Stereo Vision and Depth Prediction



Antonio Carta

2026-10-12

University of Pisa

antonio.carta@unipi.it

Introduction

Objectives

- Perspective cues in single images
- Depth prediction with stereo vision: rectification, matching, triangulation
- Depth prediction in DNNs

Motivation

- A single photo is **depth-ambiguous**: pixels can match any point in the corresponding 3D ray
- **depth cues**: converging lines, texture density, occlusion, haze
- A **second camera** adds a geometric constraint: the rays of the two views intersect in a 3D point



Figure 1: Railroad tracks: two parallel 3D lines whose images converge to a single point. Source: M. Malak, public domain, Wikimedia Commons

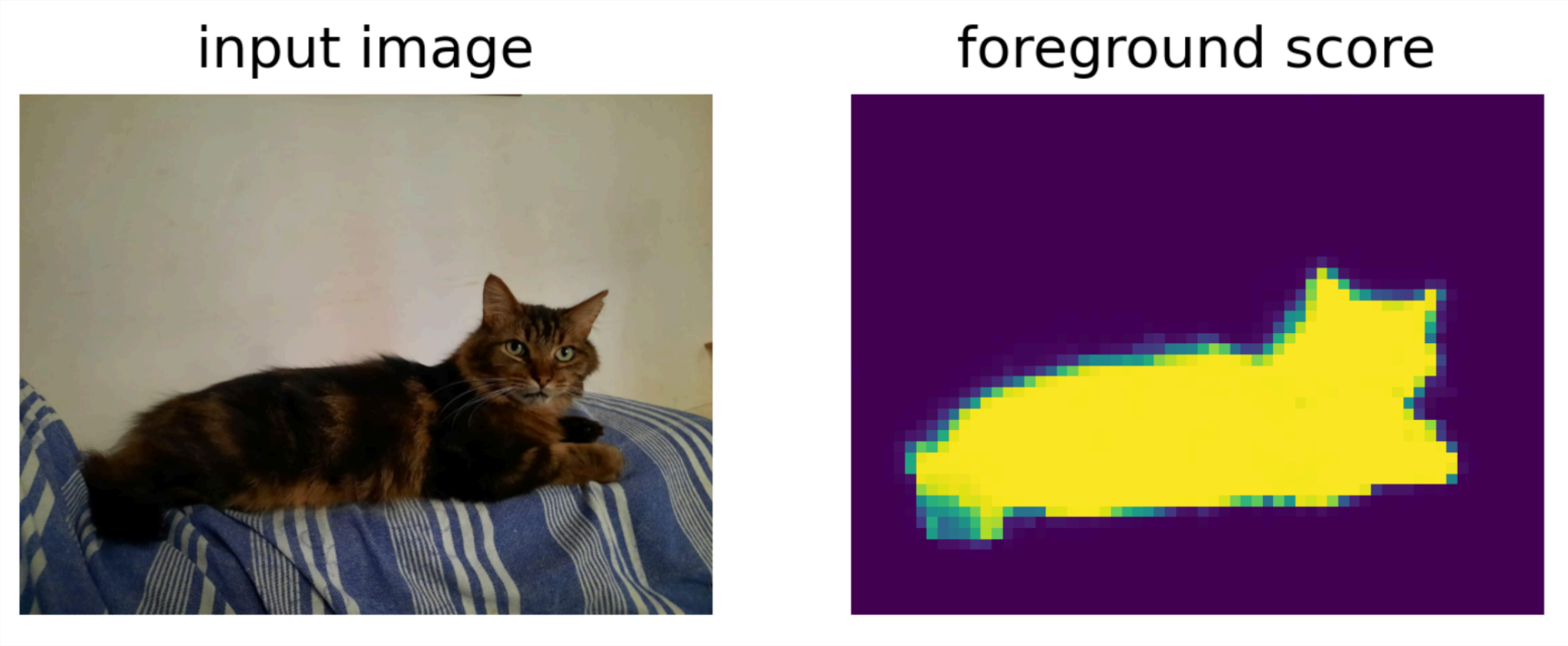


Figure 2: Foreground detection with DINOv3. Depth cues and prior knowledge allow to infer depth even from a single view.

"https://github.com/facebookresearch/dinov3/blob/main/notebooks/foreground_segmentation.ipynb"

Perspective Cues

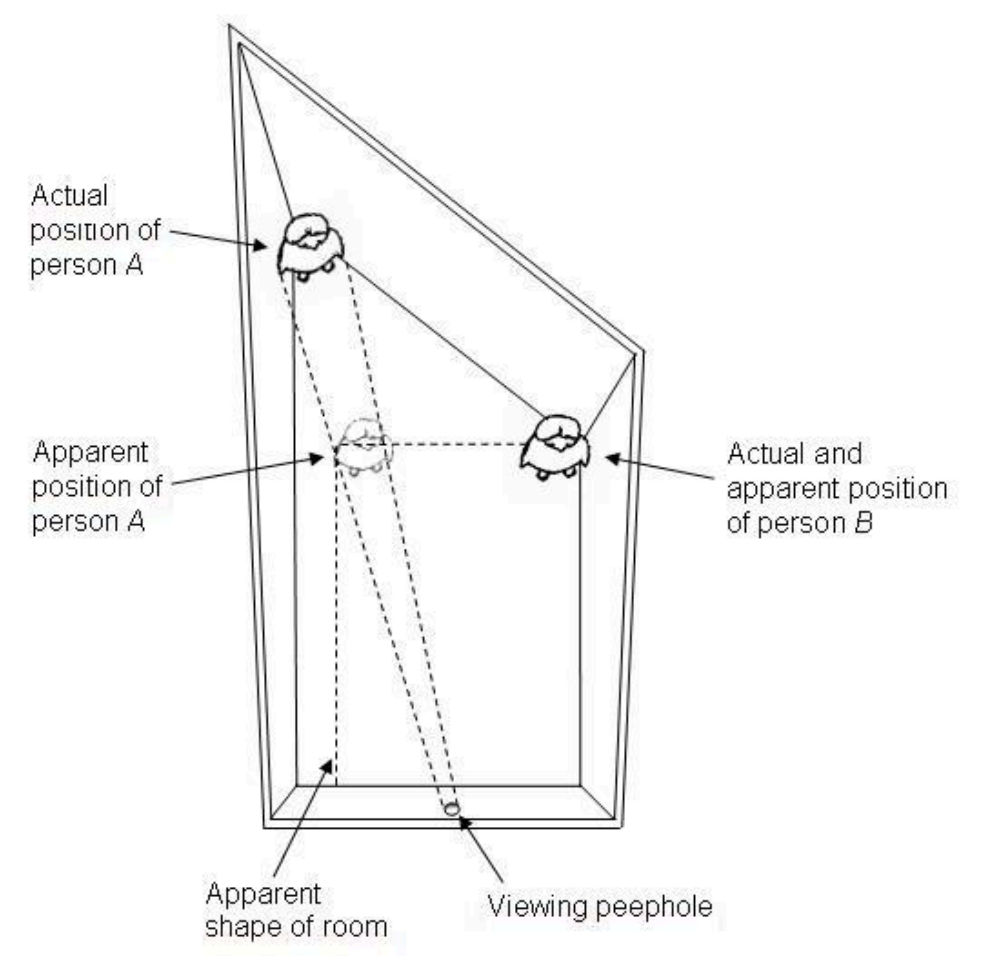


Figure 3: true vs. apparent room shape, source: Wikimedia



Figure 4: an Ames room in use, Wikimedia

Perspective Cues

- Perspective cues are assumptions and biases (like planarity/rectangularity) that generally work in natural images
- The **Ames room** is a distorted trapezoidal room, built so that (the distorted) floor, ceiling, and back wall project to the same image as an ordinary rectangular room when viewed from one specific peephole
- We assume rectangularity and infer the wrong 3D geometry

Monocular Depth Cues

Depth Perception in Humans

- Depth is not measured by the eye, which is a pinhole camera (2D projection)
- The visual system reconstructs 3D structure from
  - **single image** depth cues
  - **motion** cues
  - **binocular** vision

In CV systems:

- we need to understand when single image cues work
- we need to formalize geometry constraints for multiple views
- motion is often formalized as a special kind of multiple views (in time)

