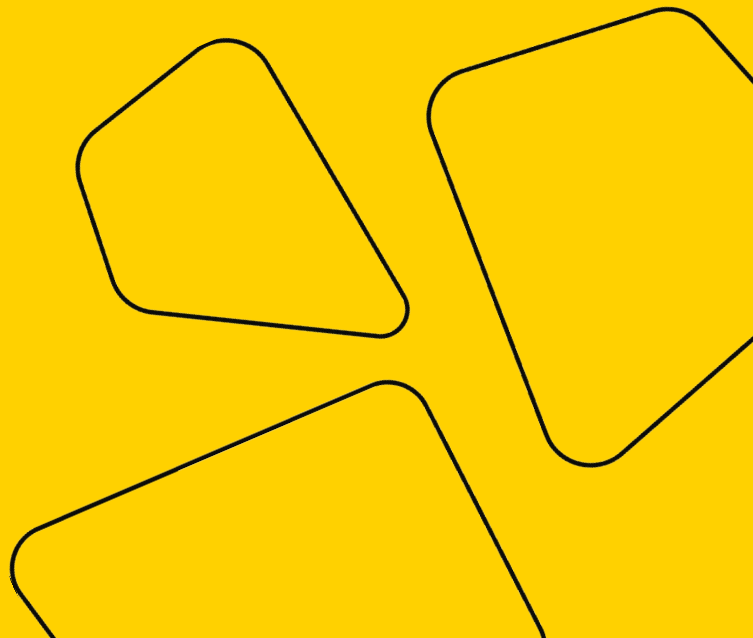
# Monocular Depth Estimation

Дмитрий Коршунов

EVOCARGO

girafe ai



# What is depth prediction?

Input video

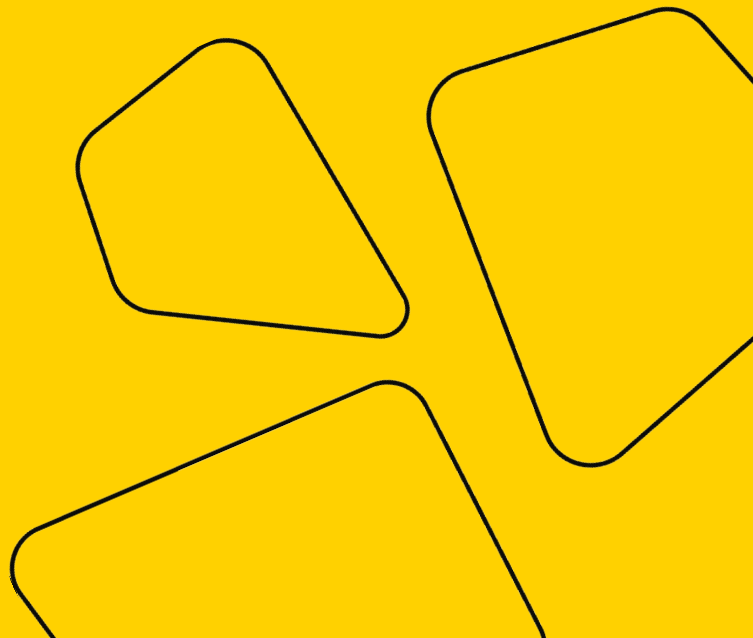
Our depth predictions

girafe ai

* https://ai.googleblog.com/2019/05/moving-camera-moving-people-deep.html

girafe
ai

# How can we estimate the depth?



# Static monocular

girafe ai

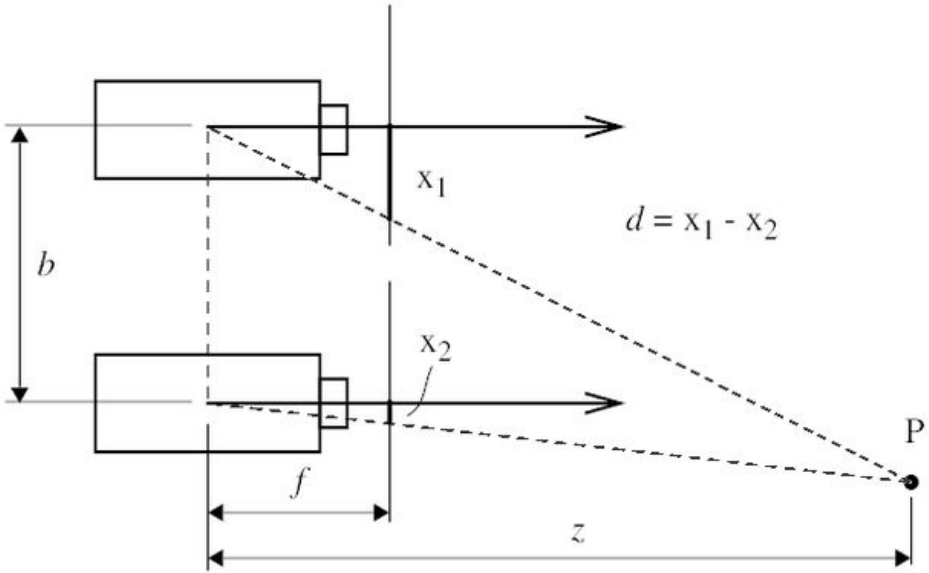
* Photo by Mateus Campos Felipe on Unsplash

