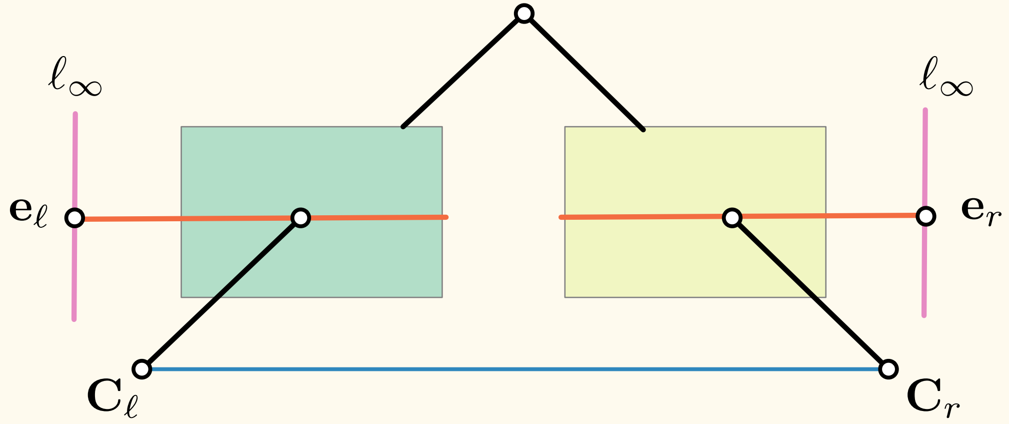
The fundamental matrix represents the condition that corresponding points $\boldsymbol{x}_r \leftrightarrow \boldsymbol{x}_l$ have to satisfy in the camera system.

This property enables computing F from pairs of corresponding points, without having to known P_ℓ and P_r

If F is the fundamental matrix of the pair of cameras P_ℓ and P_r, then F^T is the fundamental matrix of the pair of cameras in the opposite order: P_r and P_ℓ .

For any point x_ℓ in the right image the corresponding epipolar line is $\boldsymbol{l}_r = F\boldsymbol{x}_\ell$, similarly $\boldsymbol{l}_\ell = F^T\boldsymbol{x}_r$ identifies the epipolar line corresponding to $\boldsymbol{x}_r$ in the left image.





Essential matrix

When the interior parameters are known, we can assume that points are in normalized image coordinates (NIC). Using the NIC, the left and the right camera matrices can be chosen as $P_\ell = [I|\mathbf{0}]$ and $P_r = [R|\boldsymbol{t}]$

By substituting these cameras into the equation of the epipolar line, we get

$$\zeta_r \tilde{\boldsymbol{p}}_r = \boldsymbol{t} + \zeta_\ell R \tilde{\boldsymbol{p}}_\ell$$

So, the point $\tilde{\boldsymbol{p}}_r$ lies on the line through the points $\boldsymbol{t}$ and $R\tilde{\boldsymbol{p}}_\ell$:

$$\tilde{\boldsymbol{p}}_r^T (\boldsymbol{t} \times R \tilde{\boldsymbol{p}}_\ell) = 0$$

or

$$\tilde{\boldsymbol{p}}_r^T [\boldsymbol{t}]_\times R \tilde{\boldsymbol{p}}_\ell = 0$$

In summary, the relationship between the corresponding image points $\tilde{\boldsymbol{p}}_\ell \leftrightarrow \tilde{\boldsymbol{p}}_r$ in NIC is the bilinear form:

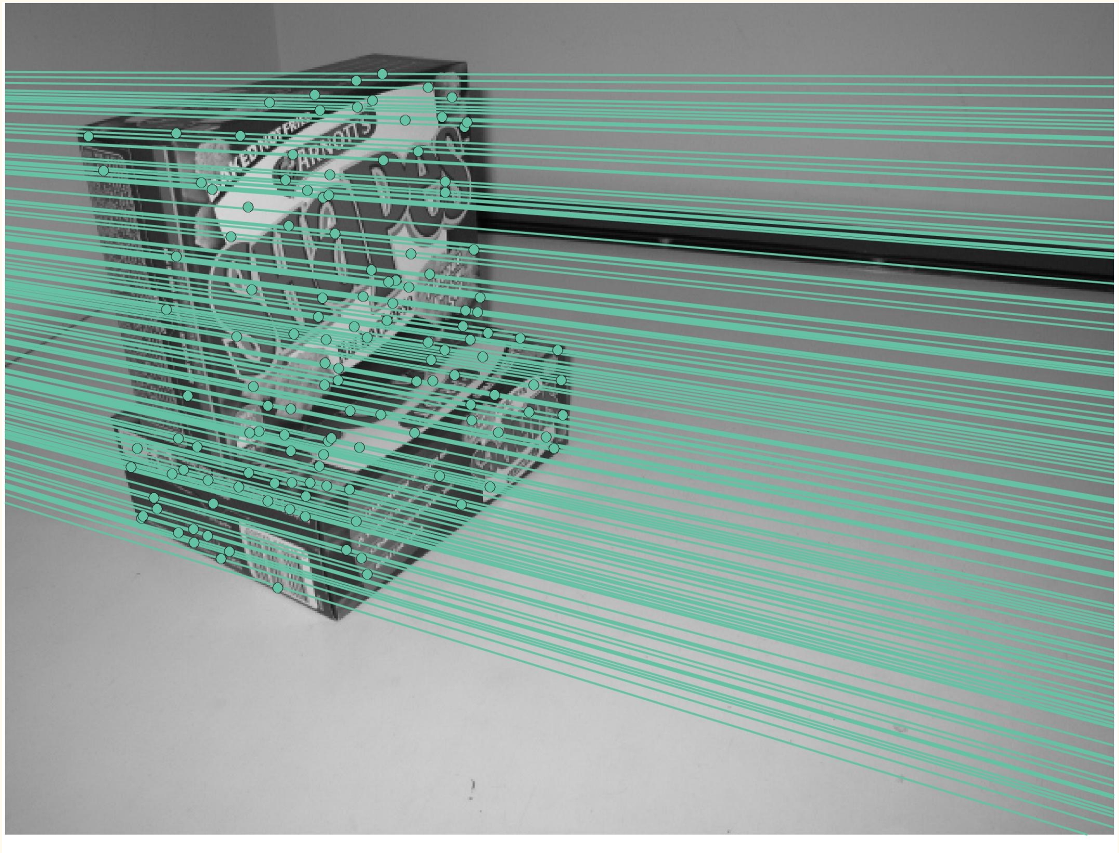
$$\tilde{\boldsymbol{p}}_r^T E \tilde{\boldsymbol{p}}_\ell = 0$$

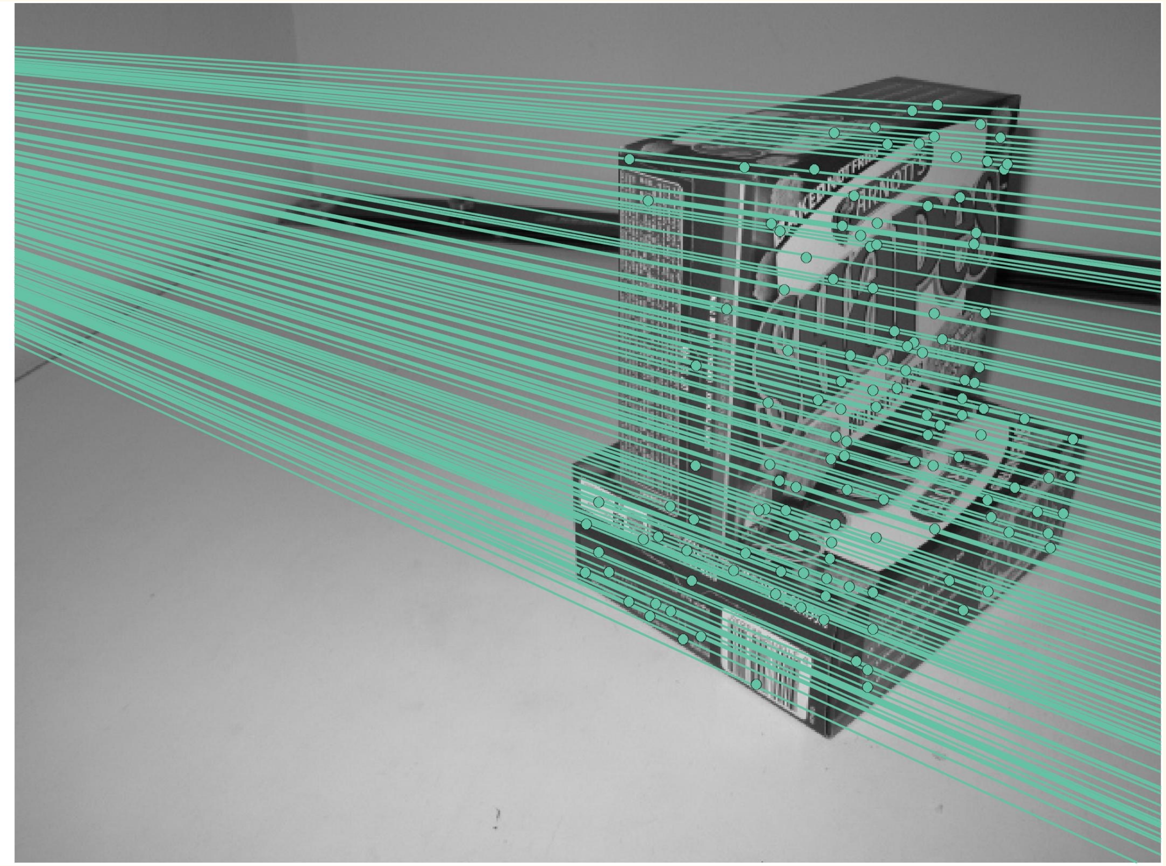
where $E = [\boldsymbol{t}]_\times R$ is called essential matrix and encodes the information on the rigid displacement between cameras. It has five degrees of freedom: a 3D rotation and a 3D translation direction.

Input images

Input images from the Adelaide RMF dataset

Correspondences and epipolar lines





The eight-points algorithm

Given a set of correspondences $\{\boldsymbol{x}_{i\ell} \leftrightarrow \boldsymbol{x}_{ir}\}$, we want to determine the matrix F that encodes the bilinear condition: $\boldsymbol{x}_{ir}^T F \boldsymbol{x}_{i\ell} = 0$

This matrix can be recovered using the property of the Kronecker product:

$$\boldsymbol{x}_{ir}^T F \boldsymbol{x}_{i\ell} = 0 \Leftrightarrow \operatorname{vec}\left(\boldsymbol{x}_{ir}^T F \boldsymbol{x}_{i\ell}\right) = 0 \Leftrightarrow \left(\boldsymbol{x}_{i\ell}^T \otimes \boldsymbol{x}_{ir}^T\right)\operatorname{vec}(F) = 0$$

Every correspondence yields a homogeneous equation in the 9 unknown of F. From n corresponding points we get the system:

$$\underbrace{\begin{bmatrix} \boldsymbol{x}_{1\ell}^T \otimes \boldsymbol{x}_{1r}^T \\ \boldsymbol{x}_{2\ell}^T \otimes \boldsymbol{x}_{2r}^T \\ \vdots \\ \boldsymbol{x}_{n\ell}^T \otimes \boldsymbol{x}_{nr}^T \end{bmatrix}}_{A_n} \operatorname{vec}(F) = 0.$$

The solution of this system is the $\ker(A_n)$. When the points are in general position and $n = 8$, the solution is determined up to a multiplicative factor. In practice, when more than 8 points are available the solution can be obtained using the SVD.