Pictorial Depth Cues: Lines

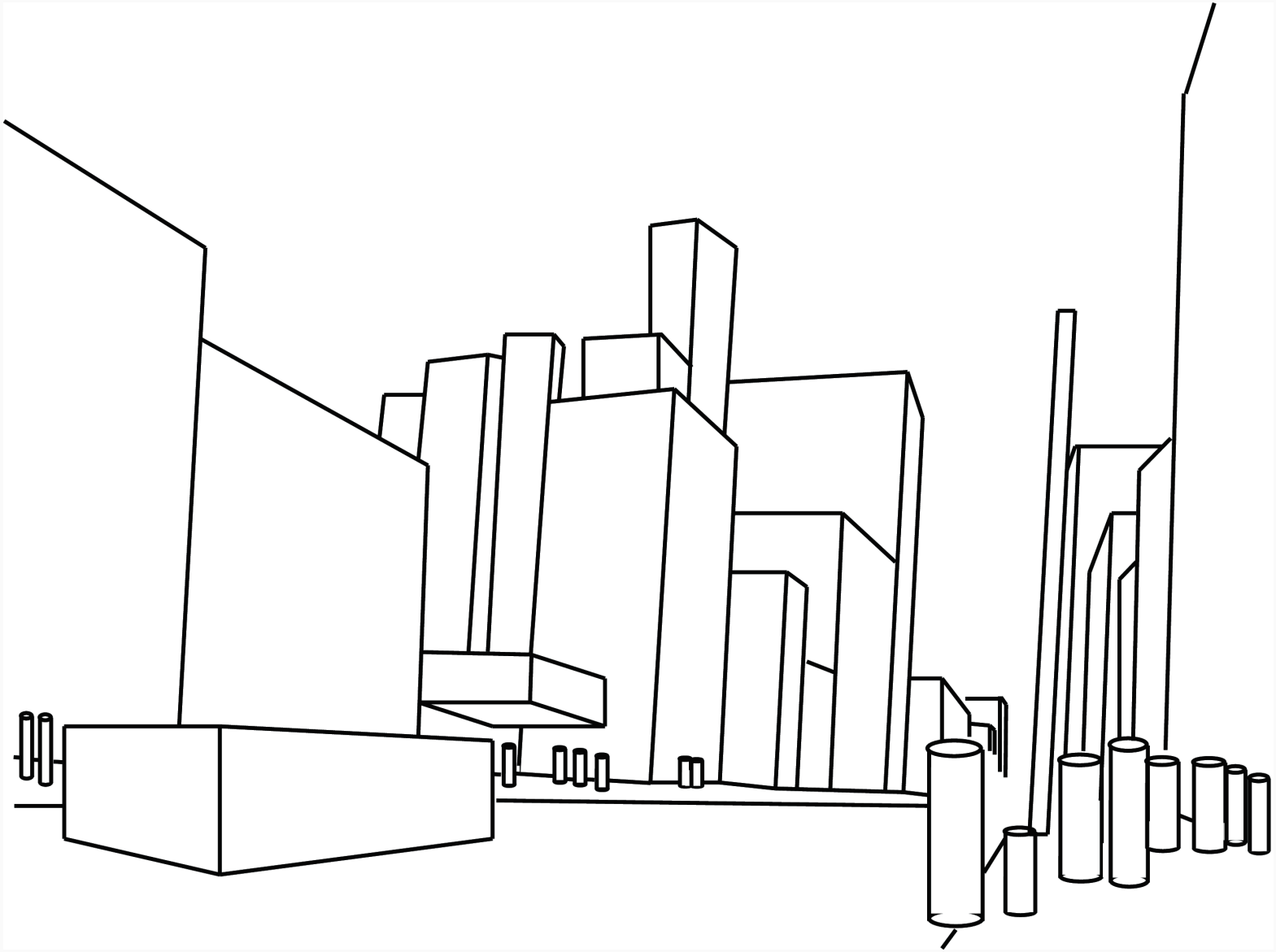


Figure 5: source: visionbook

Pictorial Depth Cues: Lines

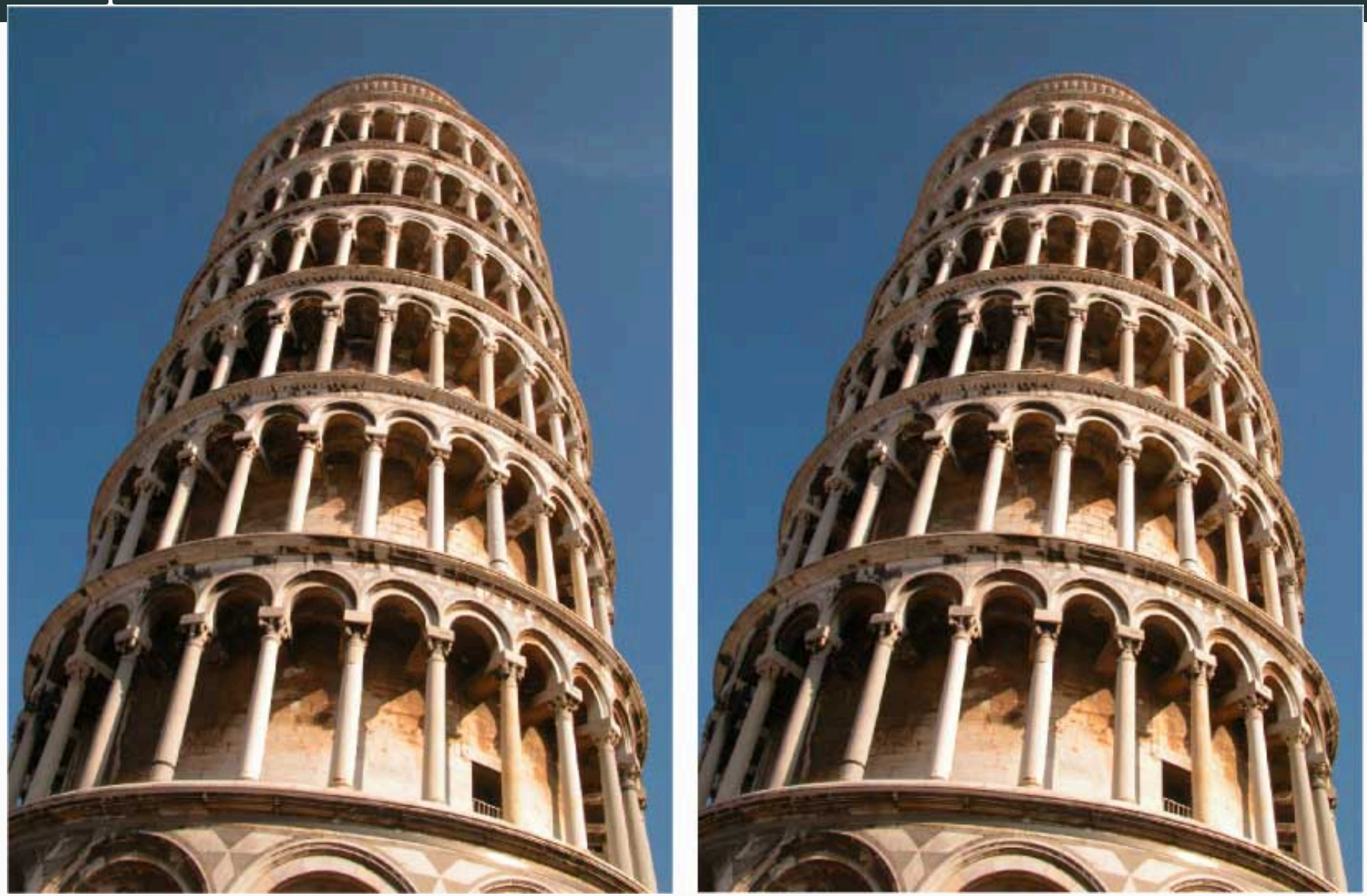


Figure 6: The picture on the right appears to be more tilted than the one on the left. source: visionbook