# Depth from motion


girafe ai

* https://gfycat.com/ru/shabbycalculatingduckling

# Binocular



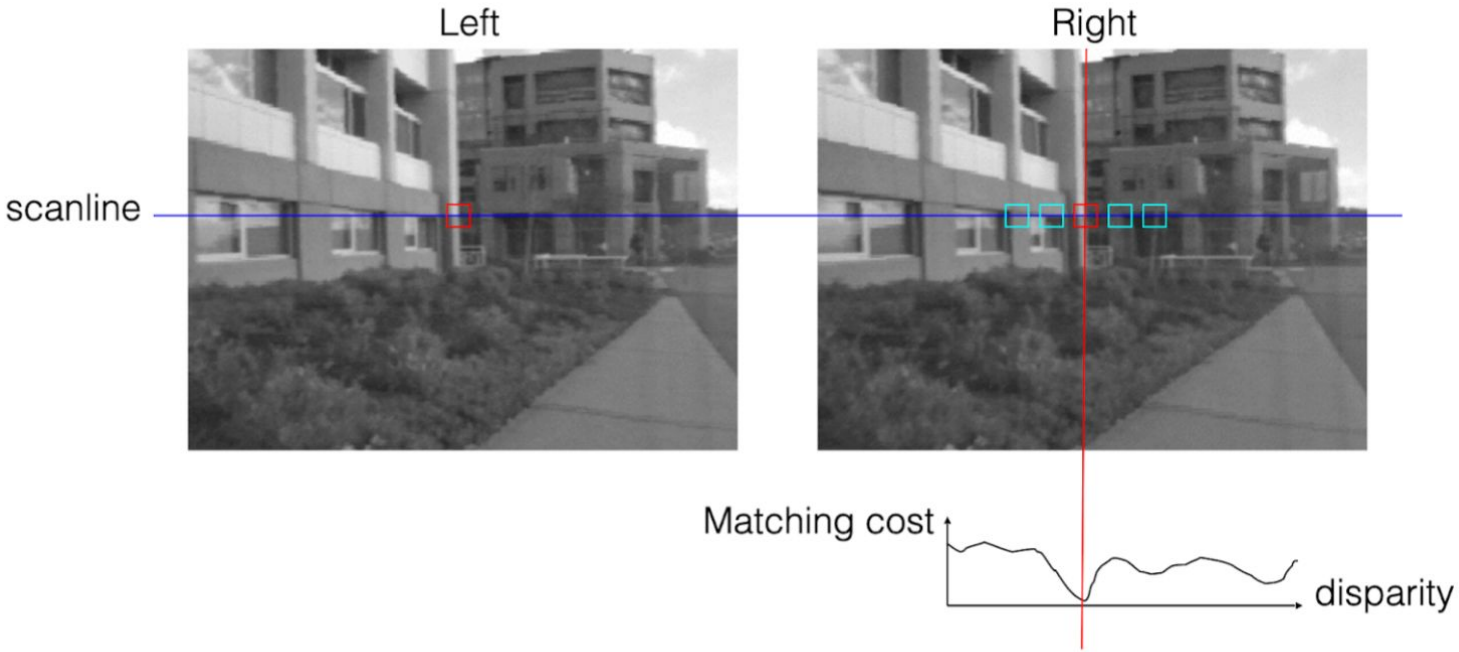

From similar triangles,

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

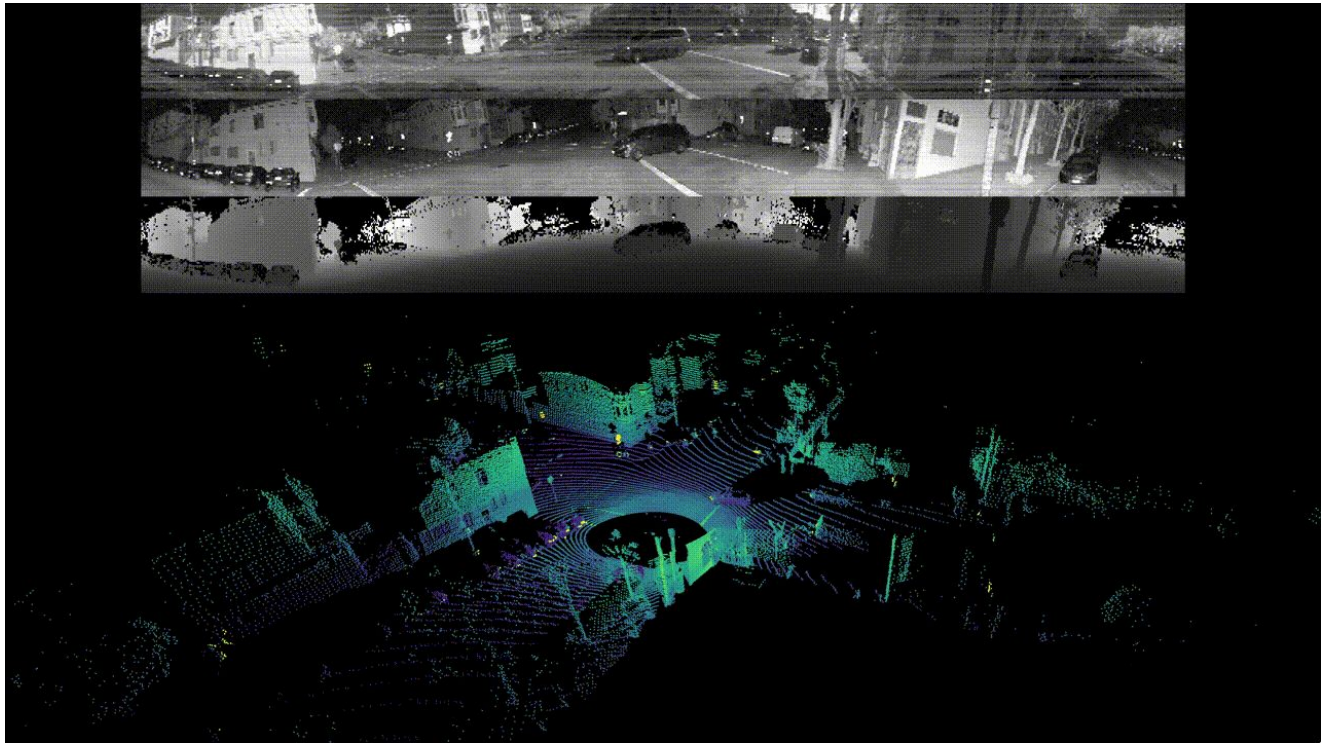
girafe ai

* https://towardsdatascience.com/depth-estimation-1-basics-and-intuition-86f2c9538cd1

# Binocular



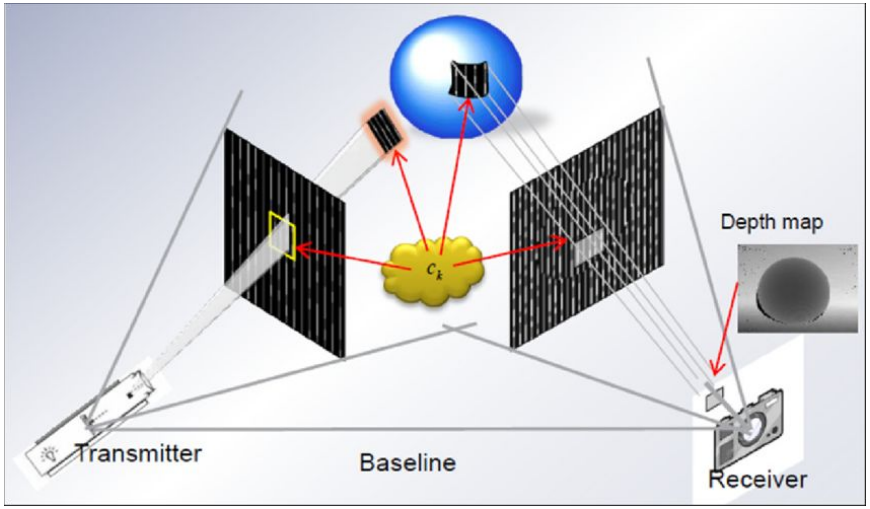
# Lidar

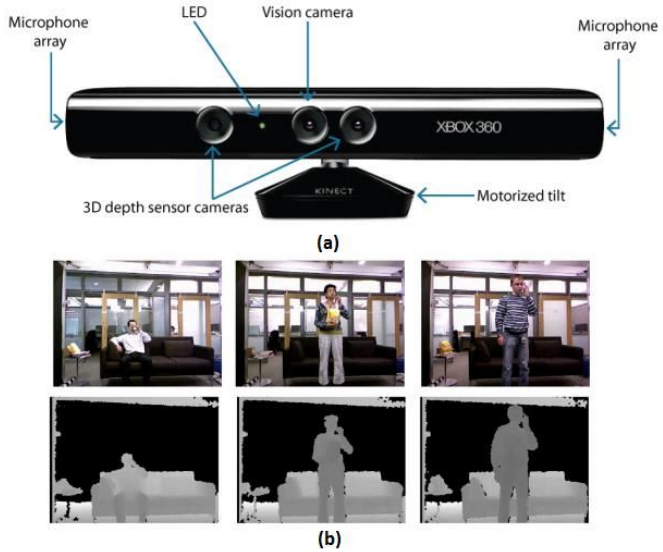


girafe ai

* https://ouster.com/blog/the-camera-is-in-the-lidar

# Structured-light camera

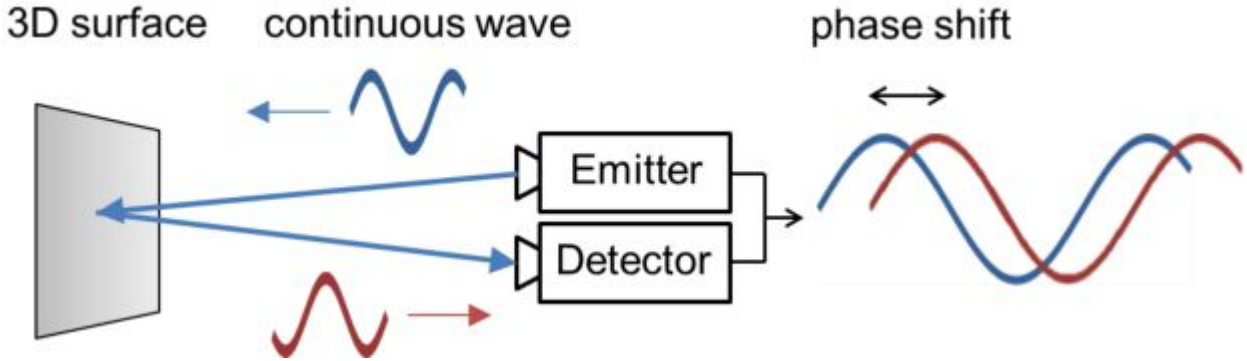



girafe ai

* https://www.researchgate.net/profile/Vikas-Ramachandra-2/publication/258813280/figure/fig1/AS:384165091528705@1468603503789/Structured-Light-System.png

# Time-of-flight camera




girafe ai

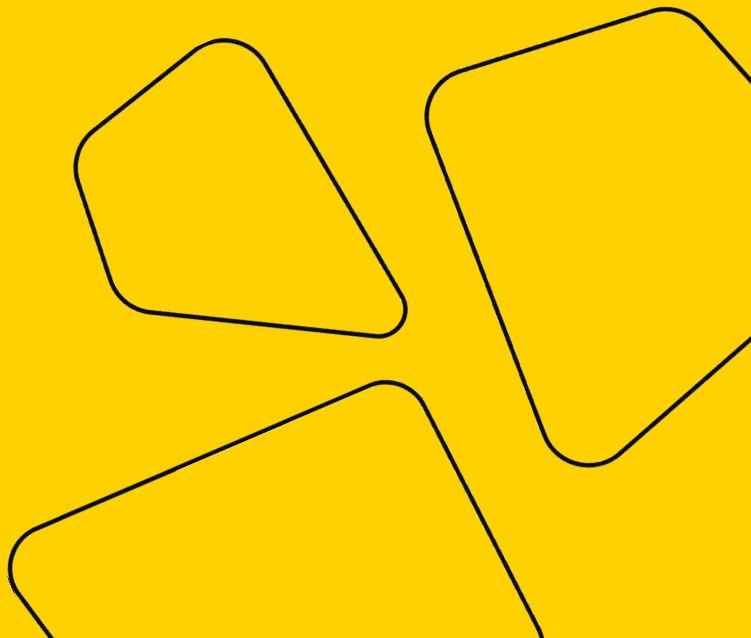
* https://www.researchgate.net/figure/Time-of-Flight-camera-principle_fig4_260432866