The projective reconstruction theorem

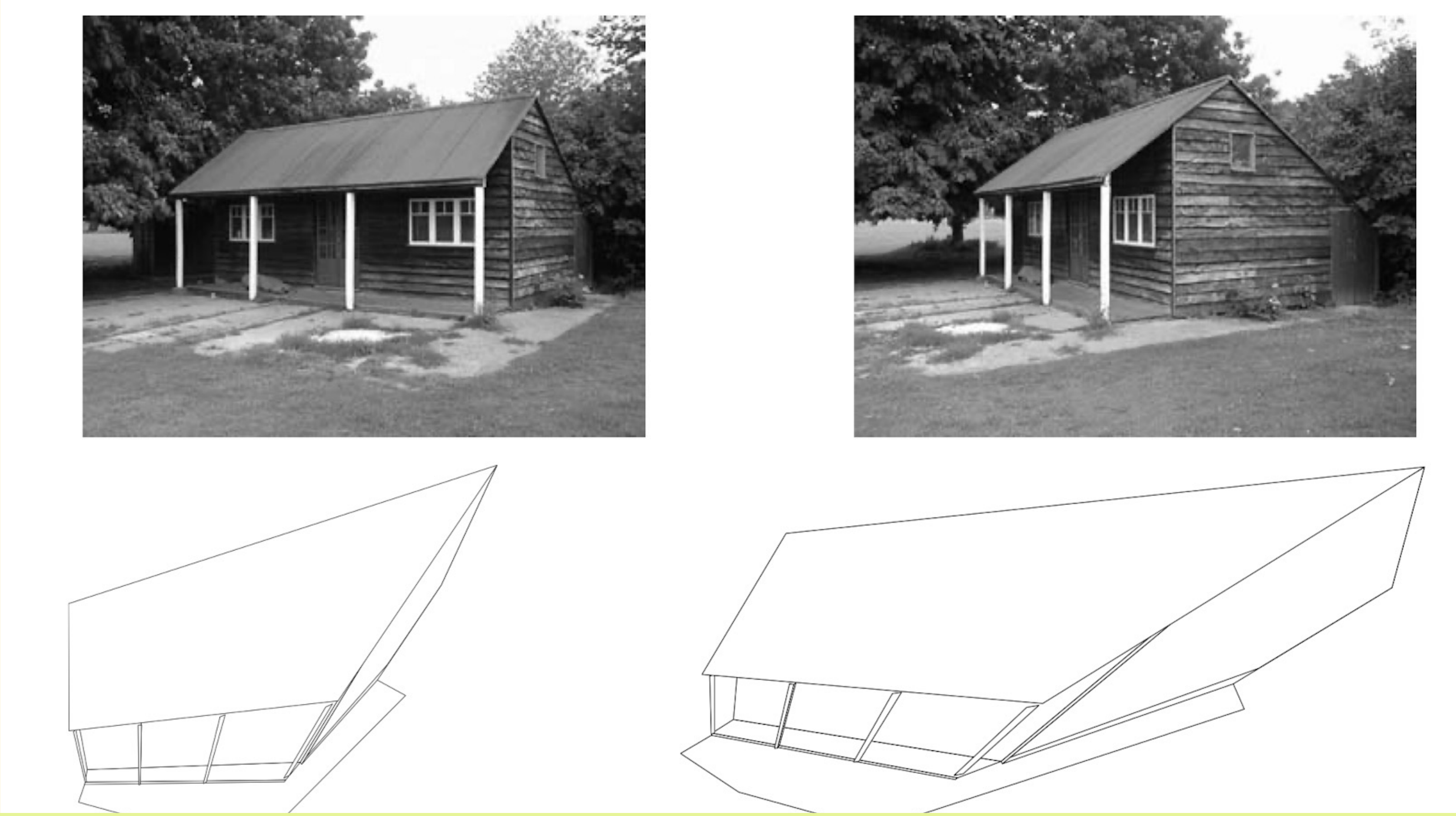


Image credits: Hartely Zisserman

One may compute a projective reconstruction of a scene from two views based on image correspondences alone, without knowing anything about the calibration or pose of the two cameras involved. In particular the true reconstruction is within a projective transformation of the projective reconstruction.

The projective reconstruction theorem

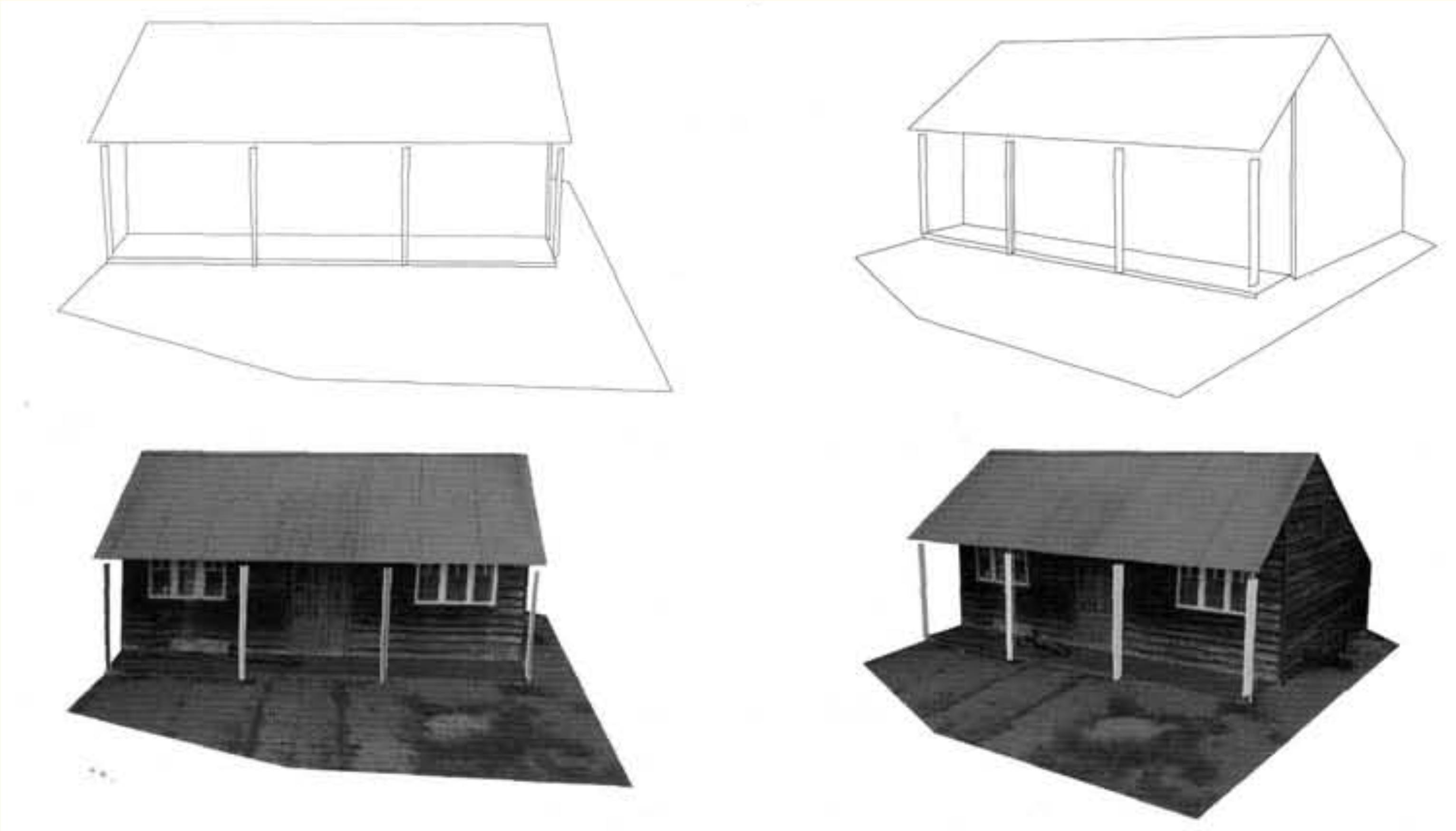


Image credits: Hartely Zisserman

If the calibration matrices are known, the scene can be reconstructed up to a similarity.
We still have scale ambiguity

Two view Structure from Motion

- We are given two image of a scene,
- we don’t know the poses of the cameras (in some cases we can assume that we do know the intrinsics)
- we want to compute the **3D Structure** of the scene **and** the **motion of the cameras**

There are several approaches to address this problem that produce different outputs:

Traditional methods

Deep learning + geometry

Traditional methods	Deep learning + geometry		
Relative depth and relative pose	Depth from a single image and relative pose	Absolute depth and absolute pose	Relative depth and relative pose

Wang, Jianyuan, et al. "Deep two-view structure-from-motion revisited." CVPR 2021.

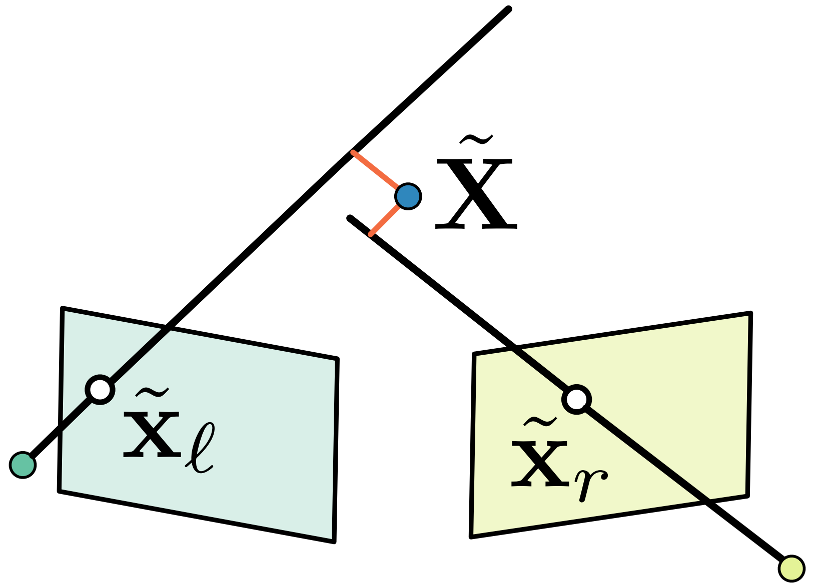
Traditional methods

Traditional methods

Deep learning + geometry

Relative depth and relative pose	Depth from a single image and relative pose	Absolute depth and absolute pose	Relative depth and relative pose

- Compute sparse correspondences (handcrafted feature)
- Estimate relative pose between the cameras (robust fitting)
- Triangulate points to get the 3D structure
- Optimize for the position of the triangulated points and for the pose of the cameras (Bundle Adjustment)
- Rectify the cameras
- Compute disparity to get dense correspondences
- Triangulate a dense point cloud



SfM learner

Traditional methods

Deep learning + geometry

Relative depth and relative pose	**Depth from a single image and relative pose**	Absolute depth and absolute pose	Relative depth and relative pose

As in stereo self-supervision, rely on view-synthesis.