Pictorial Depth Cues: Perspective

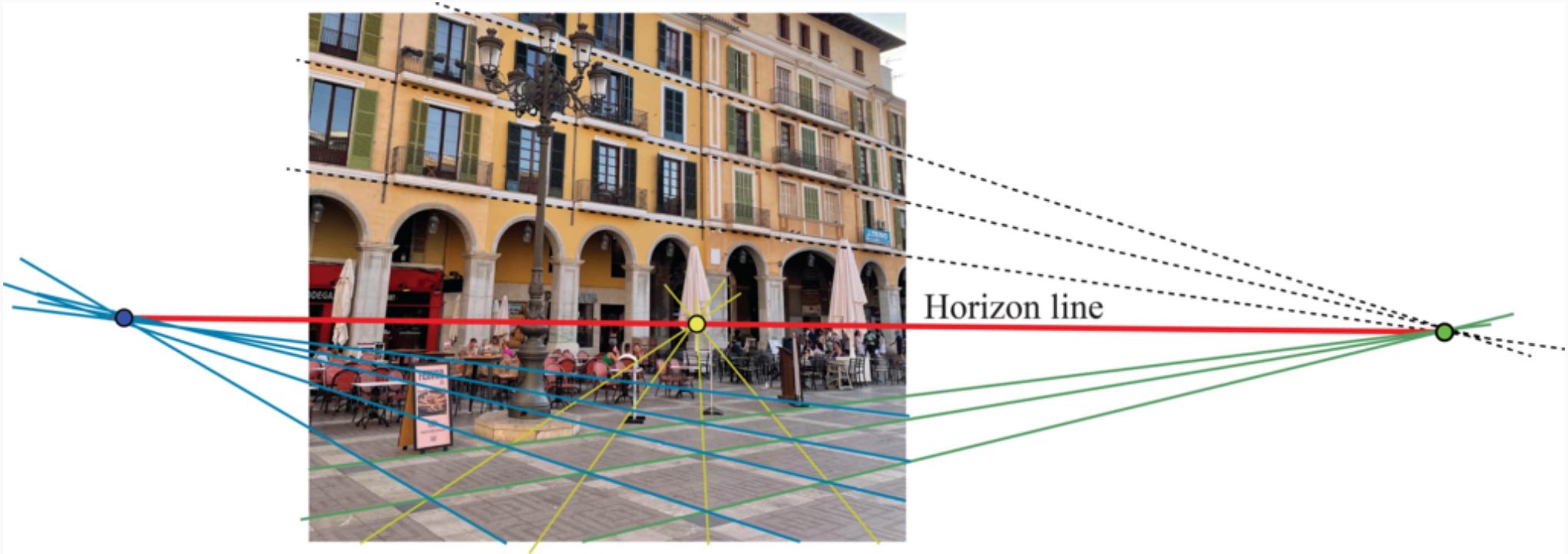


Figure 7: source: visionbook

Pictorial Depth Cues: Texture and Haze

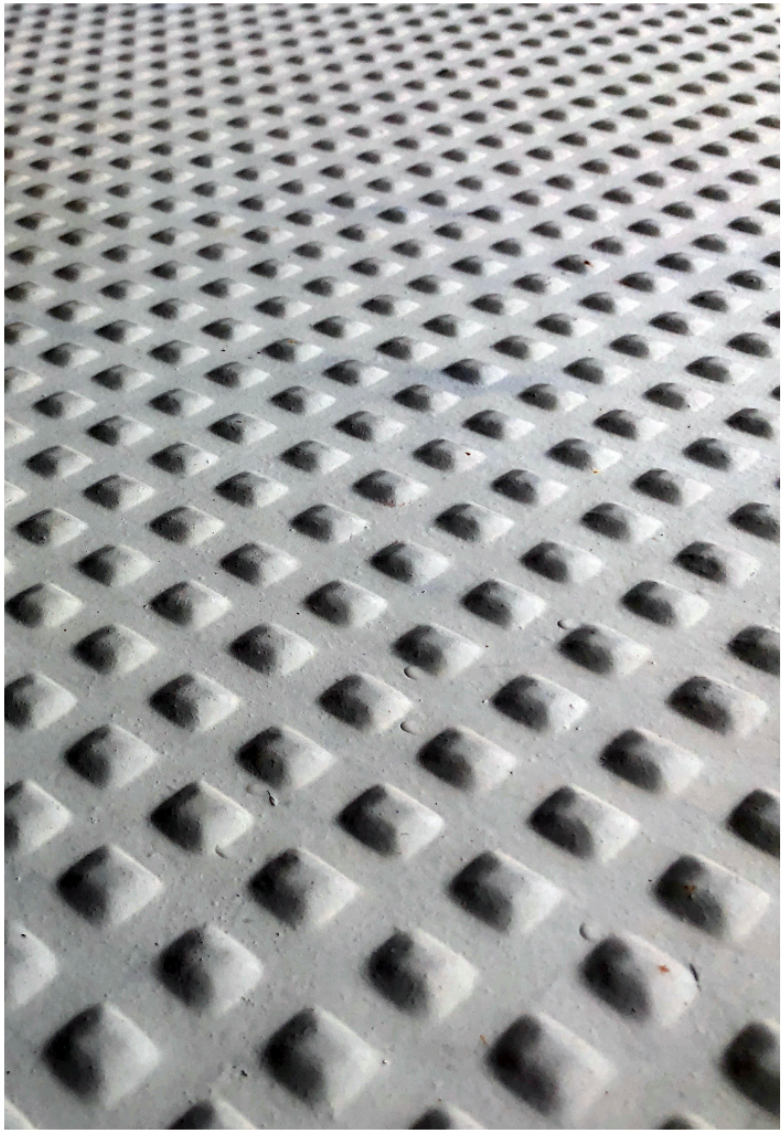


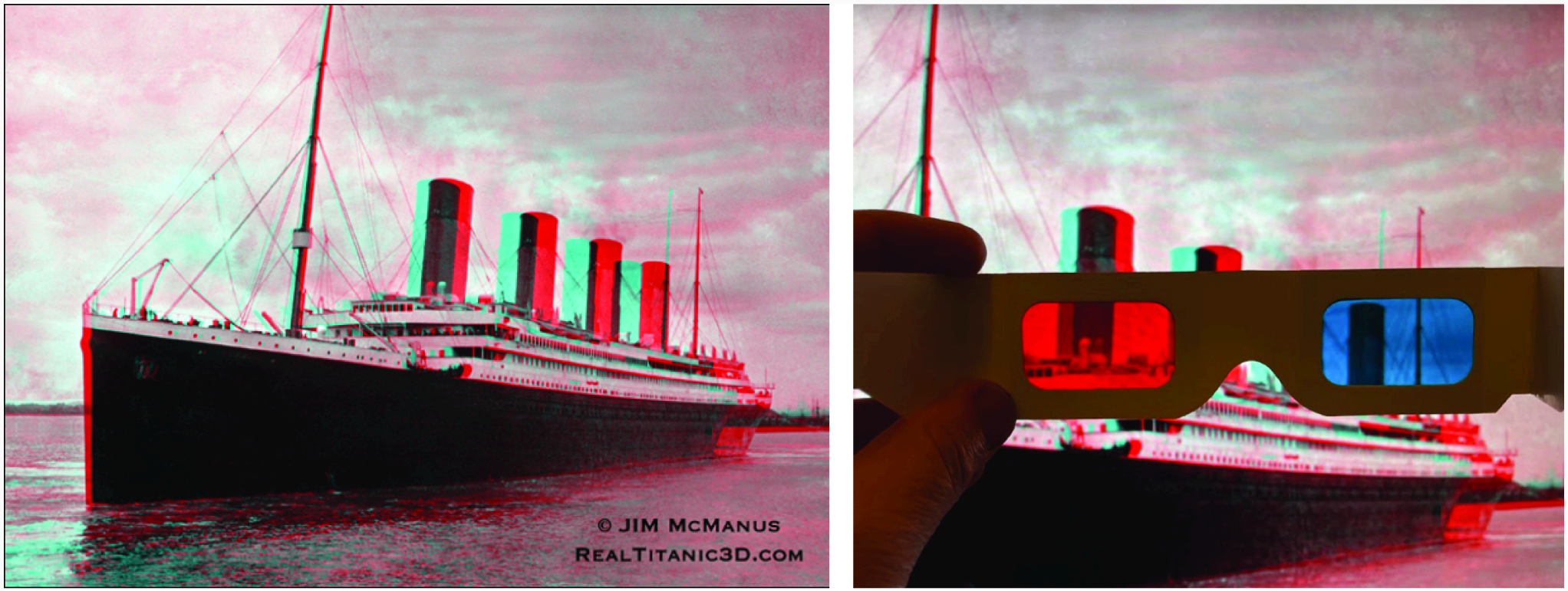


Figure 8: source: visionbook

Pictorial Depth Cues

- **Occlusion**: an object that partially blocks another is closer
- **Relative size / familiar size**: smaller size, or smaller than an object's known real size, implies farther away
- **Texture gradient**: a uniformly textured surface (grass, tiles, gravel) projects to texture that gets denser and smaller with distance
- **Shading**: for a known light direction, surface shading constrains local surface orientation
- **Aerial perspective**: atmospheric scattering reduces contrast and shifts color toward blue/haze with distance (strong only at large scales)
- **Linear perspective**: parallel lines in the 3D scene converge to vanishing points

Stereo Vision



(a) (b)

Figure 9: source: visionbook

Triangulation

IDEA: with two views, we can do triangulation, that is recover the location of the 3D point by measuring known angles and distances.

examples:

- left: $d = \frac{h}{\tan \alpha}$ (single view, known h, α)
- right: $d = t\frac{\sin(\alpha)\sin(\beta)}{\sin(\alpha+\beta)}$ (two views, known t, α, β)
- Derived from trigonometry identities. Not very precise unless h and t are very large.