# Why monocular depth?

girafe ai

girafe ai

# Two types of depth predictions subtasks

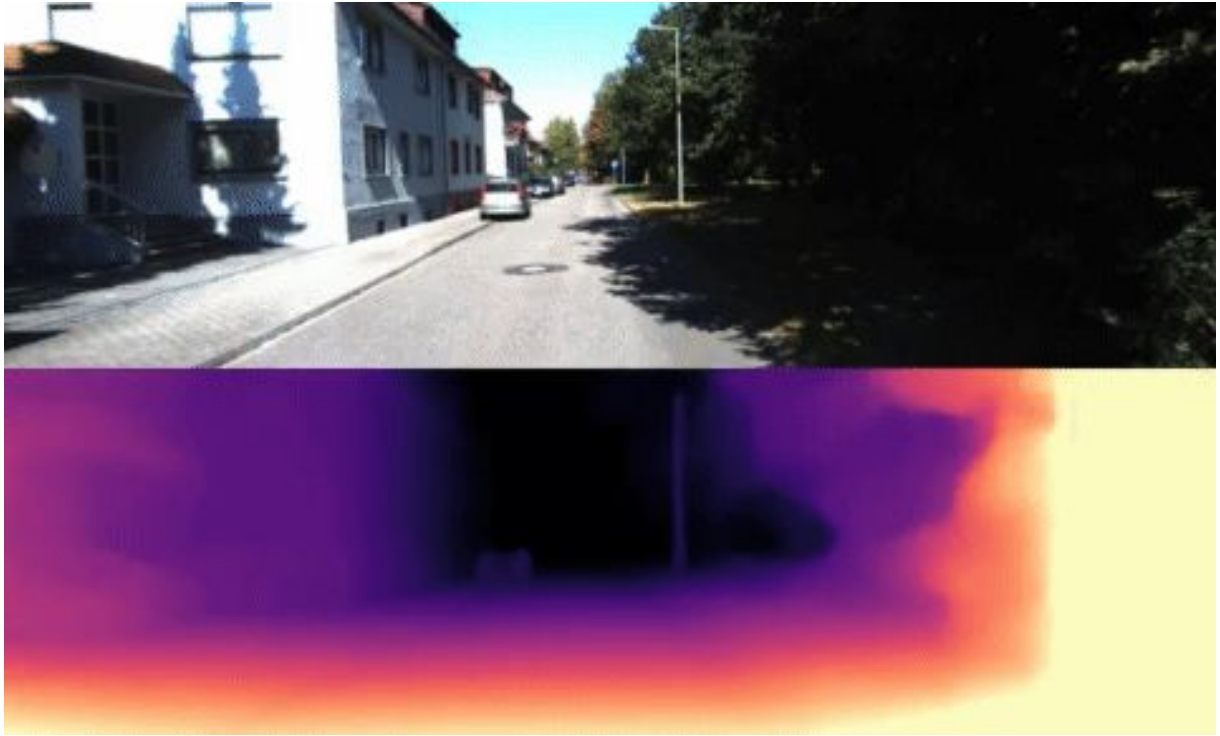


# Indoor

girafe ai

* https://github.com/facebookresearch/consistent_depth

# Outdoor



girafe ai

* https://github.com/nianticlabs/monodepth2

# Loss function

## Scale invariant logarithmic error (**SILog**)

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

$$d_i = \log y_i - \log y_i^*$$

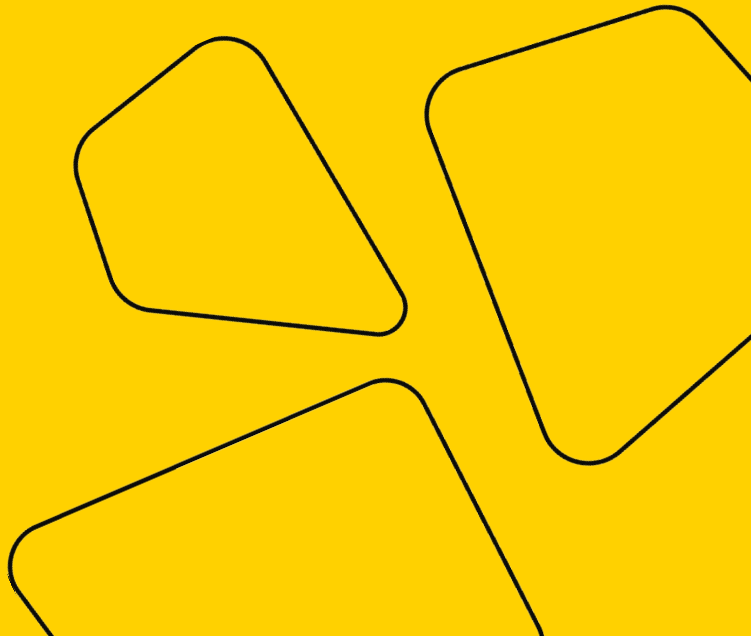
where $\lambda \in [0, 1]$. Note the output of the network is log y; that is, the final linear layer predicts the log depth. Setting $\lambda = 0$ reduces to elementwise l2, while $\lambda = 1$ is the scale-invariant error exactly.

n - quantity of pixels on image.

* https://papers.nips.cc/paper/2014/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf

girafe ai

# Key metrics



# AbsRel

(abs relative difference)

$$\frac{1}{|T|}\sum_{y\in T}|y-y^*|/y^*$$

* https://papers.nips.cc/paper/2014/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf

# δ1

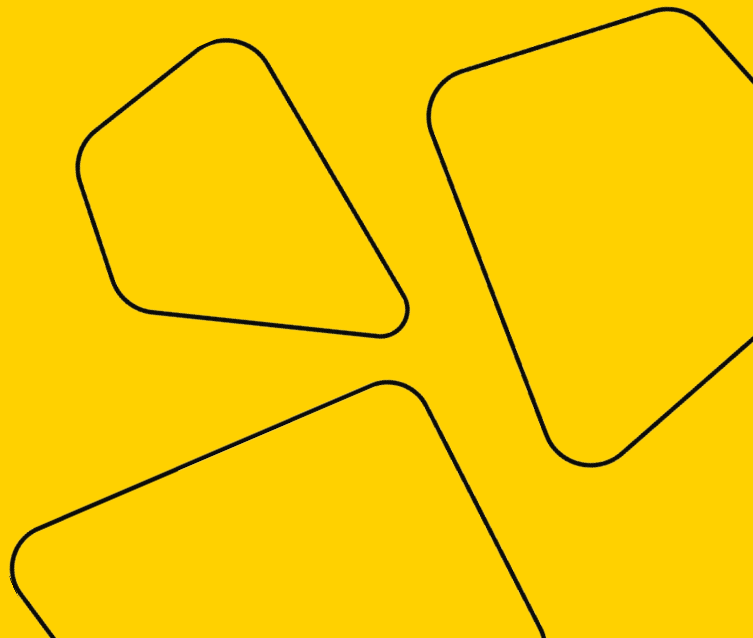
(threshold δ < 1.25)

$$\% \text{ of } y_i \text{ s.t. } \max\left(\frac{y_i}{y_i^*}, \frac{y_i^*}{y_i}\right) = \delta < thr$$

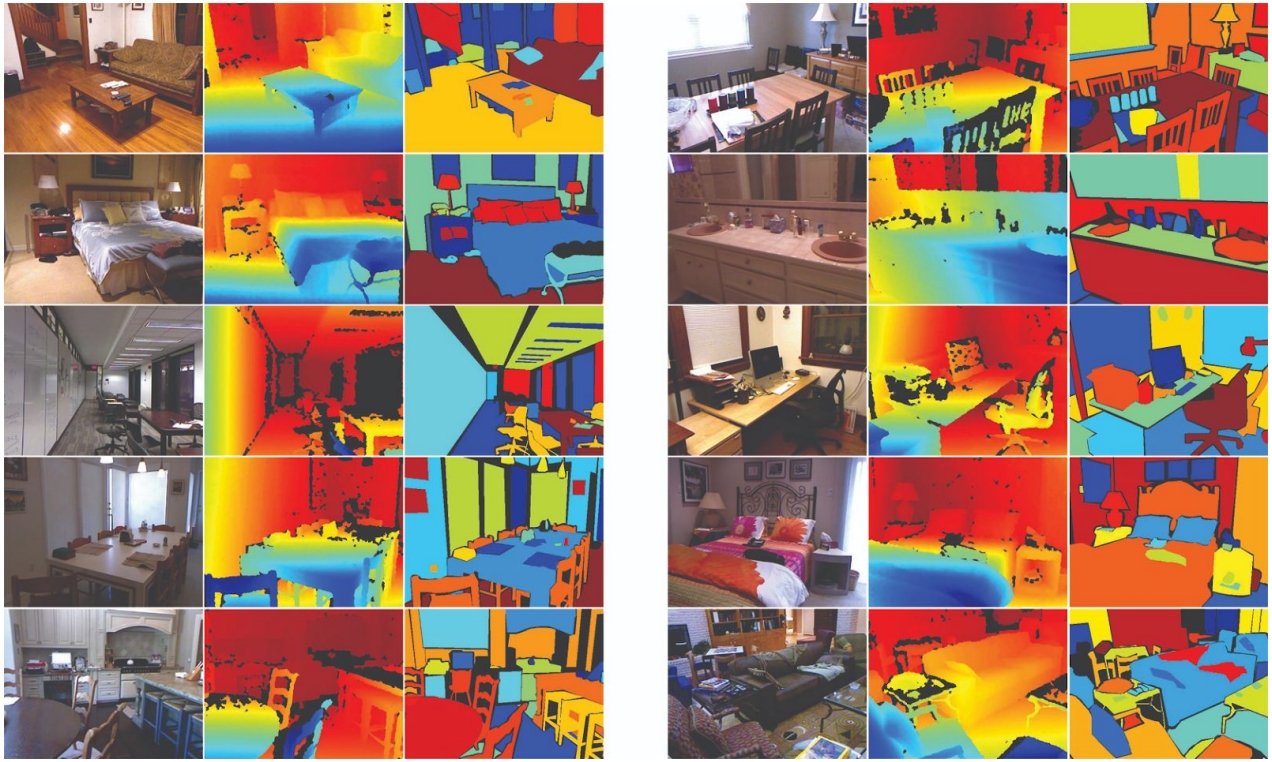
* https://papers.nips.cc/paper/2014/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf

girafe
ai

# Benchmark datasets



# NYU Depth Dataset V2



girafe ai

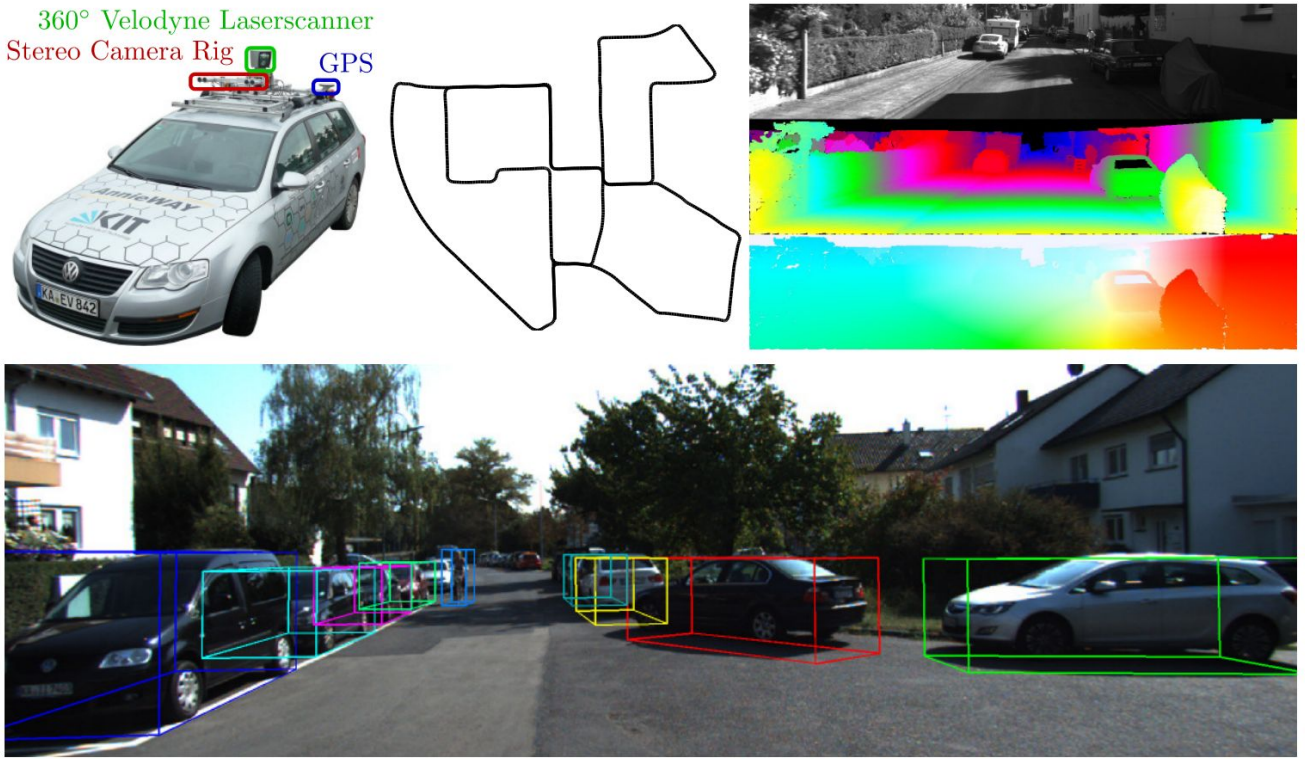
* https://cs.nyu.edu/~silberman/papers/indoor_seg_support.pdf

# General info

- Date - **2012**
- Place - **New York**
- Type - **Indoor**
- Samples - **1449**
- Size - **2.8 GB**
- Depth data source - **Kinect**

* http://www.cvlibs.net/publications/Geiger2013IJRR.pdf

# The KITTI Dataset




girafe ai

* http://www.cvlibs.net/publications/Geiger2012CVPR.pdf

# General info

- Date - **2013**
- Place - **Europe**
- Type - **Outdoor**
- Samples - **93k**
- Size - **21 GB**
- Depth data source - **Lidar**

* http://www.cvlibs.net/publications/Geiger2013IJRR.pdf

# The KITTI diversity

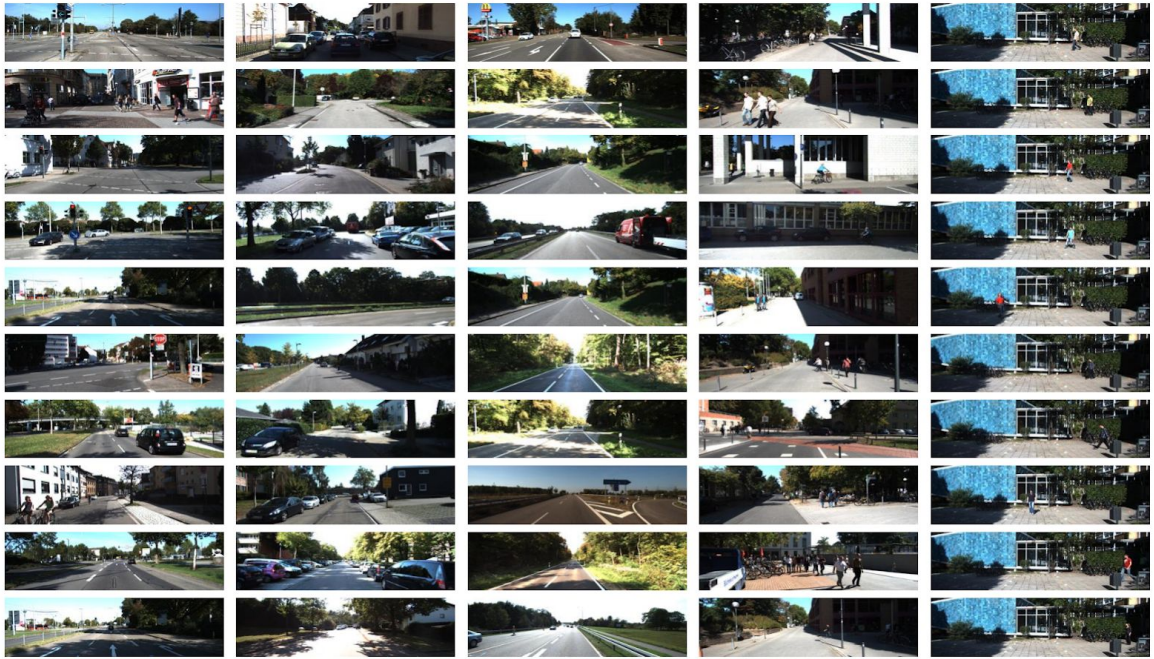



Fig. 5. **Examples from the KITTI dataset.** This figure demonstrates the diversity in our dataset. The left color camera image is shown.

girafe ai

* http://www.cvlibs.net/publications/Geiger2013IJRR.pdf

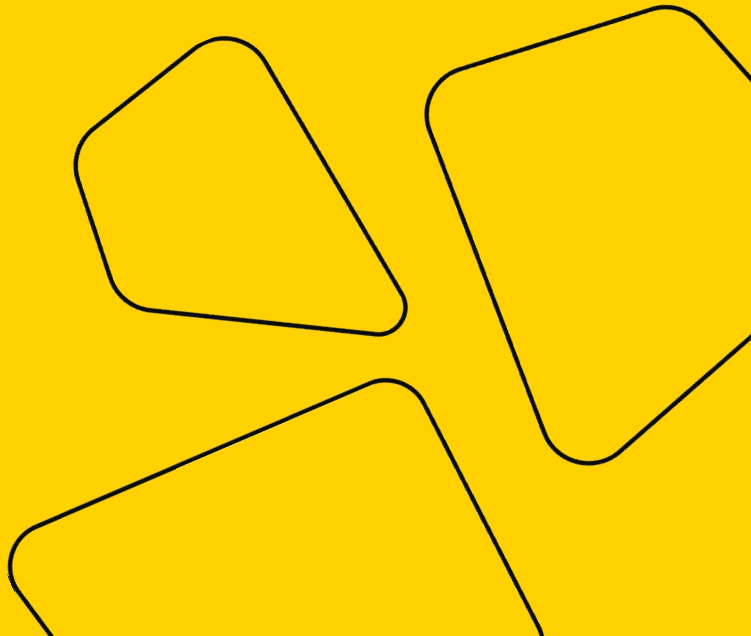
# The KITTI depth example

girafe ai

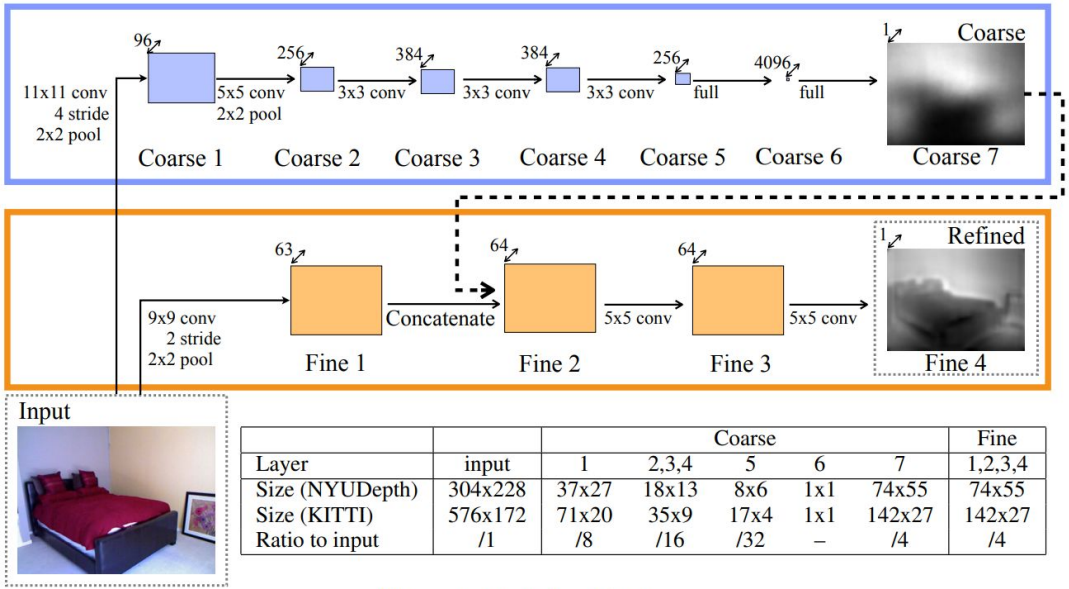
* http://www.cvlibs.net/publications/Geiger2013IJRR.pdf

girafe
ai

# Supervised



# Depth Map Prediction (2014)




| | | Coarse | | | | | Fine |
|---|---|---|---|---|---|---|---|
| Layer | input | 1 | 2,3,4 | 5 | 6 | 7 | 1,2,3,4 |
| Size (NYUDepth) | 304x228 | 37x27 | 18x13 | 8x6 | 1x1 | 74x55 | 74x55 |
| Size (KITTI) | 576x172 | 71x20 | 35x9 | 17x4 | 1x1 | 142x27 | 142x27 |
| Ratio to input | /1 | /8 | /16 | /32 | – | /4 | /4 |

Figure 1: Model architecture.