Source view I_S

view synthesis

depends on:

- depth
- pose (R, t)

Photometric error

Novel view I_N

Target view I_T

The source view I_S is warped via the estimated pose to a novel view I_N.

The loss is the the photometric error between I_N and I_T, and the network learns both the relative depth and the relative pose.

Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe, Unsupervised Learning of Depth and Ego -motion from Video, CVPR 2017

SfM learner

Traditional methods

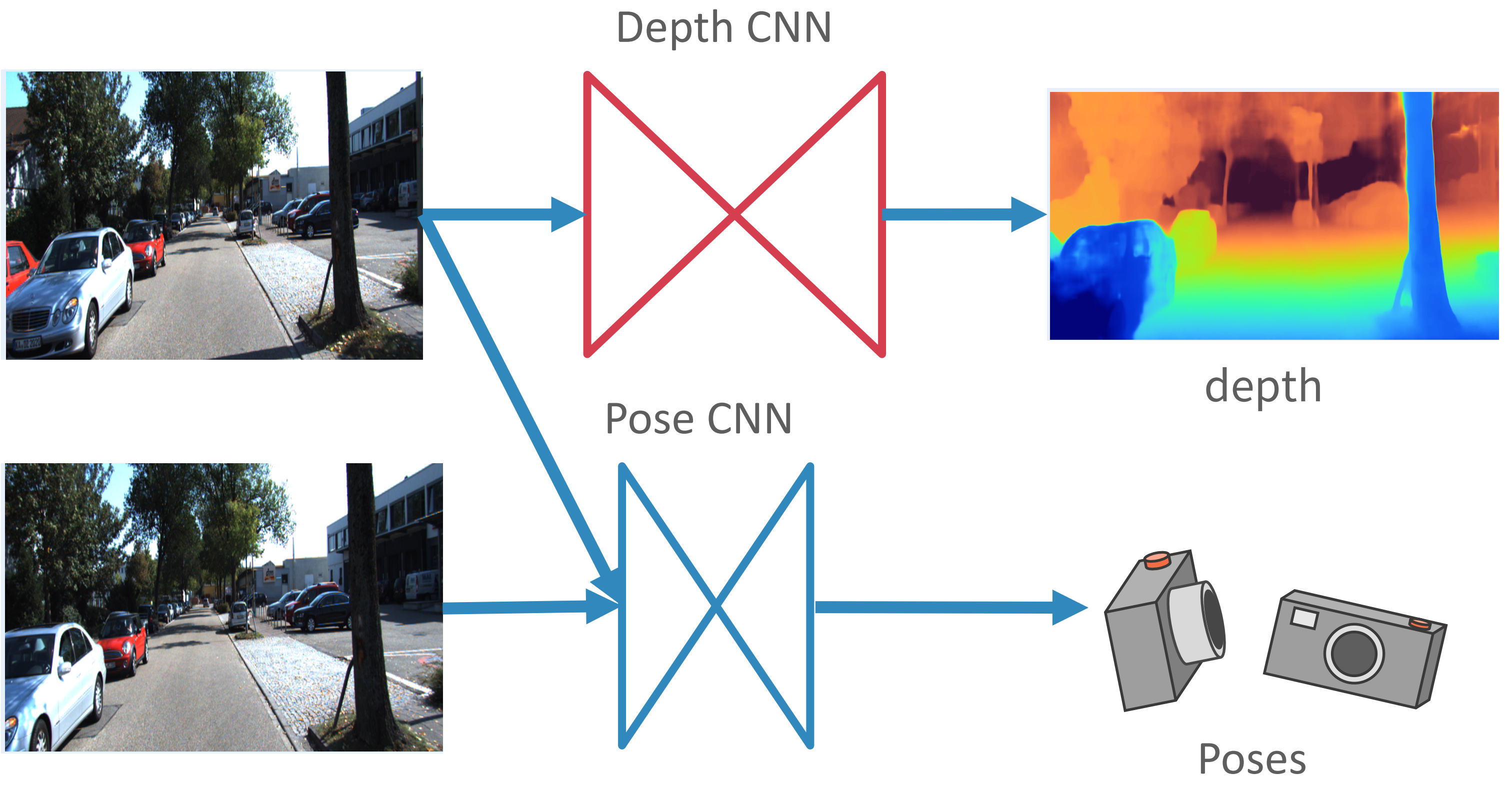
Relative depth and relative pose

Deep learning + geometry

Depth from a single image and relative pose

Absolute depth and absolute pose

Relative depth and relative pose



Depth is estimated from a single image (ill posed).
Rely on priors on the training data

The relative pose is estimated.
At test time, the Pose CNN can be discarded

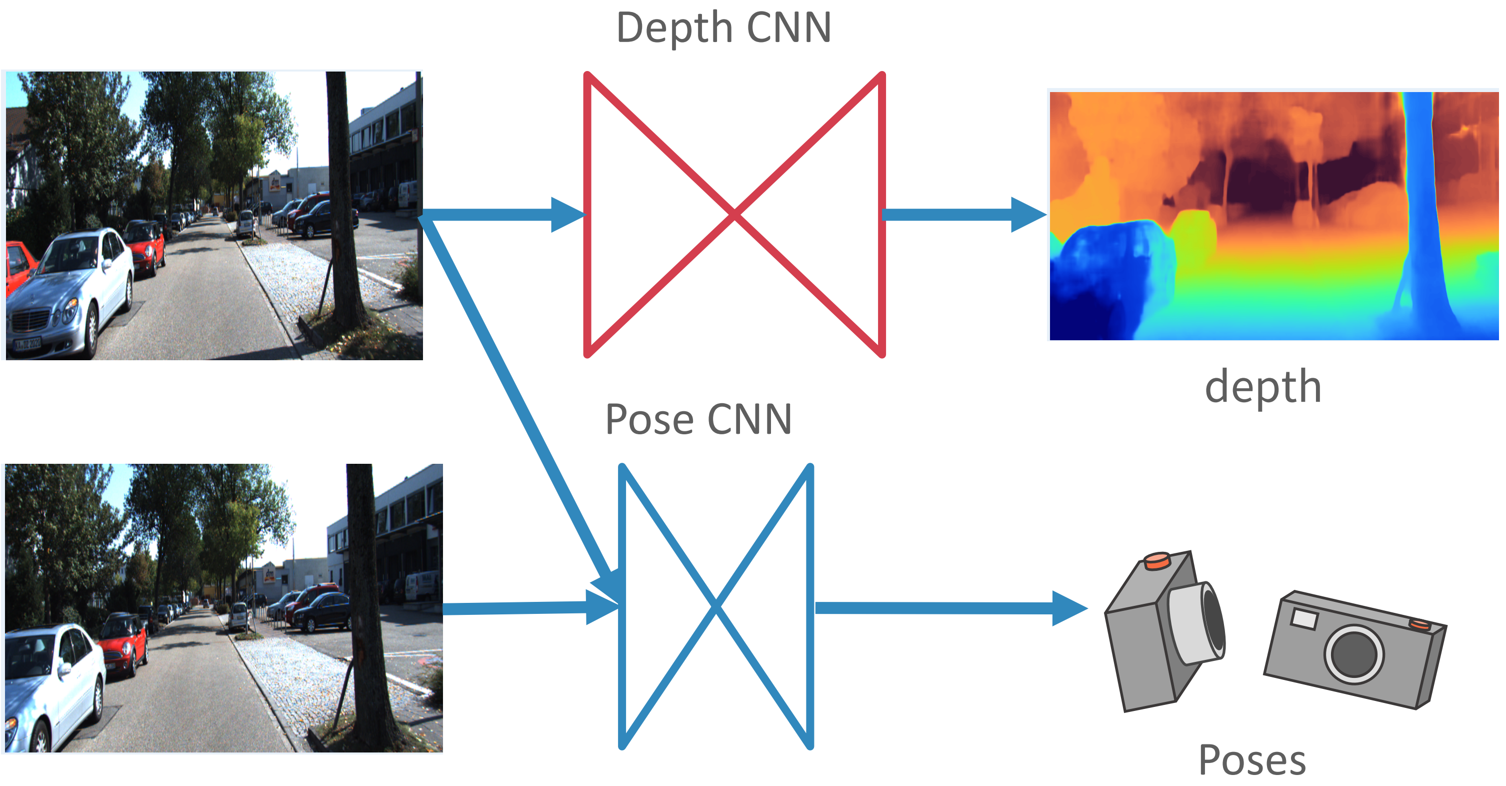
Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe, Unsupervised Learning of Depth and Ego-motion from Video, CVPR 2017

SfM learner

Traditional methods

Deep learning + geometry

Relative depth and relative pose	**Depth from a single image and relative pose**	Absolute depth and absolute pose	Relative depth and relative pose



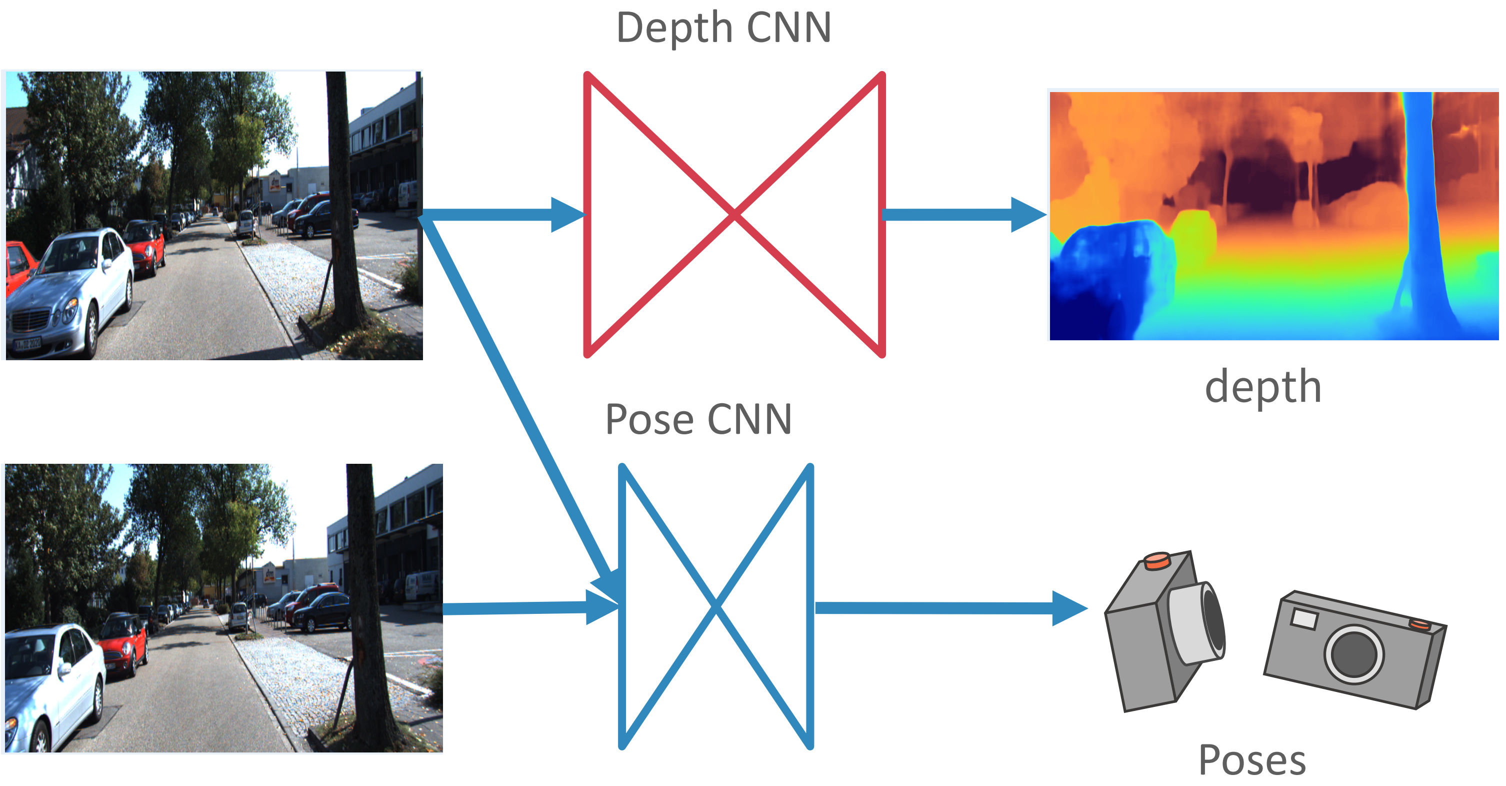
- The network implement a **multiscale** mechanism, since bilinear interpolation is too local
- It also predict a **mask** to model occlusions and moving object
- The **loss** accounts also for regularizing the mask and smooth depth values

Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe, Unsupervised Learning of Depth and Ego-motion from Video, CVPR 2017

SfM learner

Traditional methods | *Deep learning + geometry*

- Relative depth and relative pose
- **Depth from a single image and relative pose**
- Absolute depth and absolute pose
- Relative depth and relative pose



It doesn't work for
- ✗ Uniform regions
- ✗ Ego-velocity objects

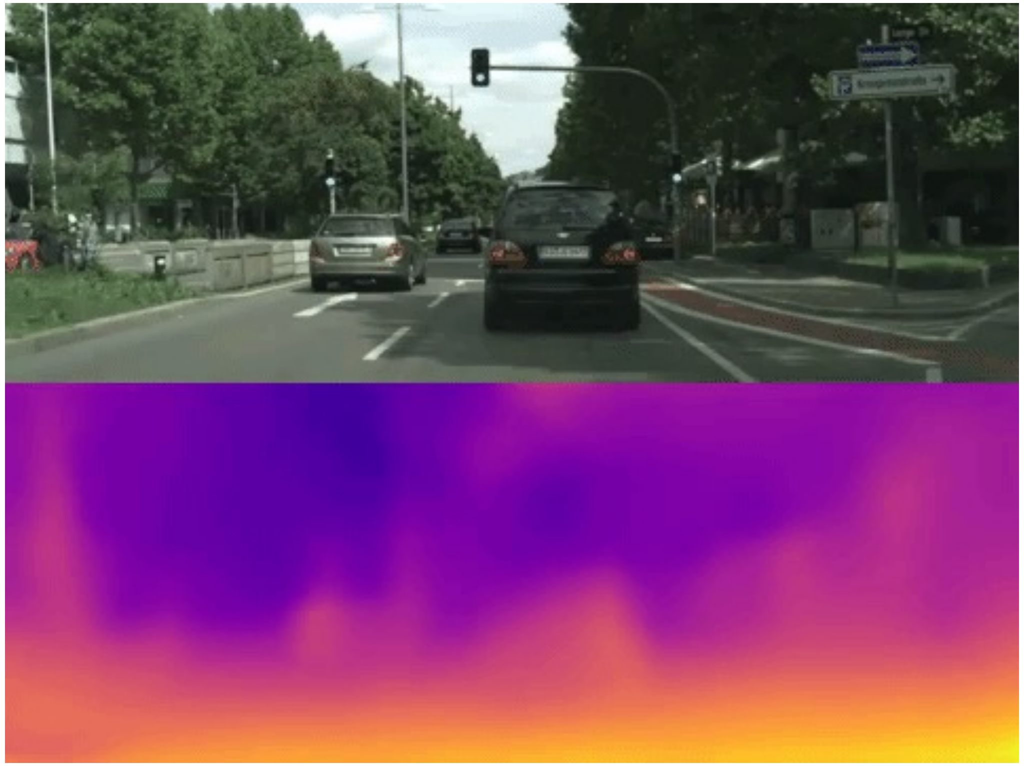
Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe, Unsupervised Learning of Depth and Ego-motion from Video, CVPR 2017

SfM learner

Traditional methods

Deep learning + geometry

- Relative depth and relative pose
- **Depth from a single image and relative pose**
- Absolute depth and absolute pose
- Relative depth and relative pose



- ✓ By using mono-training we can predict the depth-map for a video acquired by a moving camera
- ✓ Other and more robust losses can be used: such as ICP loss, motion segmentation loss, or epipolar loss.
- ✗ With respect to stereo self-supervision result are less accurate (edges are not sharp). Why?

More things to learn! (depth and pose)
Unknown scale, by estimating the depth from a single image we are addressing an ill-posed problem.

Tinghui Zhou, Matthew Brown, Noah Snavely, David Lowe, Unsupervised Learning of Depth and Ego-motion from Video, CVPR 2017

Supervised SfM

Traditional methods

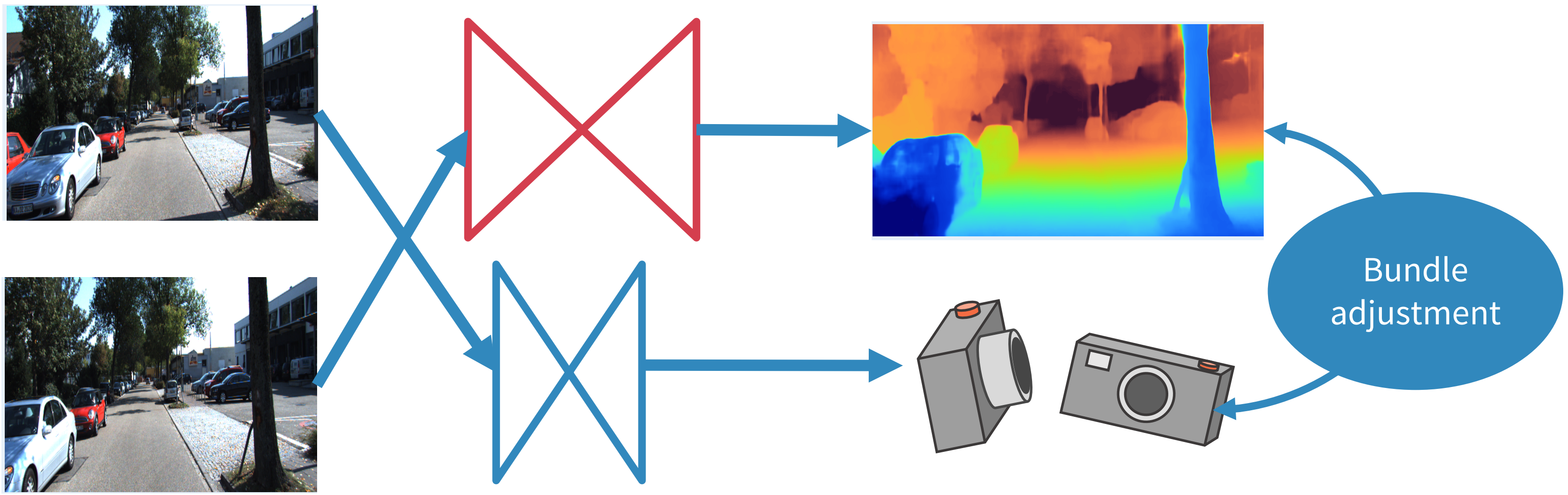
Relative depth and relative pose

Deep learning + geometry

Depth from a single image and relative pose

Absolute depth and absolute pose

Relative depth and relative pose



- Ground-truth depth as supervision,
- poseNet need to estimate camera poses with absolute scale (ill-posed)
- to mitigate this use dataset priors and semantic knowledge of the scene

Ummenhofer, Benjamin, et al. "Demon: Depth and motion network for learning monocular stereo" CVPR. 2017

Wang,et al. "Displacement-invariant matching cost learning for accurate optical flow estimation” NIPS 2020

Wei, Xingkui, et al. "Deepsfm: Structure from motion via deep bundle adjustment” ECCV 2020

Removing posenet

Traditional methods

Deep learning + geometry

Relative depth and relative pose	Depth from a single image and relative pose	Absolute depth and absolute pose	Relative depth and relative pose

Supervised SfM assume that a consistent scale of depth and pose can be learned across all input samples, which makes the learning problem harder, resulting in degraded performance and limited generalization.

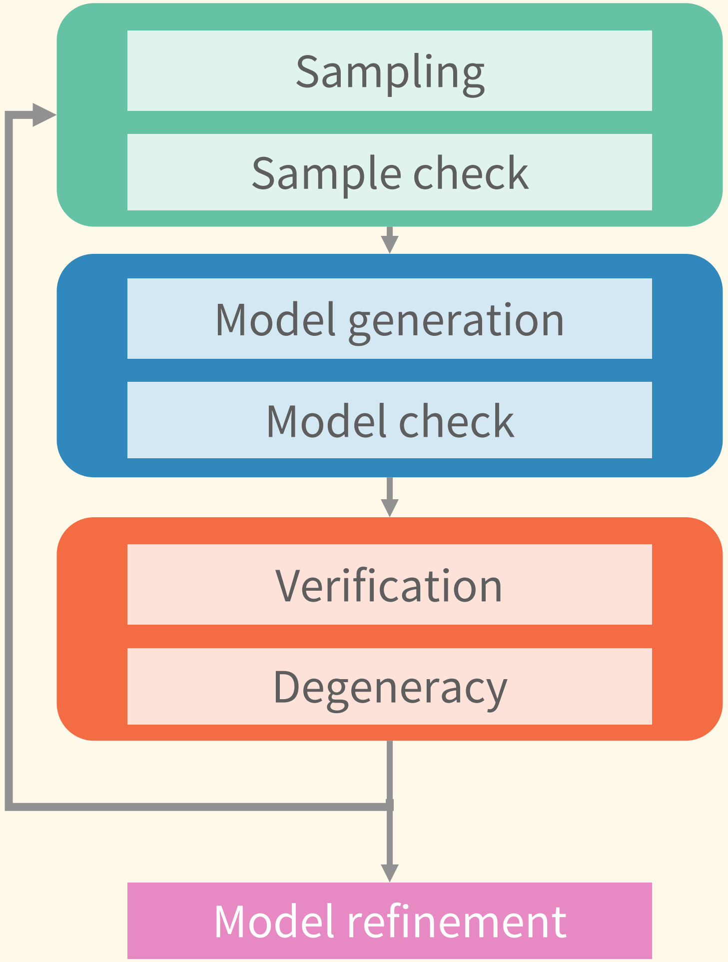
The idea is to disentangles scale from the network estimation and follow more closely traditional pipelines

	Traditional	***Deep + geometry***
Correspondences	Sparse handcrafted features	Optical flow
Relative pose	**Robust fitting (8/5 points algorithm)**	**Robust fitting (8/5 points algorithm)**
3D scene	Triangulation	Depth estimation

Zhao, Wang, et al. "Towards better generalization: Joint depth-pose learning without posenet." CVPR 2020.