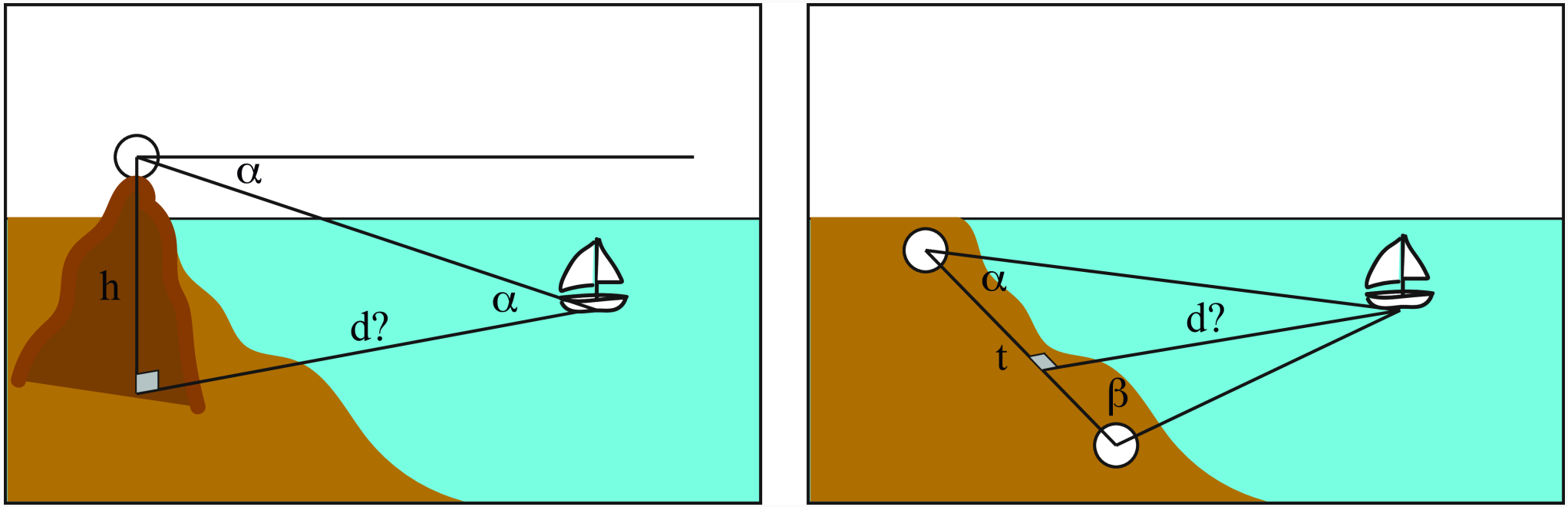


Figure 10: source: visionbook

Note: for simplicity, we start by assuming the two camera are related by only a horizontal translation T.

- we can find Z via similar triangles identities

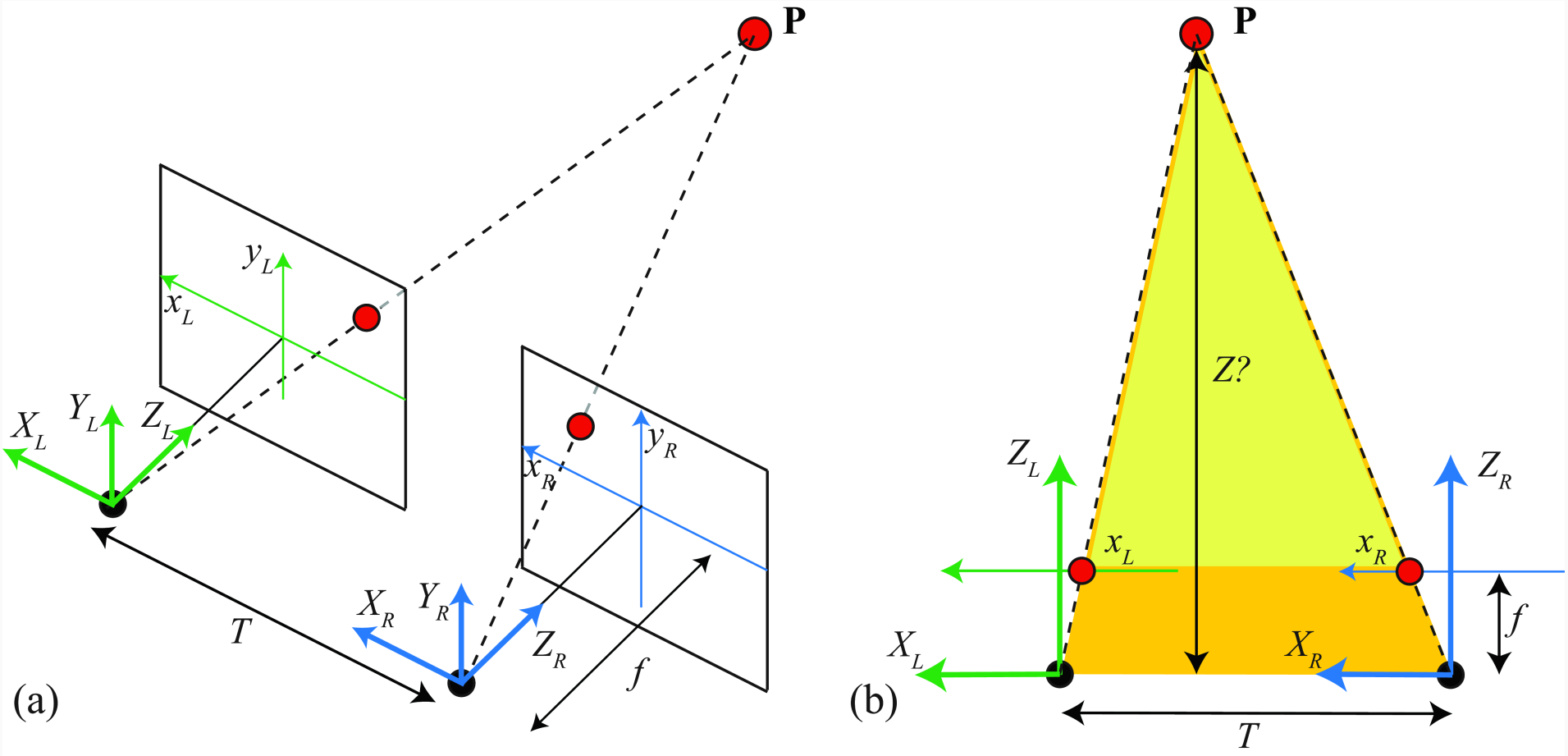


Figure 11: source: visionbook

due to similar triangles (yellow and yellow+orange) we have:

$$\frac{T + x_R - x_L}{Z - f} = \frac{T}{Z}$$

we find Z:

$$Z = \frac{fT}{x_L - x_R}$$

where $x_L - x_R$ is called the **disparity**

note that:

- f and T are parameters that depends on the camera views
- instead, we can estimate the disparity even with uncalibrated cameras