* https://papers.nips.cc/paper/2014/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf

# Interesting things

- One of the first CNN approach
- Custom scale invariant loss - SILog

# Deeper Depth Prediction (2016)

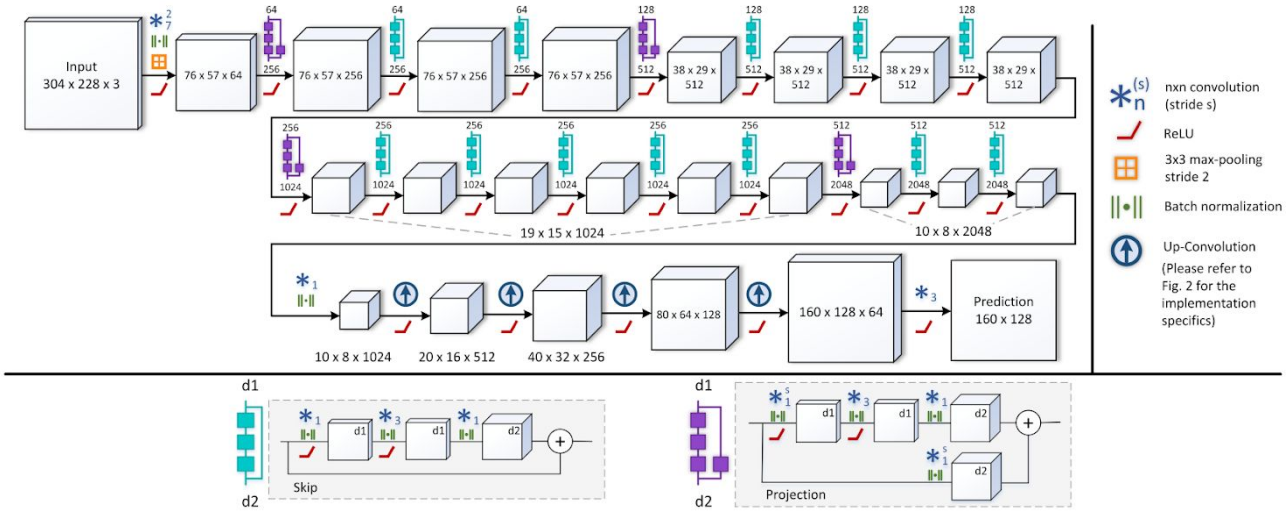



Figure 1. **Network architecture.** The proposed architecture builds upon ResNet-50. We replace the fully-connected layer, which was part of the original architecture, with our novel up-sampling blocks, yielding an output of roughly half the input resolution

girafe ai

* https://arxiv.org/pdf/1606.00373.pdf

# Interesting things

- Start using feature extractors (resnet-50 e.t.c)
- Using the reverse Huber (berHu)
- Using up-projections techniks

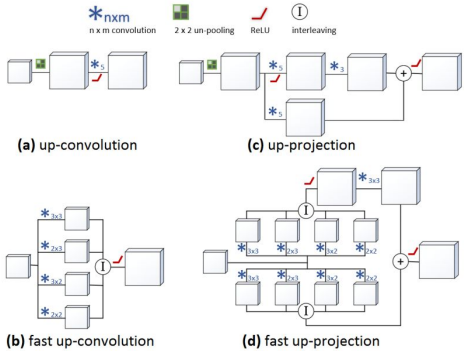



Figure 2. **From up-convolutions to up-projections. (a)** Standard up-convolution. **(b)** The equivalent but faster up-convolution. **(c)** Our novel up-projection block, following residual logic. **(d)** The faster equivalent version of (c)

# DORN (2018)

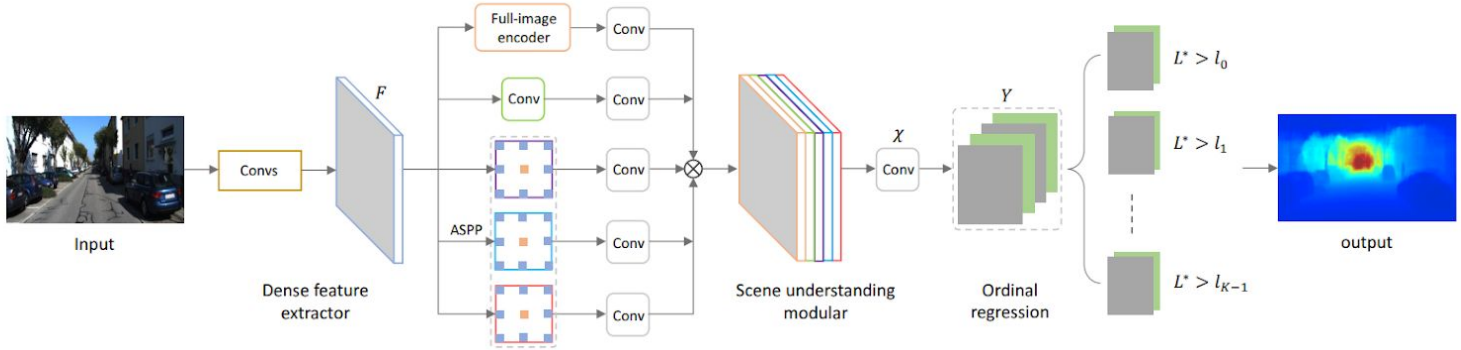



Figure 2: **Illustration of the network architecture.** The network consists of a dense feature extractor, multi-scale feature learner (*ASPP*), cross channel information learner (the pure $1 \times 1$ convolutional branch), a full-image encoder and an ordinal regression optimizer. The *Conv* components here are all with kernel size of $1 \times 1$. The *ASPP* module consists of 3 dilated convolutional layers with kernel size of $3 \times 3$ and dilated rate of 6, 12 and 18 respectively [6]. The supervised information of our network is discrete depth values output by the discretization using the *SID* strategy. The whole network is optimized by our ordinal regression training loss in an end-to-end fashion.

* https://arxiv.org/pdf/1806.02446v1.pdf

# Interesting things

- Change regression task to classification task
- Experiments with class quantity
- Custom ordinal loss
- ASPP module

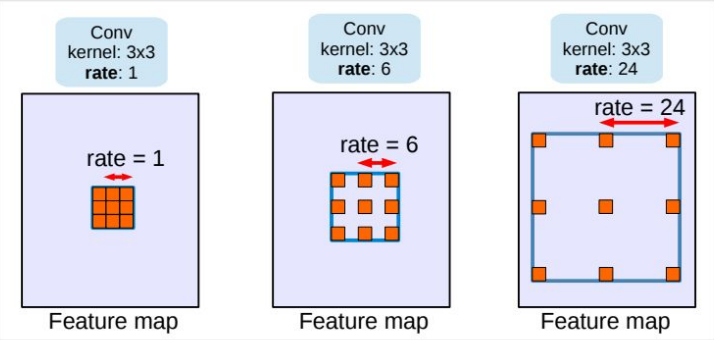



Figure 1. Atrous convolution with kernel size $3 \times 3$ and different rates. Standard convolution corresponds to atrous convolution with $rate = 1$. Employing large value of atrous rate enlarges the model's field-of-view, enabling object encoding at multiple scales.