Wang, Jianyuan, et al. "Deep two-view structure-from-motion revisited." CVPR 2021.

RanSaC in short

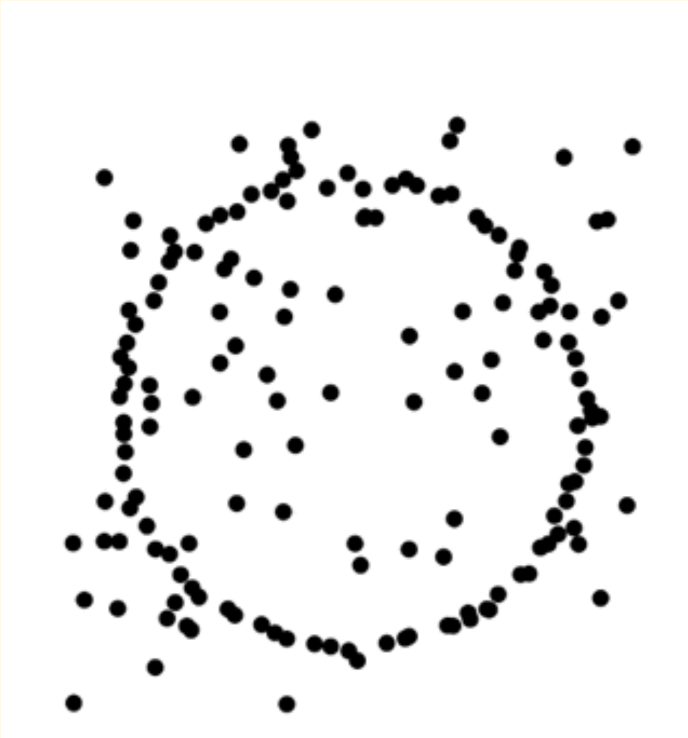


Sample minimal subset

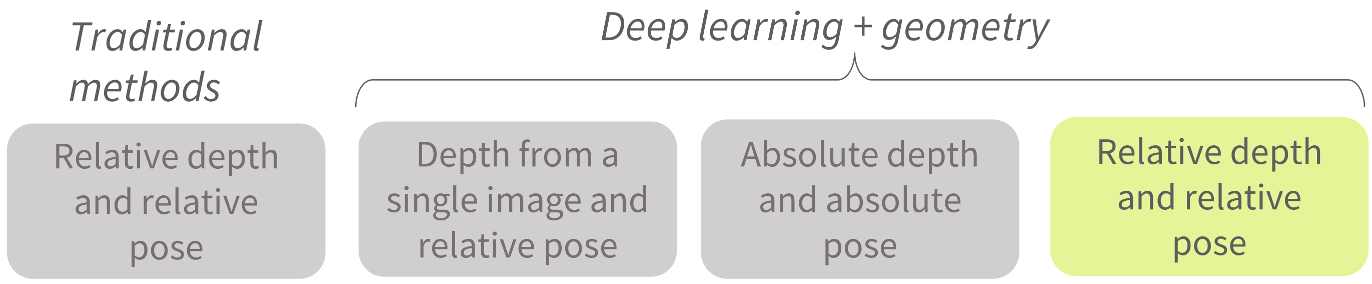
Generate minimal sample model

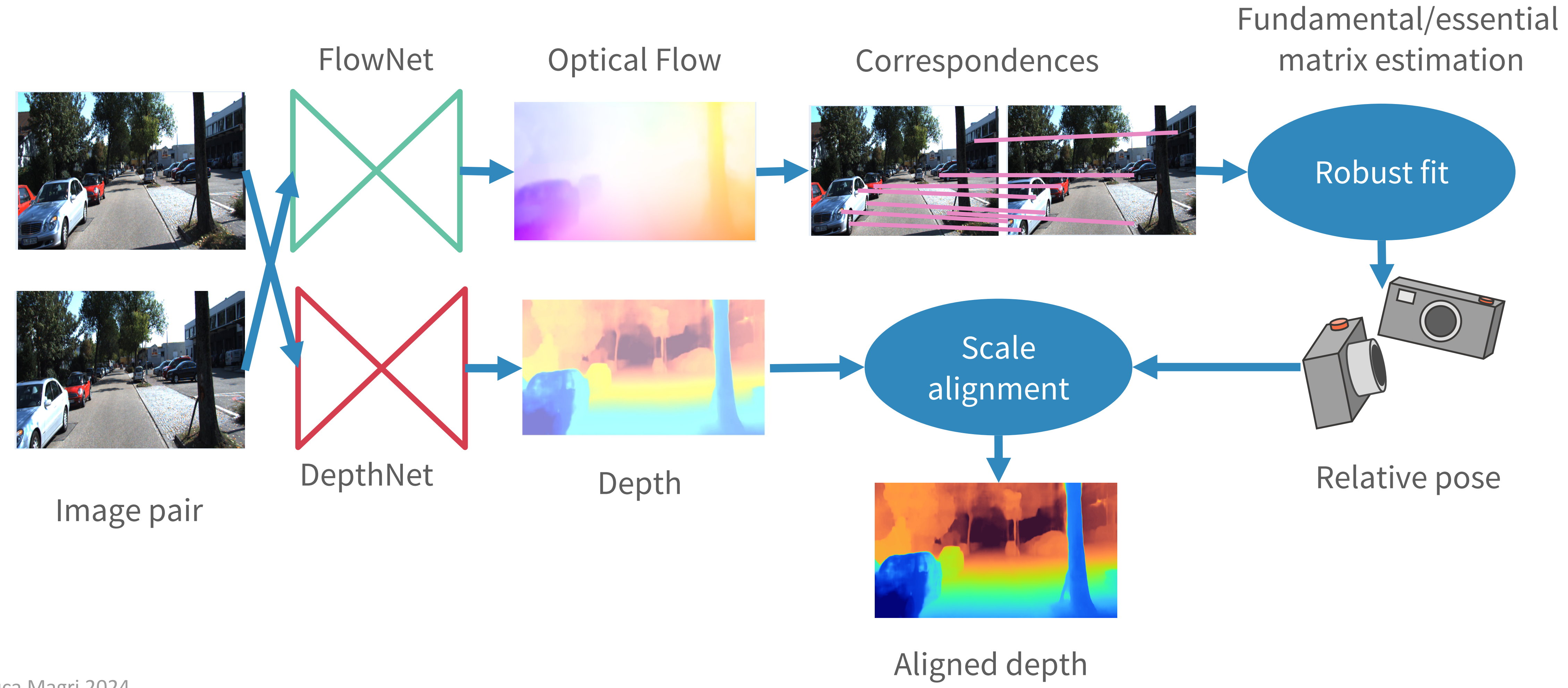
Is the model interesting? Keep it!
e.g., count inliers

Generate non minimal sample model



Removing posenet





Removing posenet: two view triangulation as depth supervision

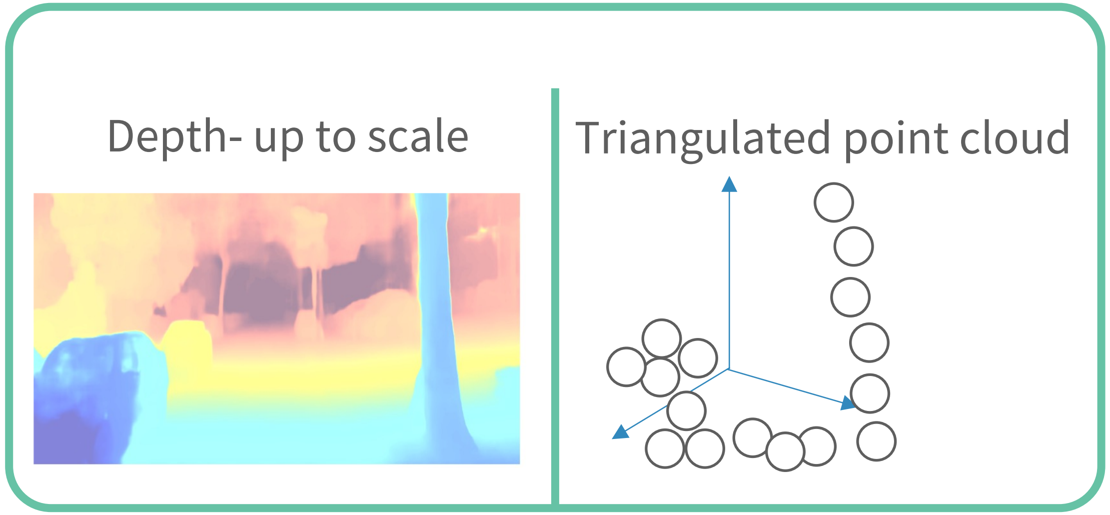
Aligning the depth with the pose

1. Select accurate correspondences (taking into consideration epipolar distance, and occlusion mask)
2. Reconstruct an up to scale structure using mid-point triangulation (differentiable)

The **loss** of the network is composed by:

- The unsupervised loss for the optical flow,
- The loss between triangulated, and predicted depth,
- The reprojection error (depth map reconstruction + flow error between optical flow and rigid flow generated by depth reprojection),
- Depth smoothness term.

Loss between triangulated and predicted depth



Zhao, Wang, et al. "Towards better generalization: Joint depth-pose learning without posenet." CVPR 2020.

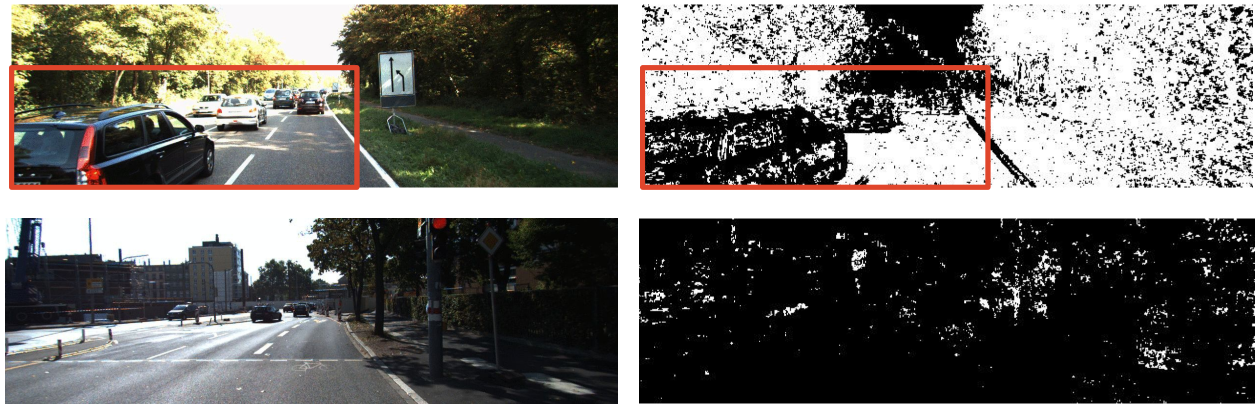
Static scene assumption

All SfM methods rely on the assumption that the scene is static, thus, as regard moving objects we have two approaches:

1. Detect and ignore: e.g.:

Masks estimated by SfMLearner

Automasking stationary pixels by Monodepthv2: ignore pixels in the loss which don't appear to change between images. Allow to ignore whole frames in monocular videos when the camera stops moving.