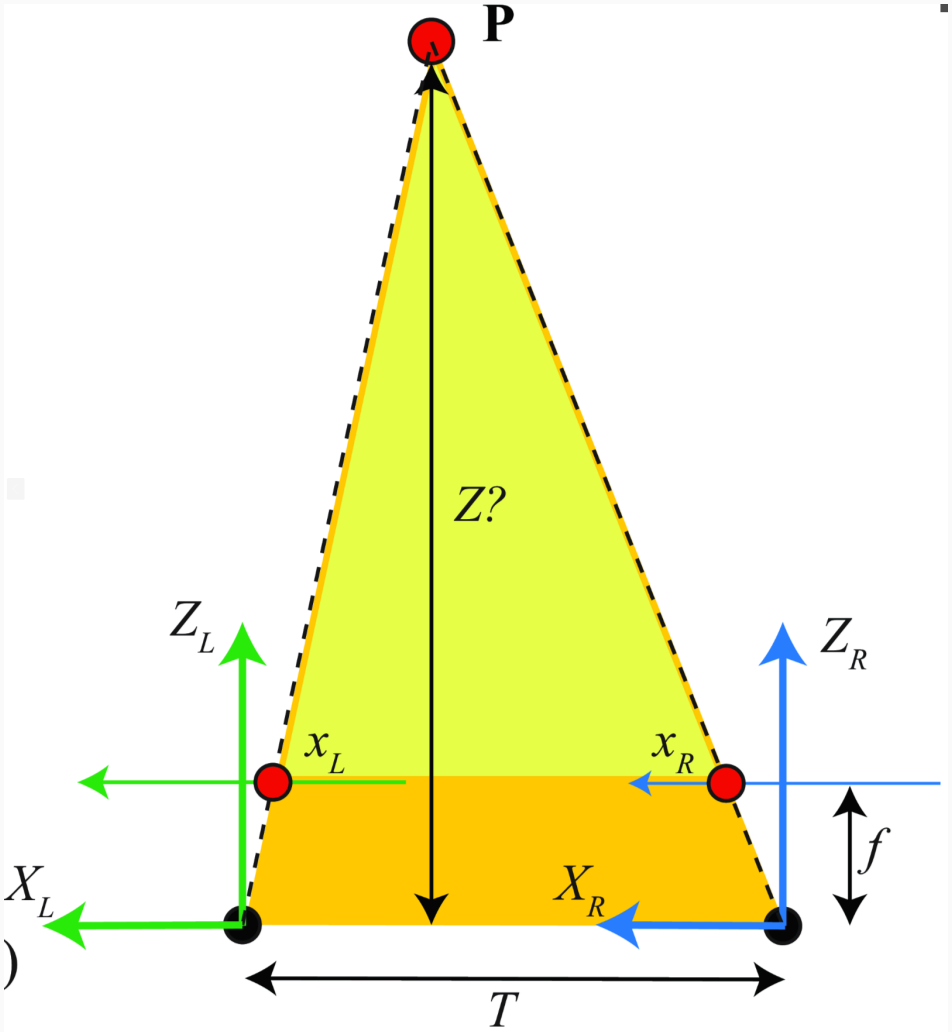


Figure 12: source: visionbook

Depth estimation

Depth prediction: given a 2D image point, estimate its corresponding 3D depth

To solve depth prediction via triangulation we need to do:

- **Rectification**: have two views that are properly aligned (i.e. camera coordinates related only by a known translation)
- **Stereo matching**: for a point p_L (left camera) know its corresponding p_R (right camera)
- **Depth computation**: compute the disparity and solve the triangulation

Rectification

Why Rectify

- we will see that matched points will lie on lines called epipolar lines
- if the two views are related only by a transalation, epipolar lines are the rows of the pixel grid
  - Epipolar lines from an arbitrary camera pair are slanted and differently oriented in the two images
  - in the general case, the depth algorithm would need to follow an arbitrary line to match pixels
- since this constraint is very convenient, we will enforce it as a preprocessing step
- **Image Rectification** warps the image such that:
  - corresponding epipolar lines become **horizontal**
  - corresponding epipolar lines in the two images fall on the **same** image row

Result: After rectification, correspondence search is a 1D search along **matching scanlines**

Stereo Matching

From Rectified Images to Disparity

- After rectification, a 3D point's images $\boldsymbol{p}_L = (x_L, y)$ and $\boldsymbol{p}_R = (x_R, y)$ share the same row y
  - we need to find row-wise pixel matches
- **Disparity**: $d = x_L - x_R$ (larger for closer points, smaller for farther points)

Stereo matching: for each pixel in the left image (L), find the best-matching pixel in the right image (R) on the same row



Figure 13: source: visionbook

Dense Stereo Matching

- in dense matching we match every pixel in L with every pixel in R
  - images are rectified, so we can limit the search row-wise
- first, we need to find correspondences between each pixel in L and R
- we can compute a local similarity measure (e.g. sum of squared difference of the intensity between local patches in L and R)

Cost Volume and Block Matching

- A **cost volume** $C(x, y, d)$ scores how well a small patch around (x, y) in L matches the patch around $(x - d, y)$ in R, for every candidate disparity d
  - search is limited to a maximum value of d
- **Block matching** compares fixed-size windows with simple, fast similarity scores:
  - SAD (sum of absolute differences)
  - SSD (sum of squared differences)
  - NCC (normalized cross-correlation, robust to brightness/contrast differences)
- The disparity estimate at each pixel is $\hat{d}(x, y) = \text{argmin}_d \, C(x, y, d)$ (or argmax for NCC)

Failure Modes

Failure modes:

- textureless regions where the cost is flat across d: no unique minimum
- repetitive patterns: many good matches
- occlusions: no correct match exists

Global / semi-global methods (e.g. Semi-Global Matching)

- add a smoothness term across neighboring pixels
- solved approximately
- trades speed for better handling of textureless/ambiguous regions

Blurring with edge-preserving filter improves the naive intensity-based approach.

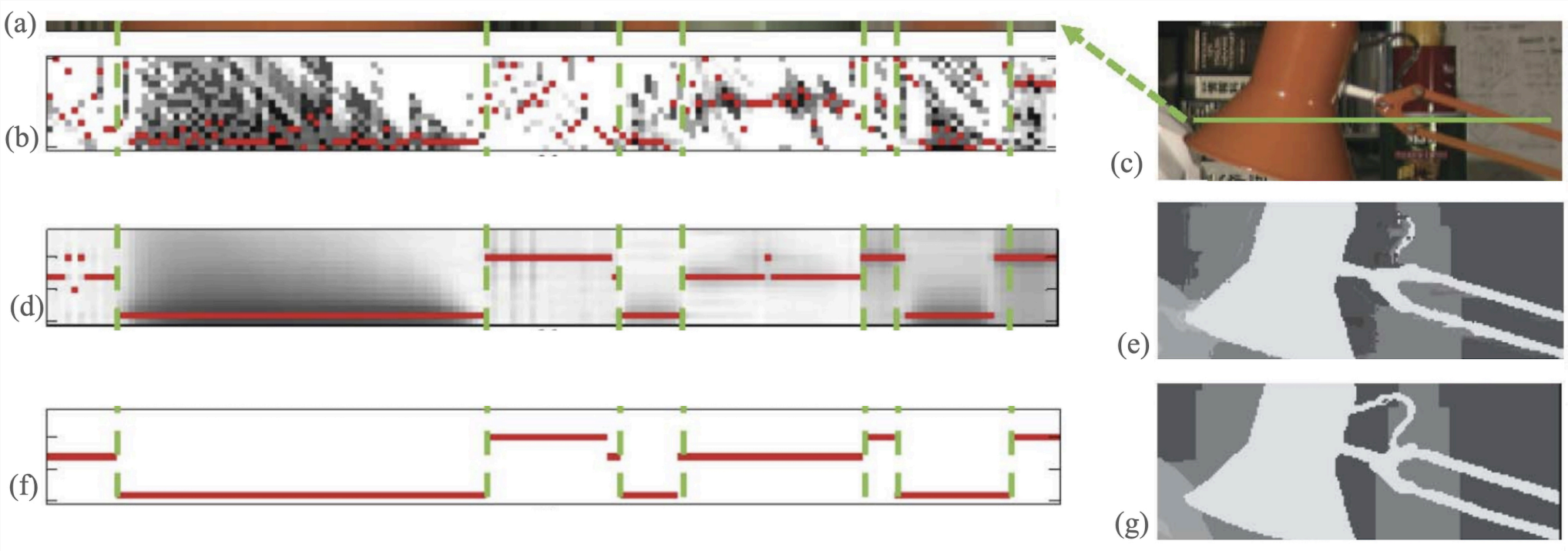


Figure 14: Disparity space image. (a) One line of one image, (c), of stereo pair. (b) Intensity matching error for different disparities, plotted vertically, as a function of horizontal position. Red shows disparities of best intensity match. (d) Smoothed matching errors, with best matches shown in red, and resulting depth image, (e). (f) Ground truth matches and depth (g). source: visionbook