* https://arxiv.org/pdf/1706.05587.pdf

# DPT-Large (2021)

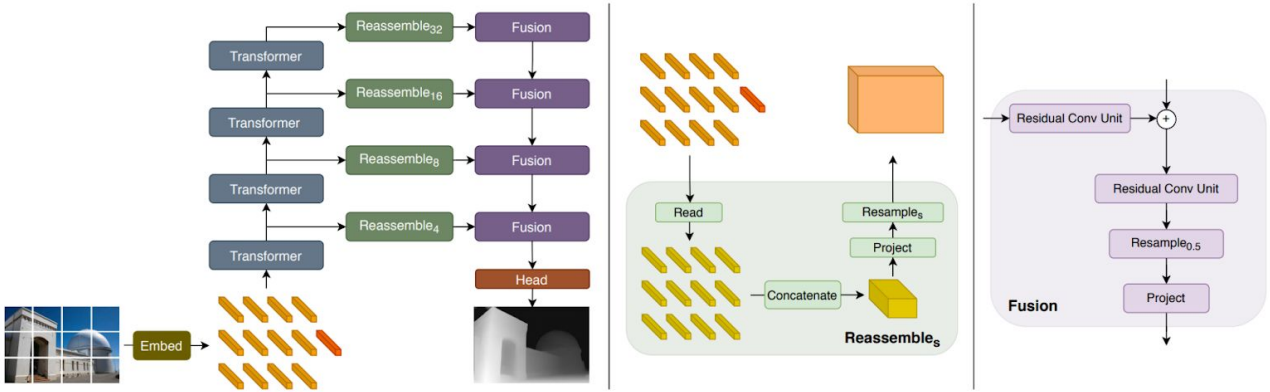



Figure 1. *Left*: Architecture overview. The input image is transformed into tokens (orange) either by extracting non-overlapping patches followed by a linear projection of their flattened representation (DPT-Base and DPT-Large) or by applying a ResNet-50 feature extractor (DPT-Hybrid). The image embedding is augmented with a positional embedding and a patch-independent readout token (red) is added. The tokens are passed through multiple transformer stages. We reassemble tokens from different stages into an image-like representation at multiple resolutions (green). Fusion modules (purple) progressively fuse and upsample the representations to generate a fine-grained prediction. *Center*: Overview of the Reassemble$_s$ operation. Tokens are assembled into feature maps with $\frac{1}{s}$ the spatial resolution of the input image. *Right*: Fusion blocks combine features using residual convolutional units [23] and upsample the feature maps.

girafe ai

* https://arxiv.org/pdf/2103.13413v1.pdf

# Interesting things

- Start transformer era in monodepth
- Using vision transformer as feature extraction
- **DPT-Large** has 343 million parameters and FPS* 29, but **DPT-Base** has 112 million parameters and FPS 59

* Intel Xeon Platinum 8280 CPU @ 2.70GHz with 8 physical cores and an Nvidia RTX 2080 GPU

# FastDepth (2019)

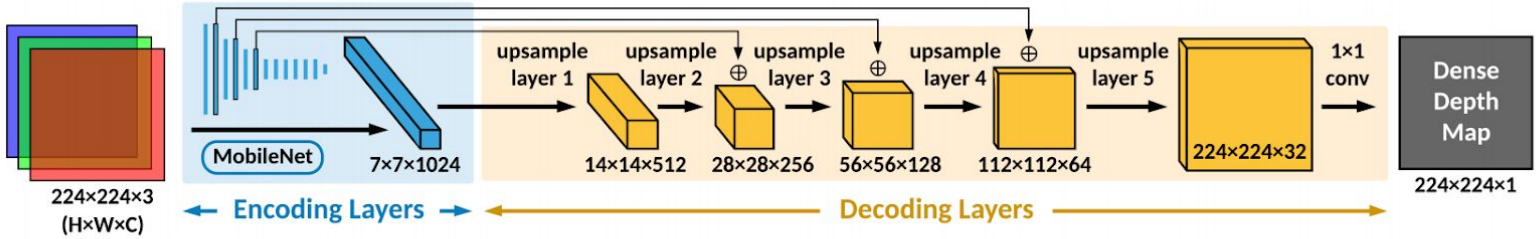
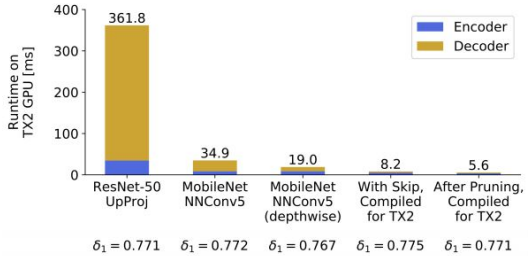
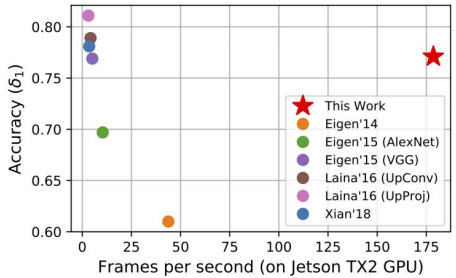



Fig. 2: Proposed network architecture. Dimensions of intermediate feature maps are given as height × width × # channels. Arrows from encoding layers to decoding layers denote additive (rather than concatenative) skip connections.




girafe ai

* https://arxiv.org/pdf/1903.03273.pdf

# Interesting things

- 178 fps on an NVIDIA Jetson TX2 GPU
- Novel NNConv5 as decoder

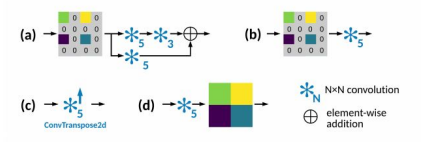
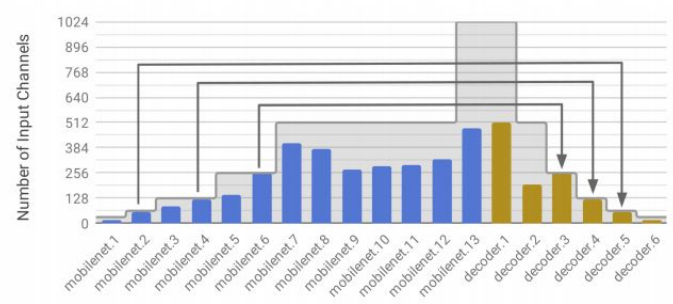



Fig. 5: Visual representations of different upsample operations. (a) UpProj, (b) UpConv, (c) DeConv5, (d) NNConv5.

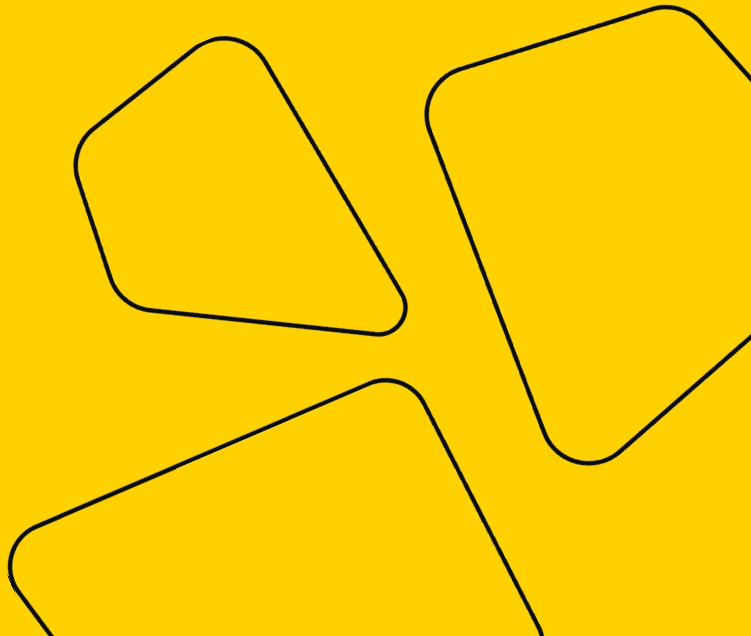
- Using NetAdapt, that automatically adapts neural network to a mobile platform




* https://arxiv.org/pdf/1804.03230.pdf

girafe ai

# Unsupervised



# Unsupervised Learning of Depth and Ego-Motion (2018)

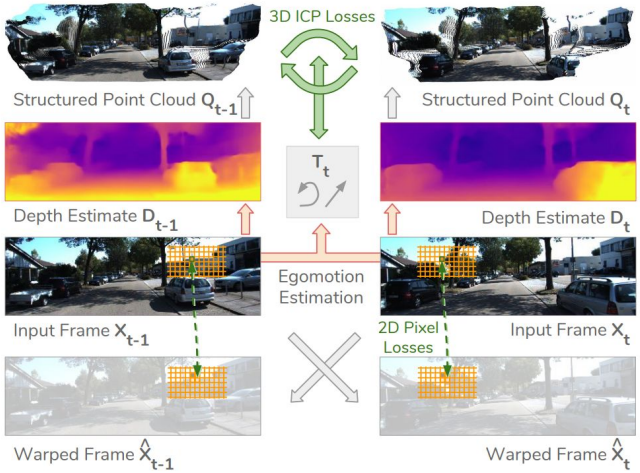



Figure 1. Overview of our method. In addition to 2D photometric losses, novel 3D geometric losses are used as supervision to adjust unsupervised depth and ego-motion estimates by the neural network. Orange arrows represent model's predictions. Gray arrows represent mechanical transformations. Green arrows represent losses. The depth images shown are sample outputs from our trained model.