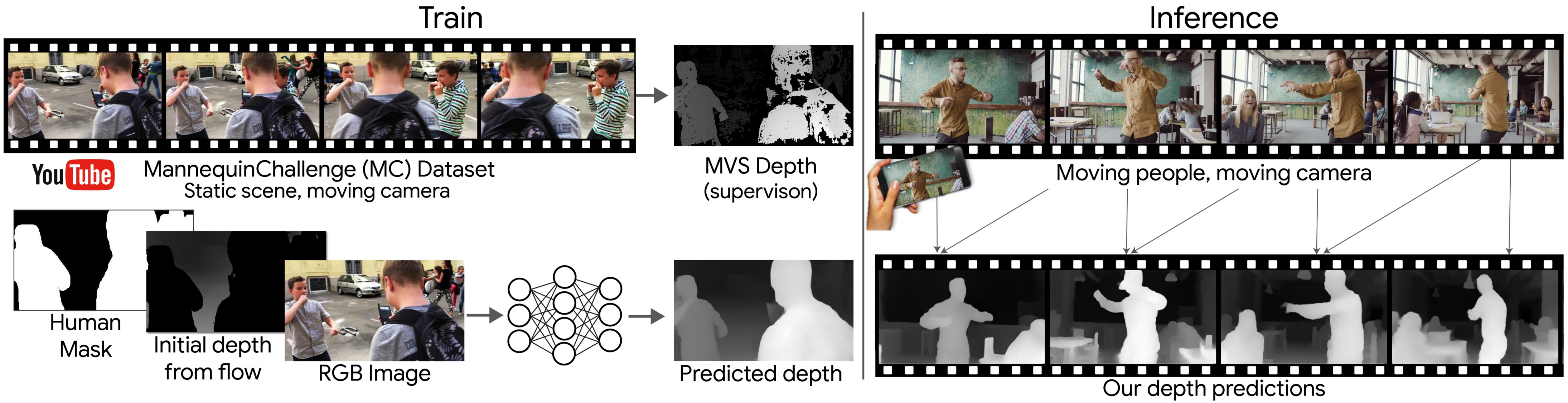


Zhao, Wang, et al. "Towards better generalization: Joint depth-pose learning without posenet." CVPR 2020.

Static scene assumption

All SfM methods rely on the assumption that the scene is static, thus, as regard moving objects we have two approaches:

1. Detect and ignore:
2. Clever tricks: Mannequin Challenge Dataset



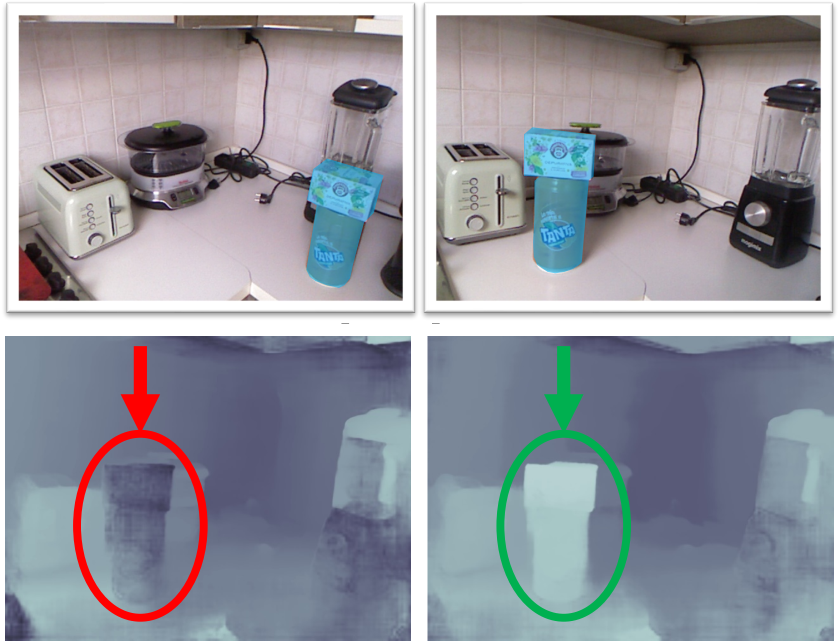
Li, Zhengqi, et al. "MannequinChallenge: Learning the depths of moving people by watching frozen people." TPAMI 2020.

Static scene assumption

All SfM methods rely on the assumption that the scene is static, thus, as regard moving objects we have two approaches:

1. Detect and ignore
2. Clever tricks
3. Modeling moving objects

Images of a scene with moving object



SOTA (DeepSfM)

Multi-Body Depth & Camera Pose Estimation

Dal Cin, Andrea Porfiri, Giacomo Boracchi, and Luca Magri. "Multi-body Depth and Camera Pose Estimation from Multiple Views." ICCV, 2023.

Multi-body Depth and Camera Pose Estimation

Setup:

- The scene is composed by multiple rigid bodies moving independently
- We have a **sparse** set of images (this differs from the previous monodepth approach where the input is typically a video)

Problem:

If one is able to segment the scene, using SfM pipelines you get indepent reconstruction each in its own scale.

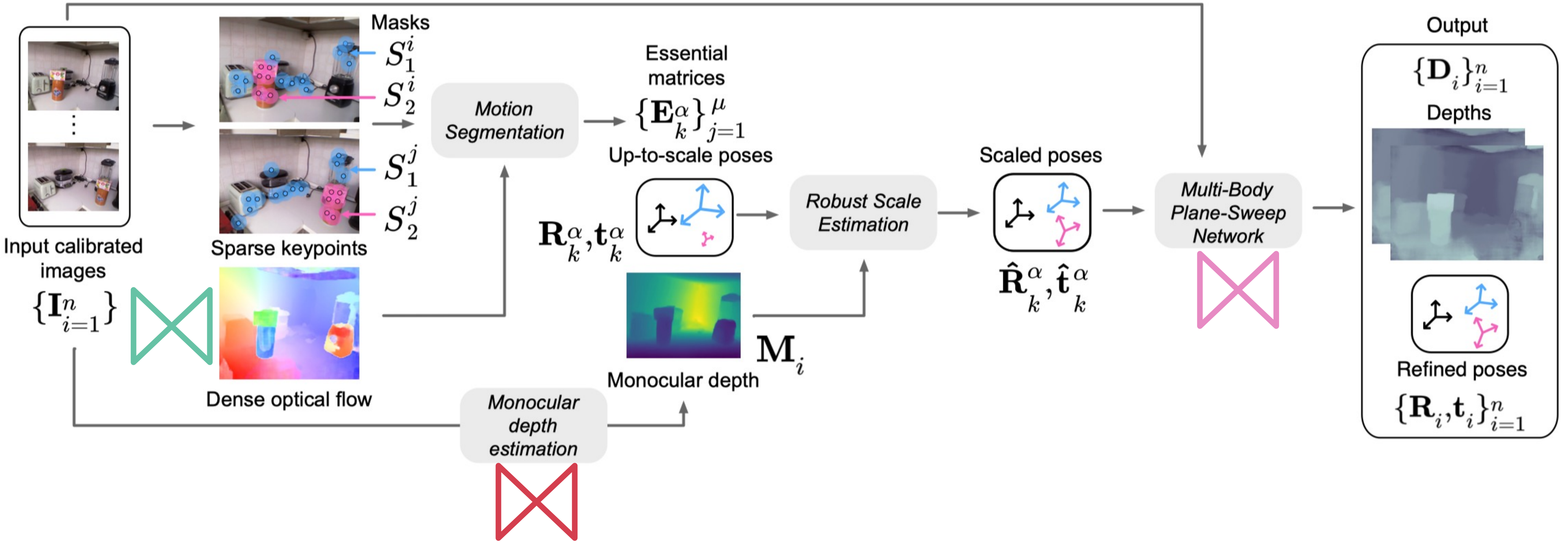
Goal:

We want to reconcile all the reconstruction to the same scale

Dal Cin, Andrea Porfiri, Giacomo Boracchi, and Luca Magri. "Multi-body Depth and Camera Pose Estimation from Multiple Views." ICCV, 2023.

Multi-body Depth and Camera Pose Estimation

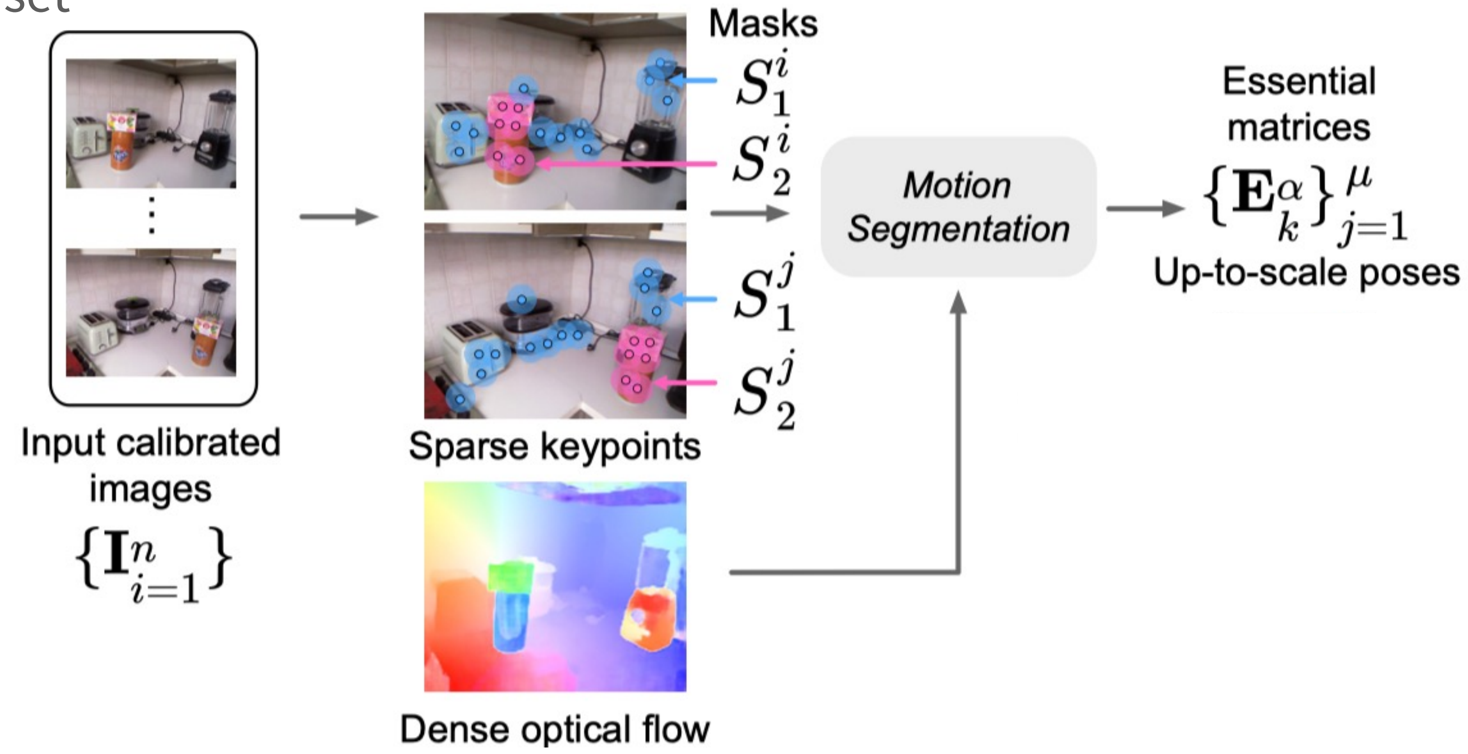
Combine several ingredients that we have seen so far…



Multi-body Depth and Camera Pose Estimation

Motion segmentation leverages multi-model robust fitting on the optical flow between the two input images to estimate several essential matrices.