Sparse Stereo Matching

we can reuse the idea of keypoints to solve depth prediction only for the keypoints:

- extract keypoints from L and R (e.g. SIFT)
- match keypoints (e.g. nearest neighbor + Lowe's ratio test)
  - in a perfectly registered image correct matches lie in horizontal lines, so we can discard any match that does not satisfy this geometric constraint

Sparse Stereo Matching

- Notice the different keypoints in L and R.
- after matching, the depth image (right) is computed via interpolation algorithms

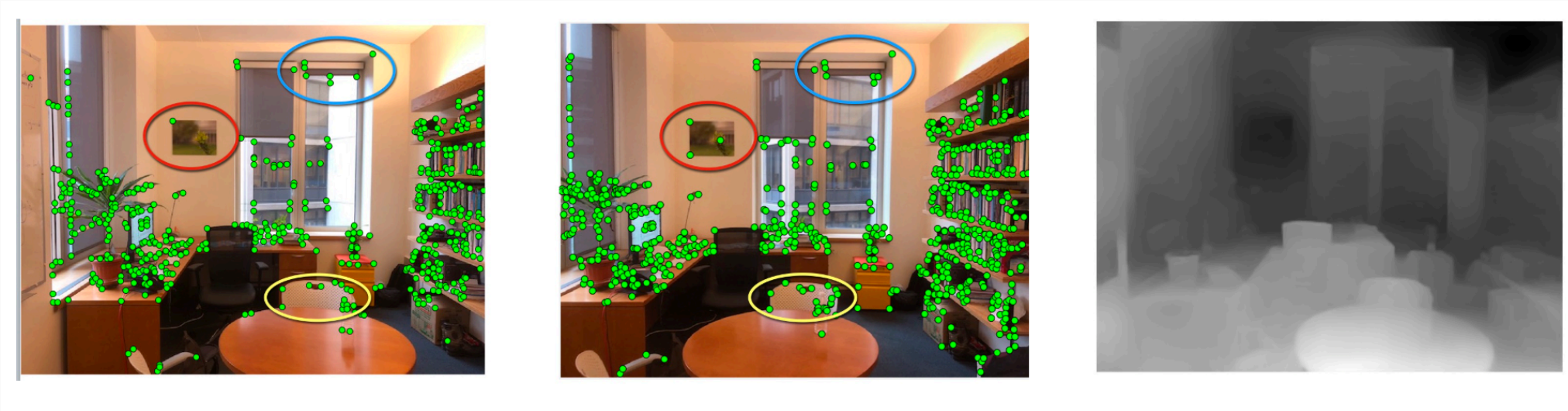


Figure 15: source: visionbook

Epipolar Geometry

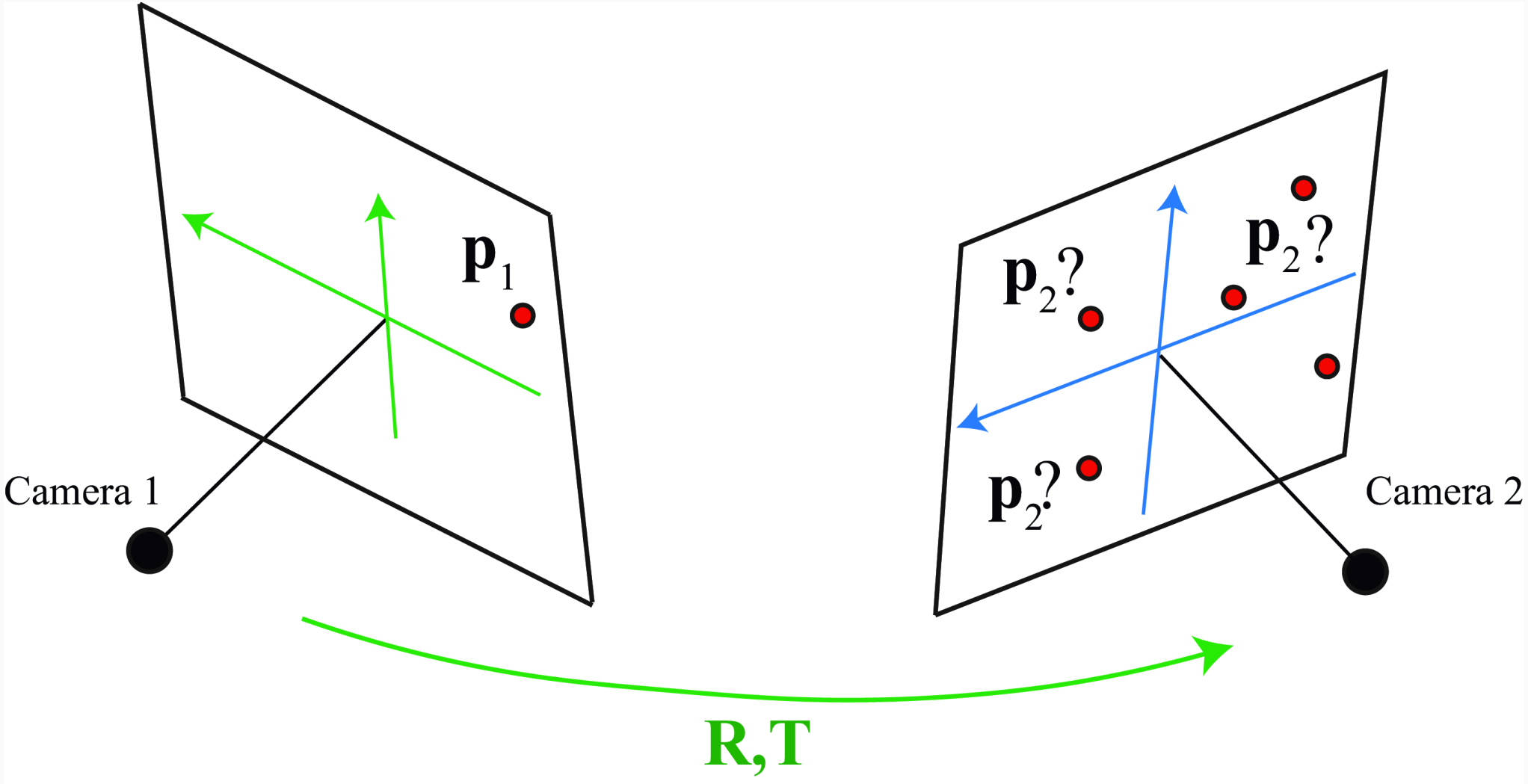


Figure 16: source: visionbook

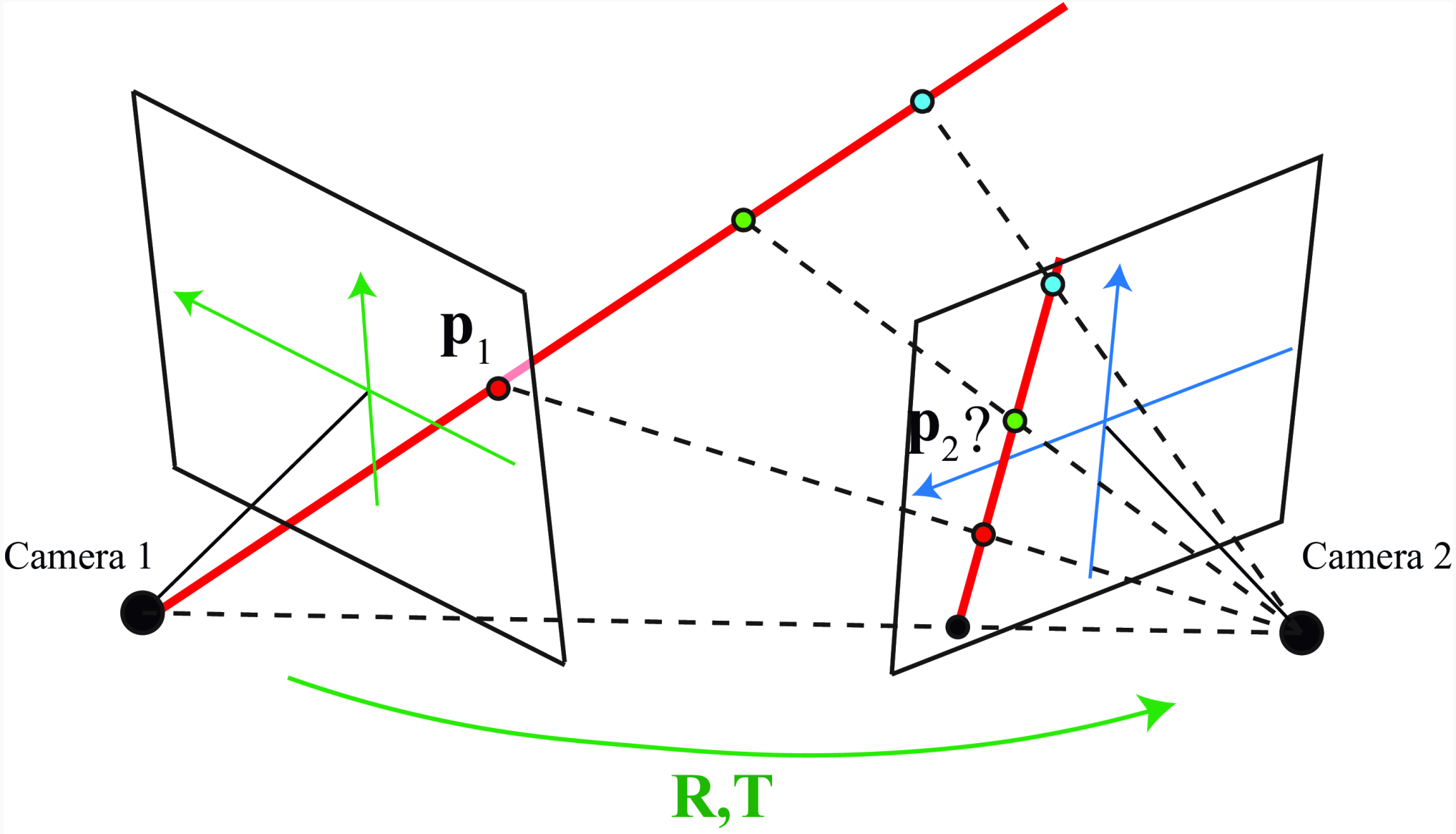


Figure 17: source: visionbook

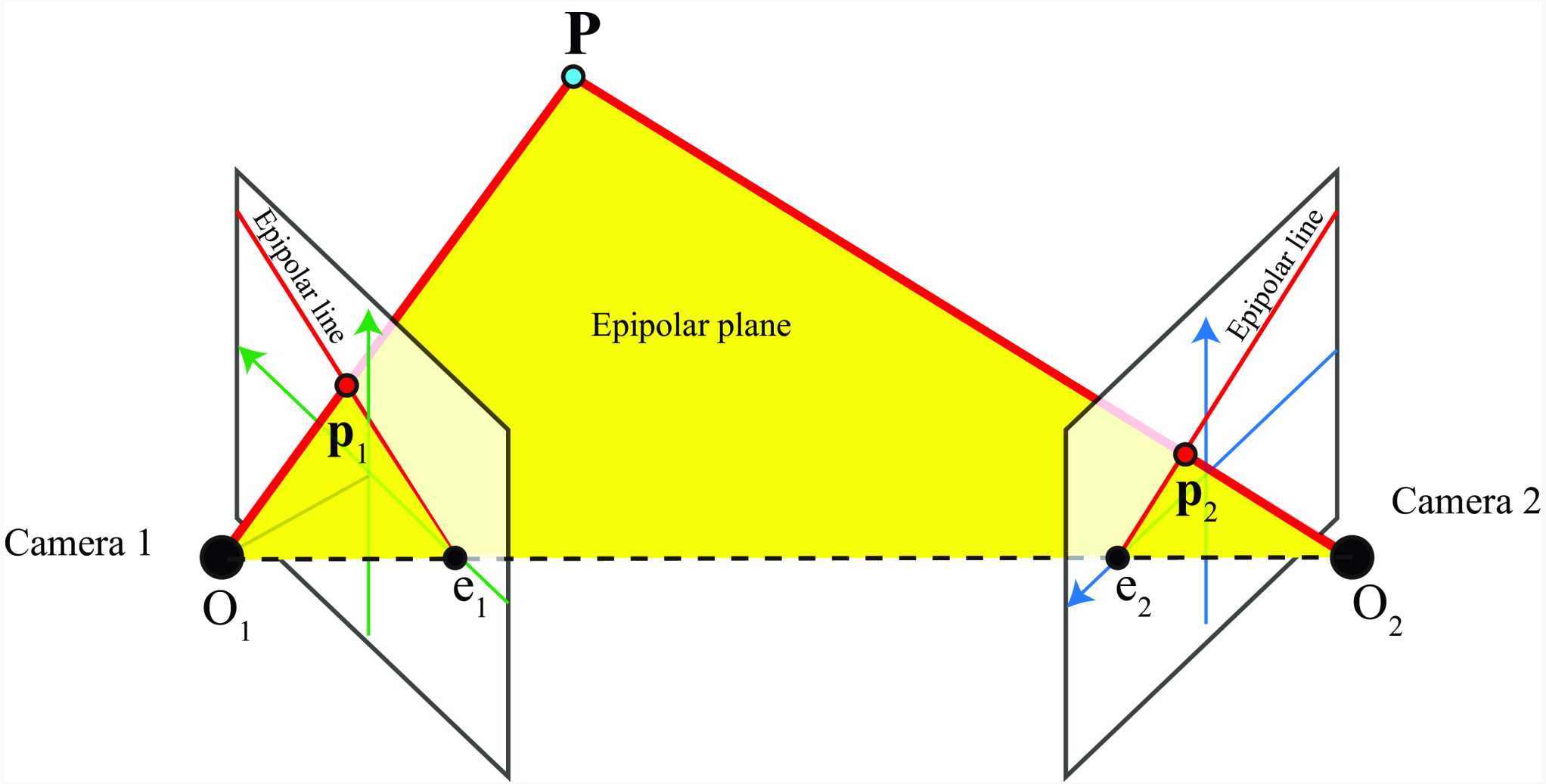


Figure 18: source: visionbook

Epipolar Constraint

Goal: we want to find a geometric constraint that correspond to

"we must search for p_2 along the correct epipolar line"

- we have a point $\boldsymbol{P}$
  - in the camera 1 frame we have 3D coordinates $\boldsymbol{P}_1$
  - in the camera 2 frame we have 3D coordinates $\boldsymbol{P}_2$
- the two coordinate systems are related via a rotation $\boldsymbol{R}$ and a translation $\boldsymbol{T}$
- in image coordinates $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$

$$\boldsymbol{P}_2 = \boldsymbol{R}\boldsymbol{P}_1 + \boldsymbol{T}$$