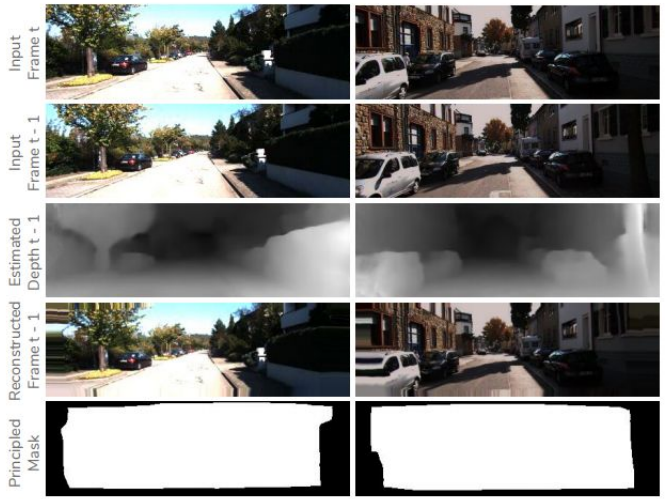



Figure 3. Principled Masks. The masks shown are examples of $M_{t-1}$, which indicate which pixel coordinates are valid when reconstructing $\hat{X}_{t-1}$ from $X_t$. There is a complementary set of masks $M_t$ (not shown), which indicate the valid pixel coordinates when reconstructing $\hat{X}_t$ from $X_{t-1}$.

girafe ai

* https://arxiv.org/pdf/1802.05522v2.pdf

# Interesting things

- Novel 3D geometric losses
- Principled Masks
- Rely on existence of ego-motion and consecutive frames

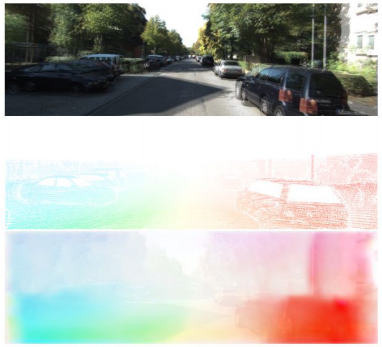


Figure 3: KITTI example. (top-to-bottom) Input image frames overlaid, groundtruth flow, and predicted flow from unsupervised network overlaid on the first input image.

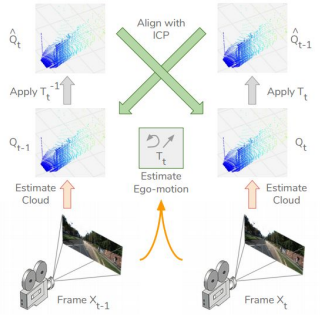
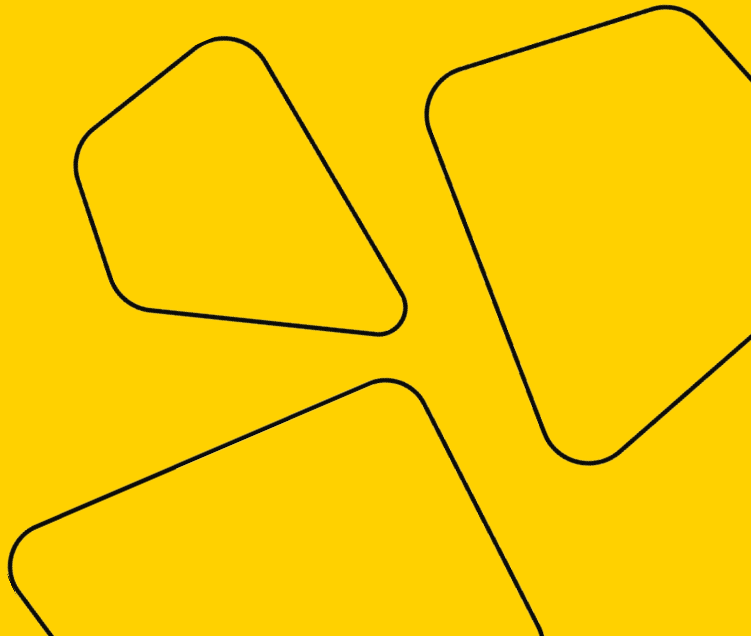



Figure 2. The 3D loss: ICP is applied symmetrically in forward and backward directions to bring the depth and ego-motion estimates from two consecutive frames into agreement. The products of ICP generate gradients which are used to improve the depth and ego-motion estimates.

* https://arxiv.org/pdf/1608.05842.pdf

girafe
ai

# Self-Supervised



# Monodepth2 (2019)

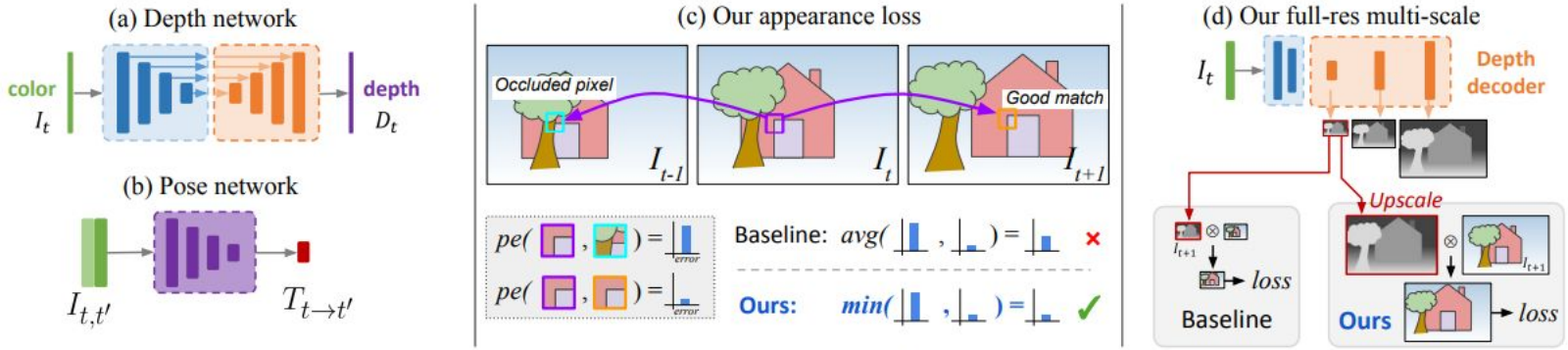



Figure 3. **Overview. (a) Depth network:** We use a standard, fully convolutional, U-Net to predict depth. **(b) Pose network:** Pose between a pair of frames is predicted with a separate pose network. **(c) Per-pixel minimum reprojection:** When correspondences are *good*, the reprojection loss should be *low*. However, occlusions and disocclusions result in pixels from the current time step not appearing in both the previous and next frames. The baseline *average* loss forces the network to match occluded pixels, whereas our *minimum reprojection* loss only matches each pixel to the view in which it is visible, leading to sharper results. **(d) Full-resolution multi-scale:** We upsample depth predictions at intermediate layers and compute all losses at the input resolution, reducing texture-copy artifacts.



* https://arxiv.org/pdf/1806.01260v4.pdf

# Interesting things

- minimum reprojection loss, designed to robustly handle occlusions
- auto-masking loss to ignore training pixels that violate camera motion assumptions

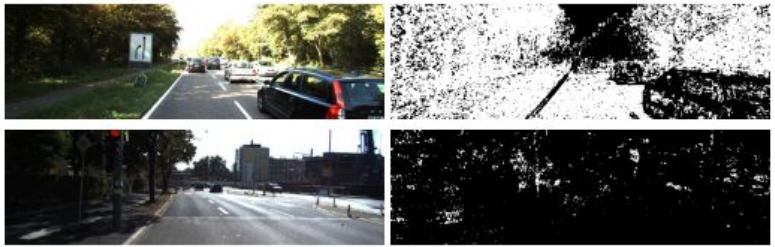
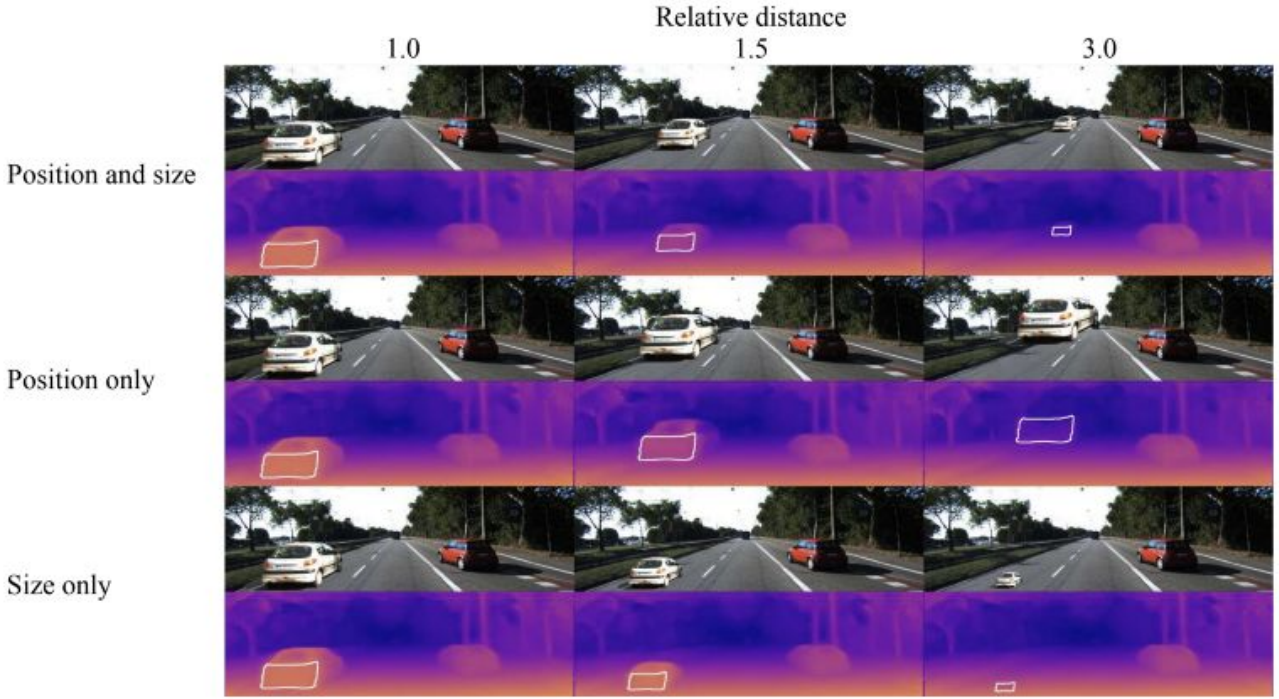