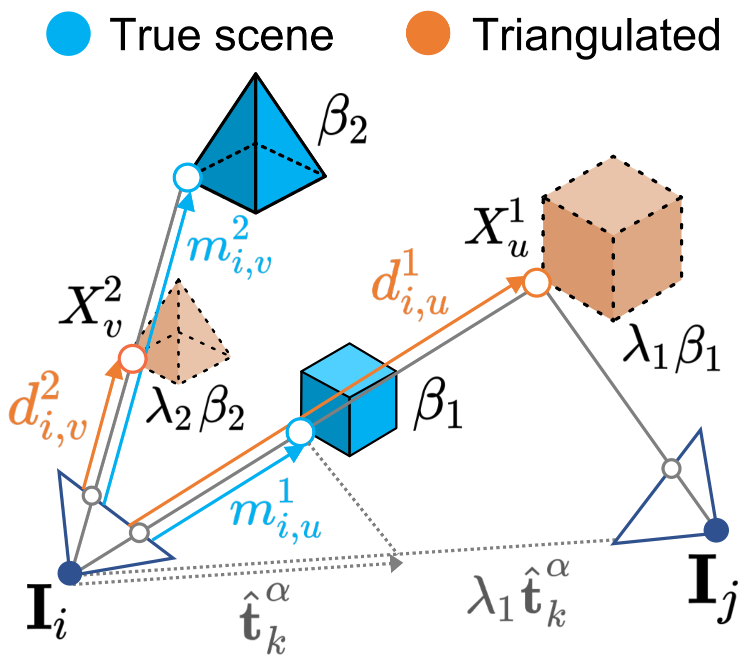
- SIFT keypoints are clustered according to object motion (multi-model fitting + synchronization)
- Dense optical flow matches augment the set of keypoints
- For each motion in the scene, we compute essential matrices, and **up-to-scale** camera poses

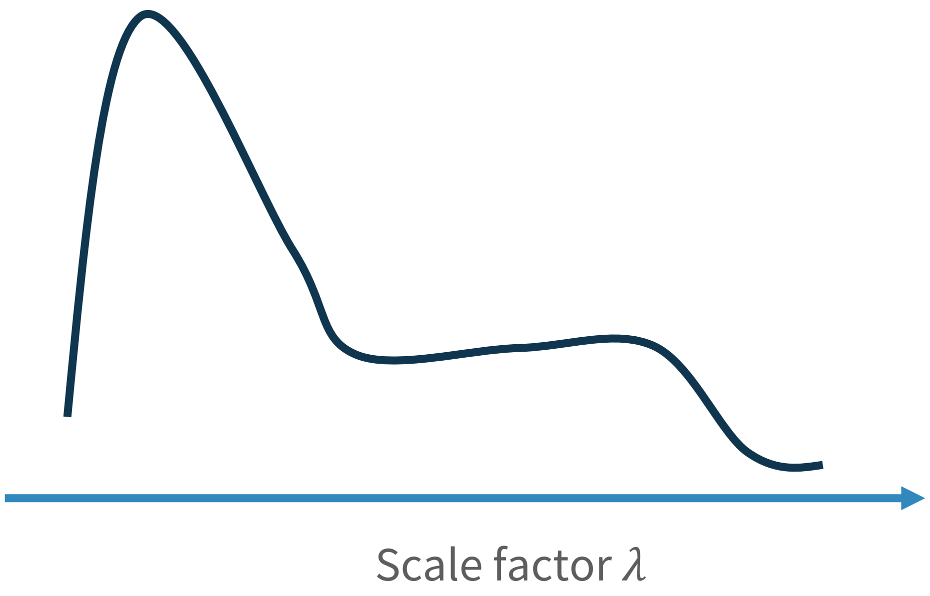


Multi-body Depth and Camera Pose Estimation

Scale estimation: The monocular depth is used as a prior that can be used to reconcile all the poses in the same scale.

For each image and each moving object, the ratio between the mono and triangulated depth is computed using a Kernel Density Voting.

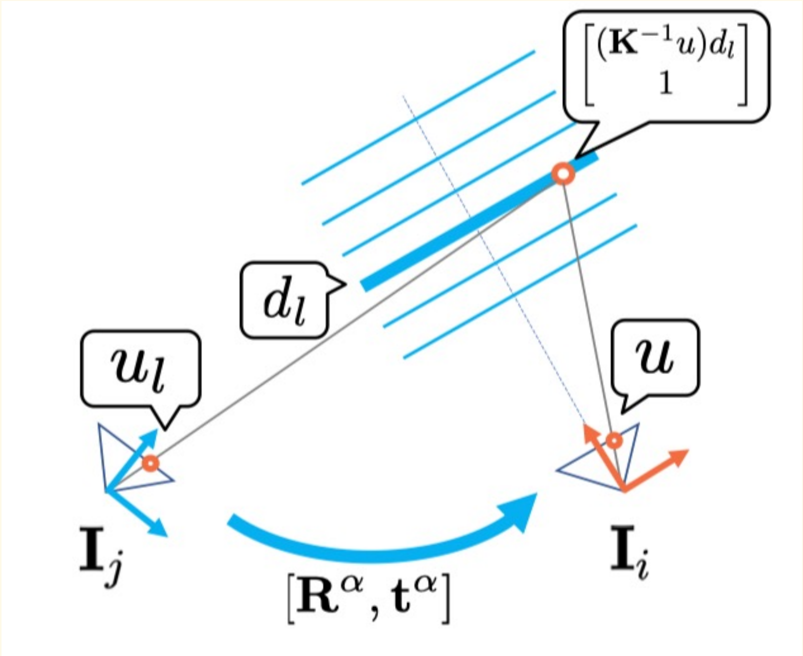




Plane sweep

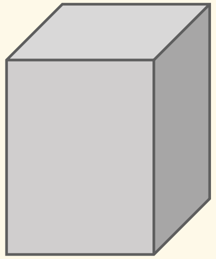
At the core of several traditional plane estimation algorithm, based on photoconsitency

Input: an image pair of a source and a target image



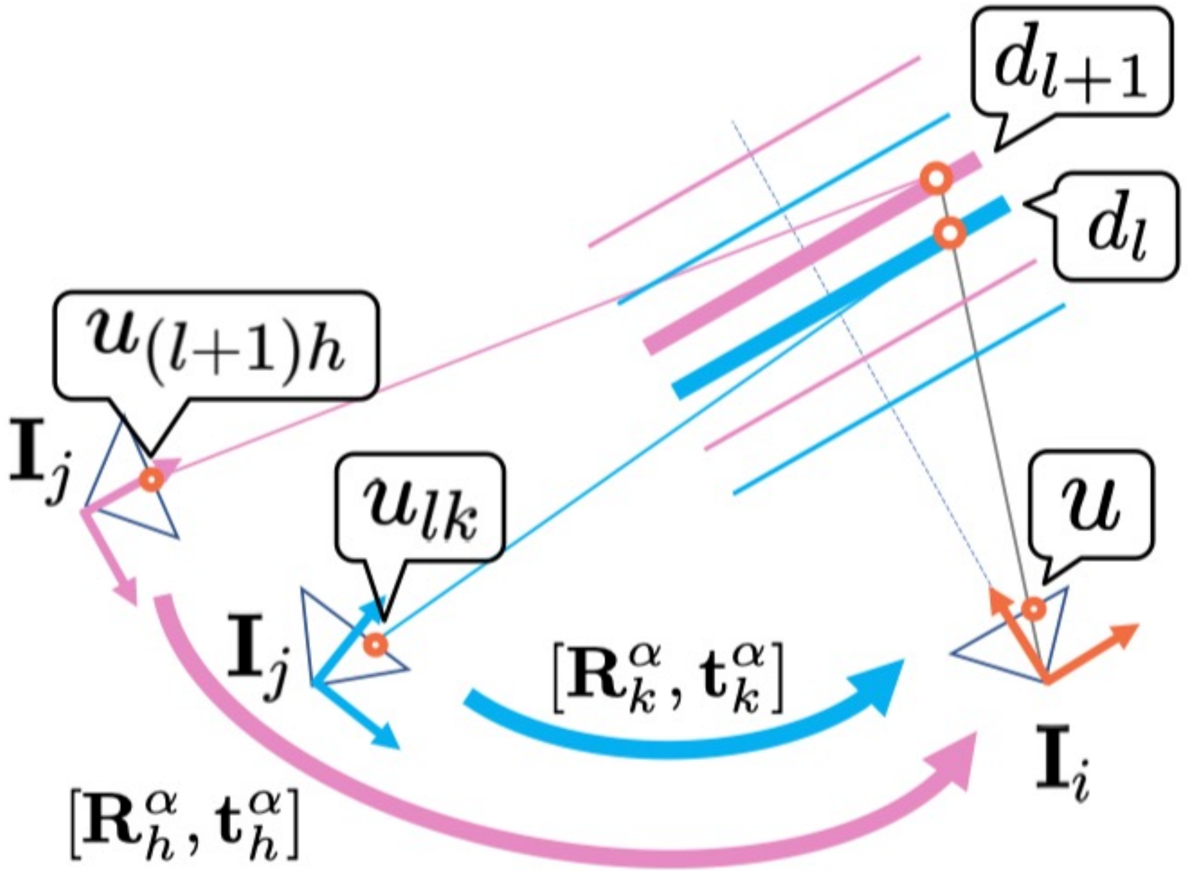
- Tentative depth planes parallel to the target are sampled
- For each pixel u, and each plane at depth d_l the intersection of the optical ray and the plane is projected onto the source view (the projection u_l depends both on depth and relative pose)
- A photometric error is computed by comparing the image values of pixels u and u_l

The errors can be packaged in a cost volume and the depth of the scene is the surface with the minimum cost