Essential and Fundamental Matrix

- The two coordinates are related by a matrix $\boldsymbol{E}$ called the **essential matrix** (full derivation in the book) by the **epipolar constraint**:

$$\boldsymbol{p}_2^\top \boldsymbol{E} \boldsymbol{p}_1 = 0$$

- given a point p_1, its projection p_2 must satisfy the epipolar constraint
- pixel coordinates c_1 and c_2 are similarly related by the **fundamental matrix** $\boldsymbol{F}$:

$$c_2^T \boldsymbol{F} c_1 = 0$$

- $\boldsymbol{E}$ and $\boldsymbol{F}$ can be estimated with DLT-like method applied to the epipolar constraints

Learning-Based methods

Learned Stereo Depth Estimation

- **Learned stereo networks** predict disparity since it doesn't depend on the camera parameters
  - depth is inversely proportional to disparity
- classical block matching is usually faster
- learned methods are typically more accurate, especially in hard regions

Supervised Stereo Depth Estimation

- predicted disparity: $\hat{d} = f_{\theta(v_1, v_2)}$
  - f the DNN with parameters θ
  - v_1, v_2 are the two views
- the supervised loss is the distance between the predicted and target disparity:
  - example: $J(\theta) = \sum_t |d - f_{\theta(v_1, v_2)}|^2$
- standard supervised DNN training: θ optimize by SGD to minimize the loss
- **two-stage architecture**: $f(v_1, v_2) = c(g(v_1), g(v_2))$
  - g processes the two views independently