Figure 5. **Auto-masking.** We show auto-masks computed after one epoch, where black pixels are removed from the loss (*i.e.* $\mu = 0$). The mask prevents objects moving at similar speeds to the camera (top) and whole frames where the camera is static (bottom) from contaminating the loss. The mask is computed from the input frames and network predictions using Eqn. 5.

girafe ai

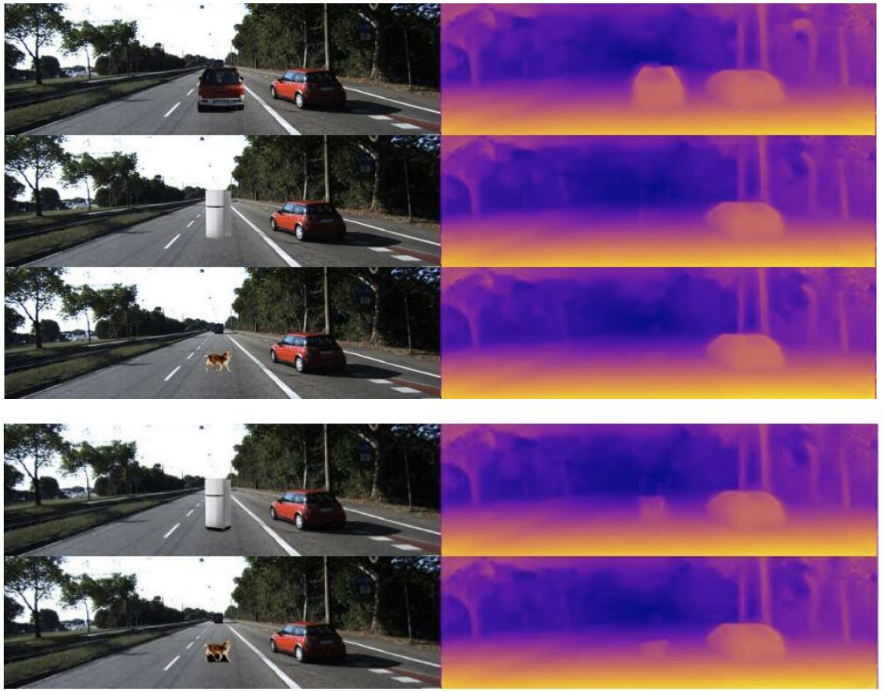
# How do neural networks see depth in single images?

# Position of objects provides strong contextual information




* https://openaccess.thecvf.com/content_ICCV_2019/papers/van_Dijk_How_Do_Neural_Networks_See_Depth_in_Single_Images_ICCV_2019_paper.pdf

# Shape does not matter but shadow does



girafe ai

* https://openaccess.thecvf.com/content_ICCV_2019/papers/van_Dijk_How_Do_Neural_Networks_See_Depth_in_Single_Images_ICCV_2019_paper.pdf

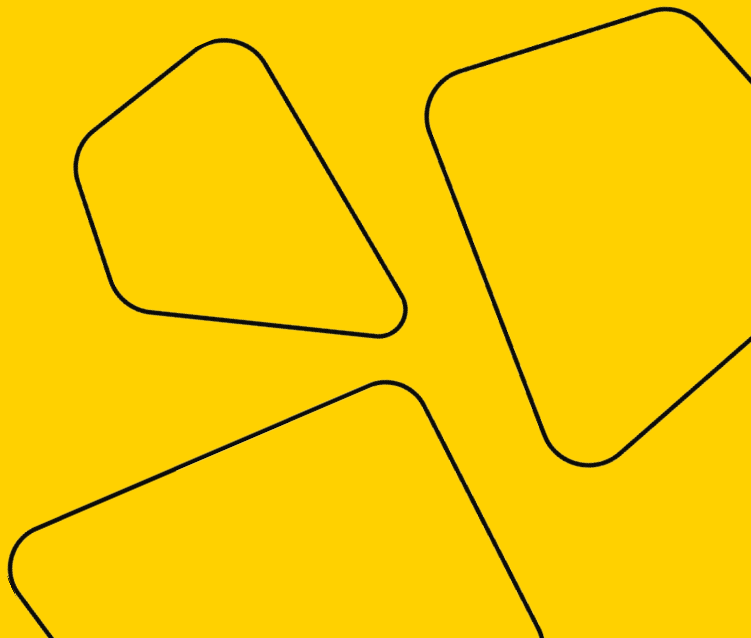
# The lower part of the image is always closed to the camera

girafe ai

* https://arxiv.org/pdf/1907.01341.pdf

girafe ai

# Why is measuring depth so difficult?



# Vision has to solve an ill-posed problem

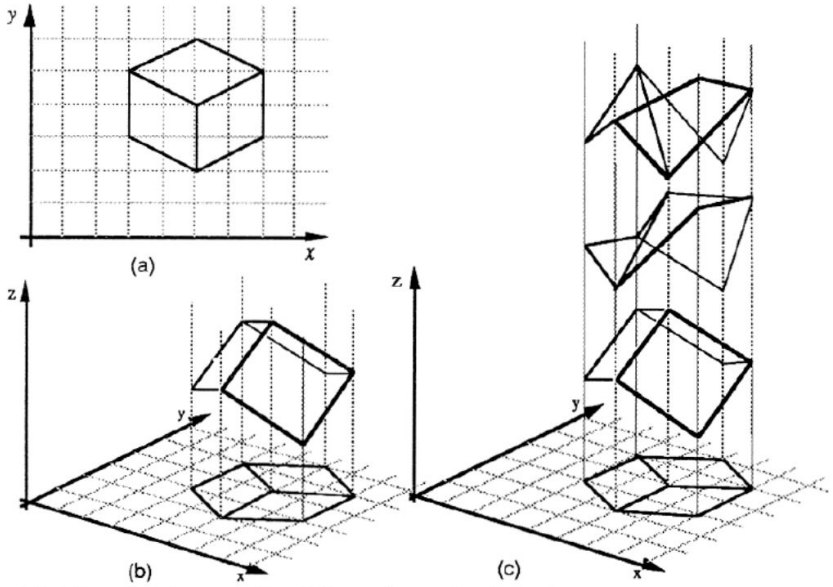


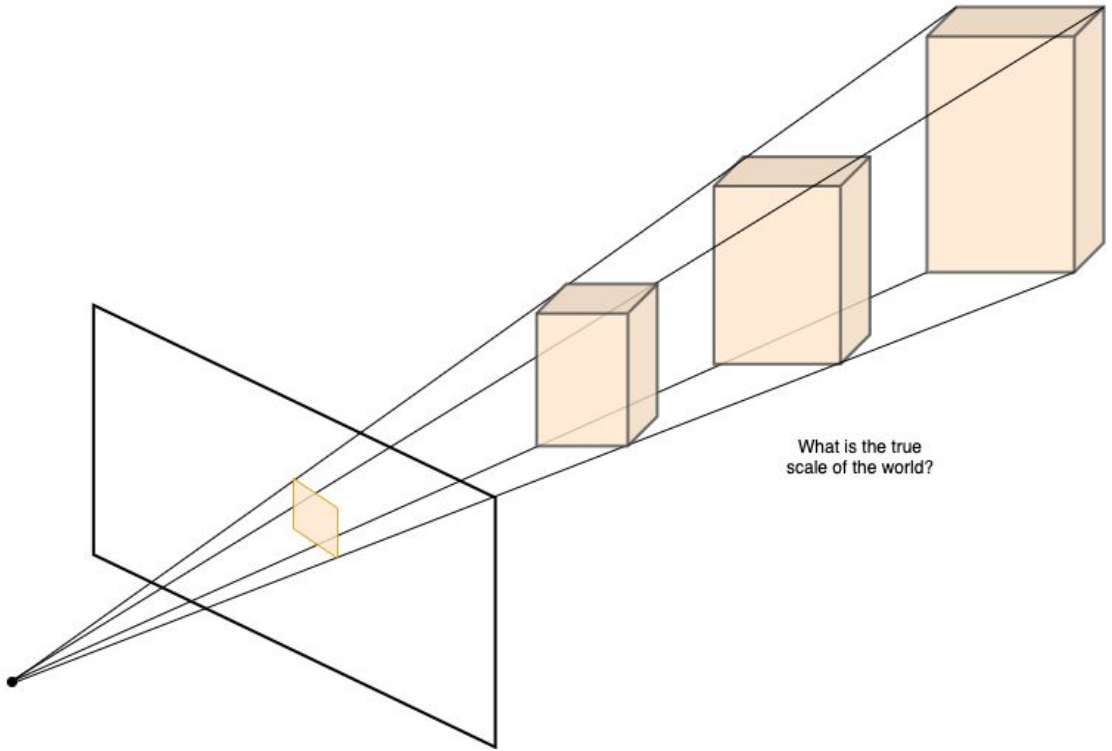

*Figure 1. (a) A line drawing provides information only about the x, y coordinates of points lying along the object contours. (b) The human visual system is usually able to reconstruct an object in three dimensions given only a single 2D projection (c) Any planar line-drawing is geometrically consistent with infinitely many 3D structures.*

Sinha & Adelson 93 Dr Chris Town

girafe ai

* https://www.cl.cam.ac.uk/teaching/1011/CompVision/Town%20-%20ComputerVision%20-%20L1.pdf

# Scale ambiguity




girafe ai

* https://towardsdatascience.com/depth-estimation-1-basics-and-intuition-86f2c9538cd1

girafe
ai

# Thank you!