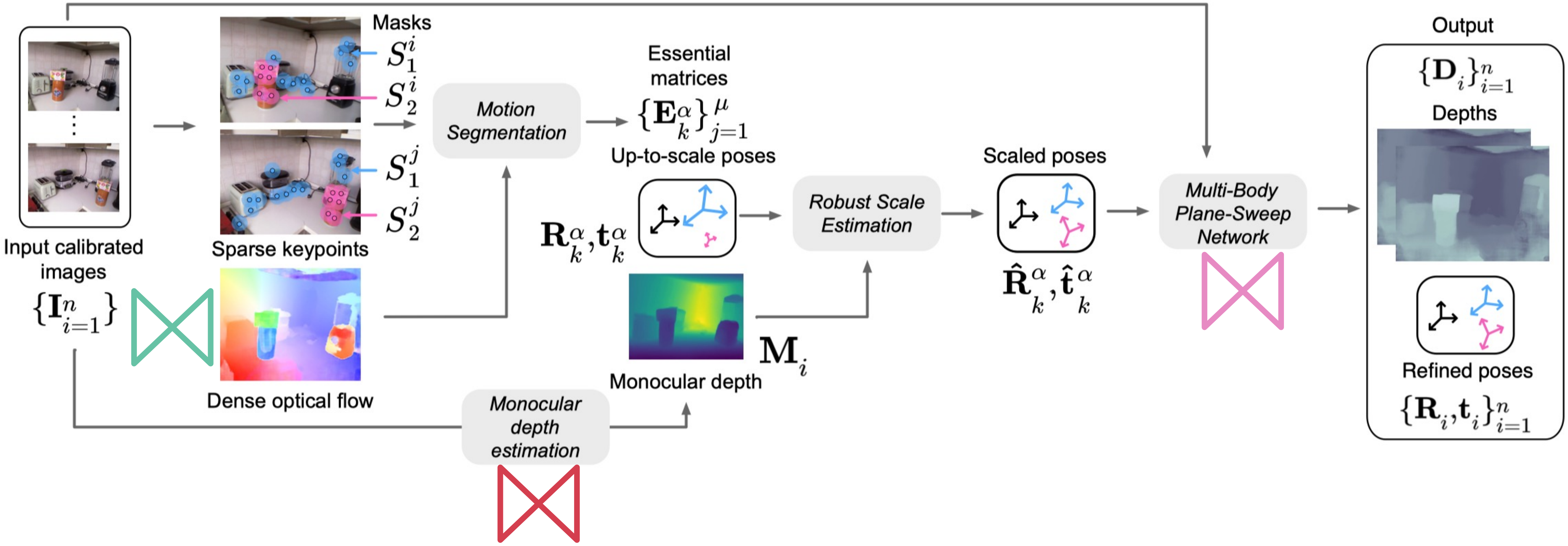


Multi-body plane sweep

This time, each pixel can be projected in different ways according to all the relative motions involved in the dynamic scenes.
All scene motions are considered when constructing the depth cost volume

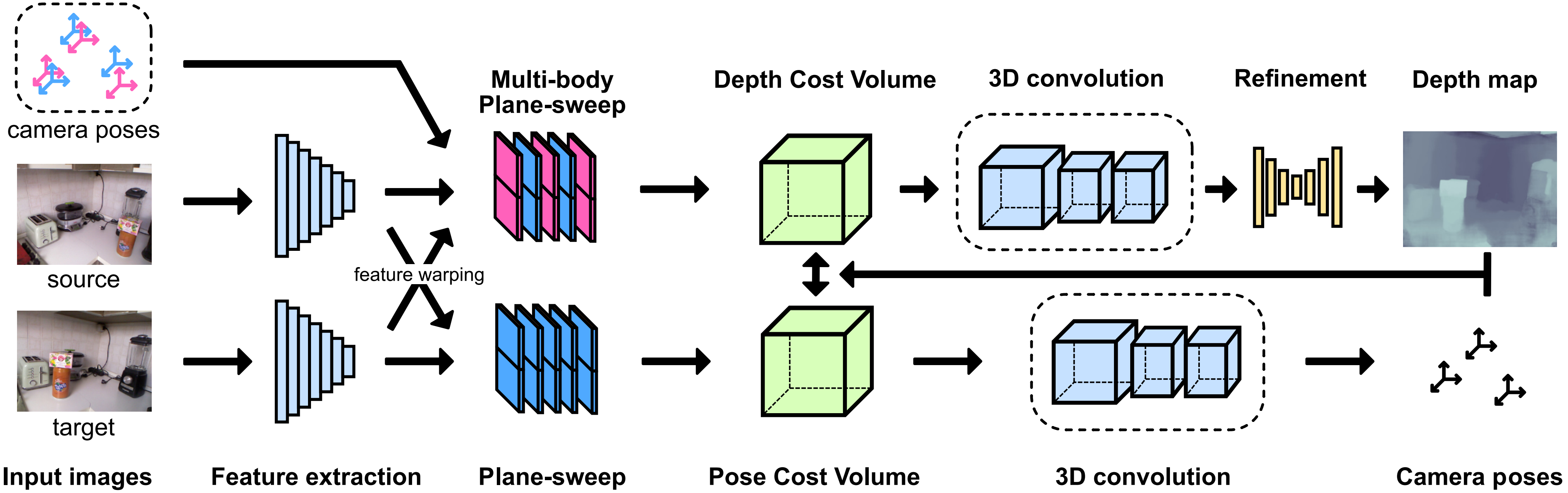


Multi-body plane sweep network



Multi-body plane sweep network

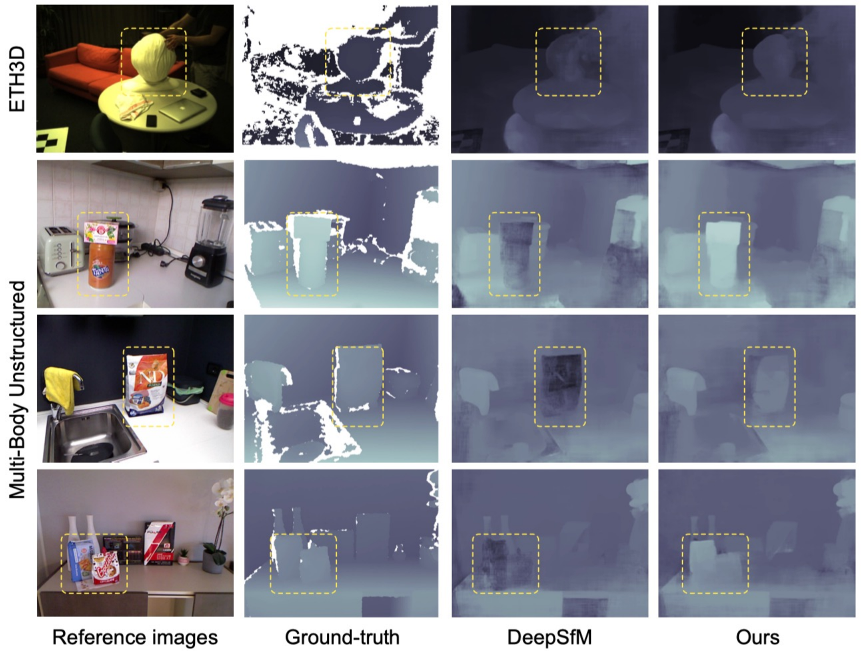
- Receives RGB images and correctly scaled camera poses as input
- Outputs dense depth maps and refined camera poses for each input image
- Includes our multi-body plane sweep algorithm to regress geometrically consistent depths

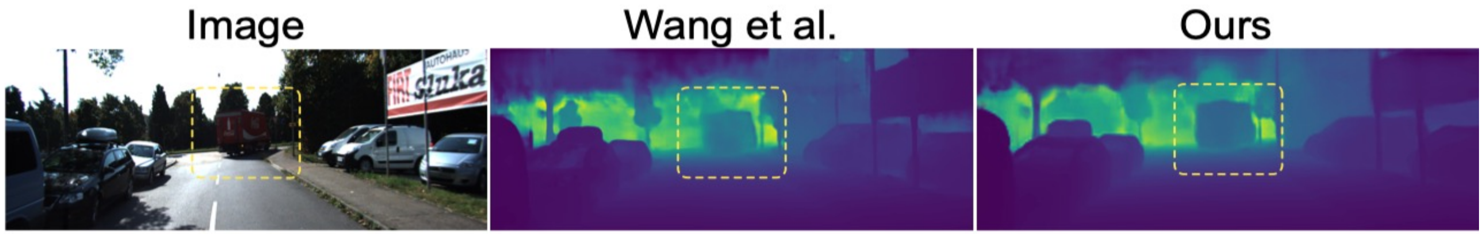


Multi-body plane sweep network

Qualitative comparisons on ETH3D and Multi-body Unstructured.

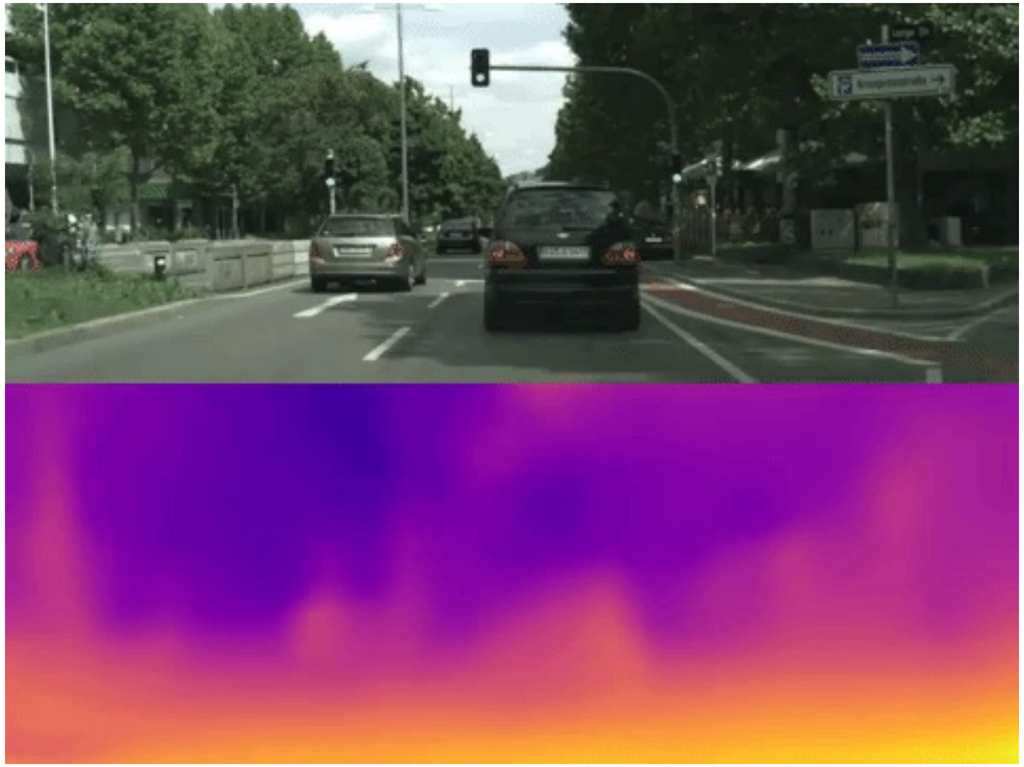
The yellow boxes highlight moving objects reconstructed by our method but not by the state-of-the-art DeepSfM





Monodepth training

- More abundant data since we can use video sequences
- Multiple viewpoint for reprojection improving the robustness
- Uniform region and moving object must be handled with care



To sum up

- Depth estimation from a single image is possible
- Compared to other tasks (e.g. object detection, semantic segmentation…) accurate manual annotation is unfeasible
- **Geometry come to rescue**: self-supervision is possible by exploiting stonger or weaker constraints…

What's next?

Monocular networks can still be easily fooled!
Although self-supervised techniques allow to increase the amount of training data with low effort, we are far from considering single image depth estimation to be solved.

Conversely to other task, such as Optical Flow and stereo, synthetic images have been rarely used, pre-trainig on synthetic samples and fine-tuning on the domain at hand could improve the results.

Even when 3D data are not needed for training, still they are needed for testing.