Stereo Depth Prediction architecture

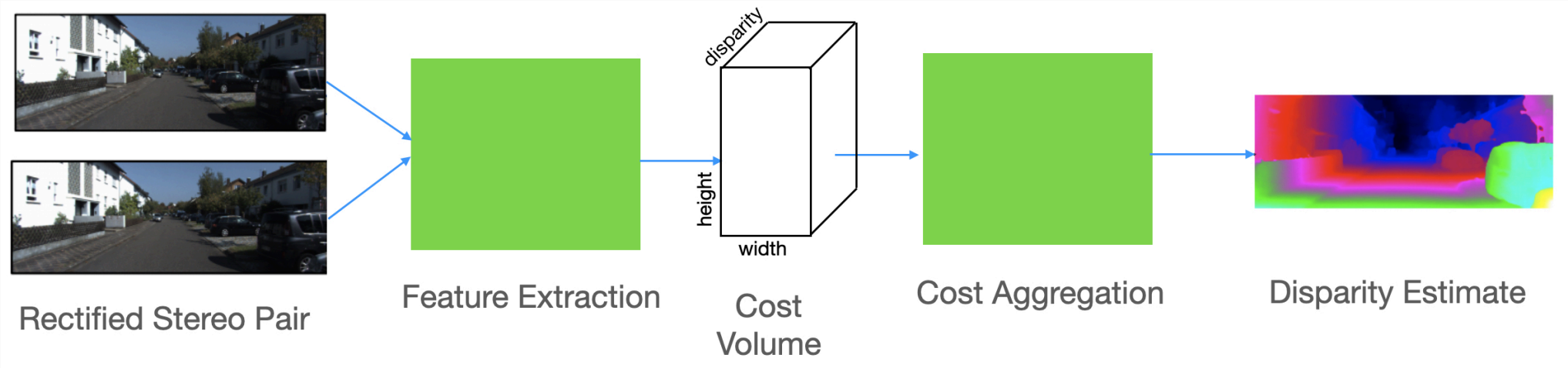


Figure 19: source: visionbook

Pipeline

- **Rectify the stereo pair** so that scanlines in each image depict the same points in the world
- **Extract features from each image**, using a pair of networks with shared weights.
- **Form a 3D cost volume** indicating the local visual evidence of a match between the two images for each possible pixel position and each possible stereo disparity.
- Train and apply a **3D CNN to aggregate the costs** over the 3D cost volume to estimate a single disparity estimate for each pixel position.
  - Implicitly performs the function of the regularization step of non-neural approaches (e.g. we have seen smoothing)

DNN Generalization

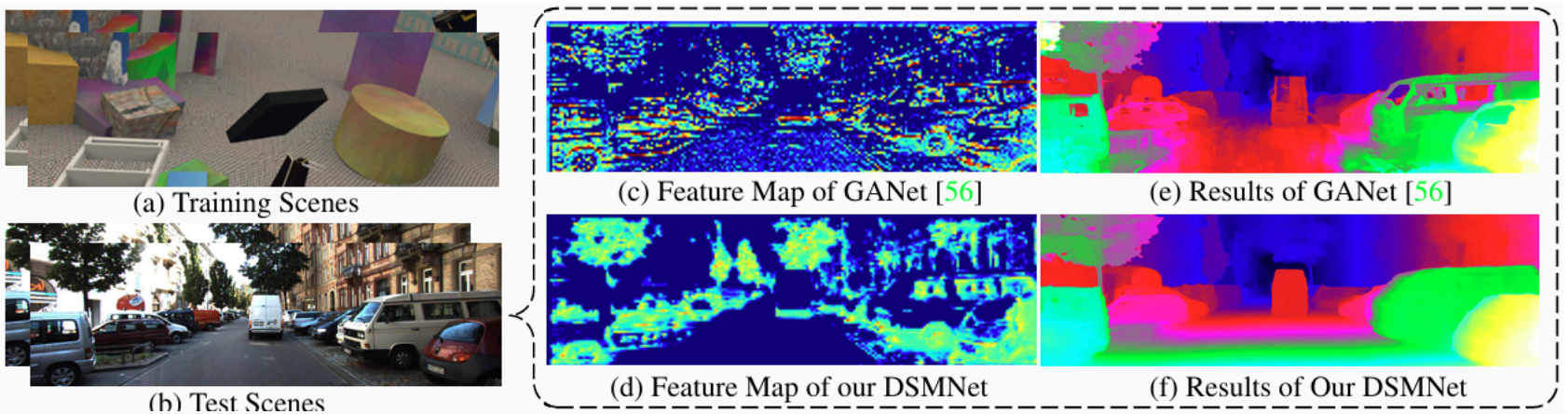
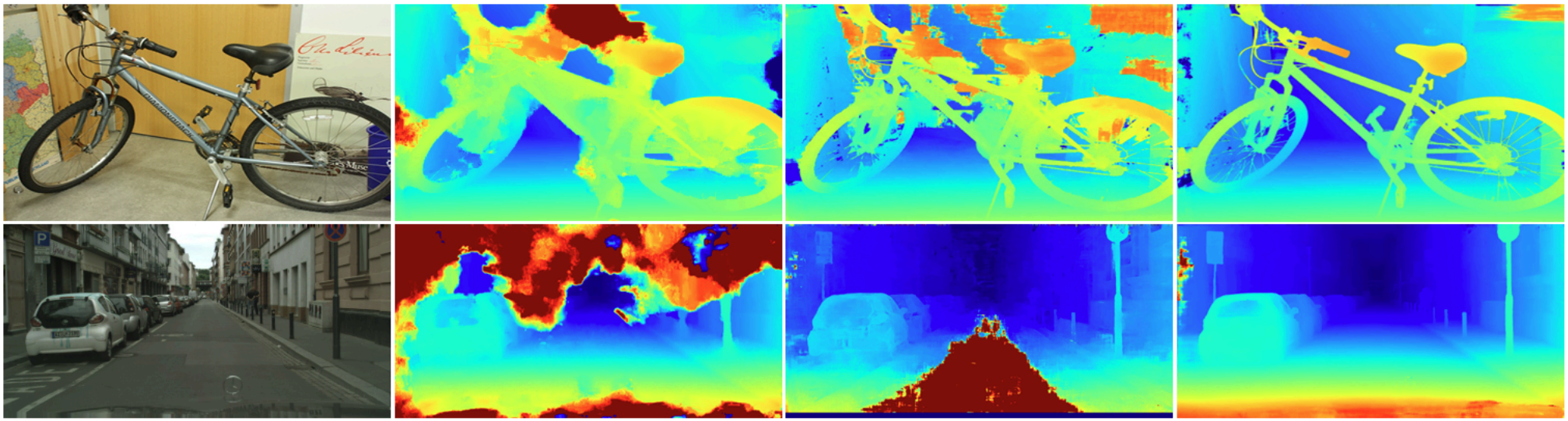


Figure 20: DNN can generalize depth prediction to novel domains. source: Zhang, Feihu, et al. 'Domain-invariant stereo matching networks.' ECCV 2020.



(a) Input view (b) HD^3 [53] (c) PSMNet [4] (d) Our DSMNet

Figure 7: Comparisons with state-of-the-art models. Models are trained on synthetic data and evaluated on high-resolution real datasets (Middlebury and CityScapes). Our DSMNet can produce much more accurate disparity estimation. (See supplementary for more results.

Figure 21: Example of inference. source: Zhang, Feihu, et al. 'Domain-invariant stereo matching networks.' ECCV 2020.

Depth from a single view

- 3D cues allow to predict depth from single images
- depth prediction becomes a pixel-wise regression task: predict depth/disparity for each pixel in the image
  - $\mathcal{L}(\hat{z}, z) = \sum_{n,m} (\hat{z}[n,m] - z[n,m])^2$
  - $\hat{z}$ estimated per-pixel depth and z groud-truth per-pixel depth
- **scale-invariant loss**: $\mathcal{L}(\hat{z}, z) = \sum_{n,m} (\hat{z}[n,m] - z[n,m] - \alpha)^2$
  - $\alpha = \frac{1}{NM} \sum_{n,m} (\hat{z}[n,m] - z[n,m])$
  - explicit smoothness/regularity priors are often added as additional losses to regularize the outputs
- datasets are usually collected with depth sensors (e.g. Kinect, LiDAR)

Single View Encoder

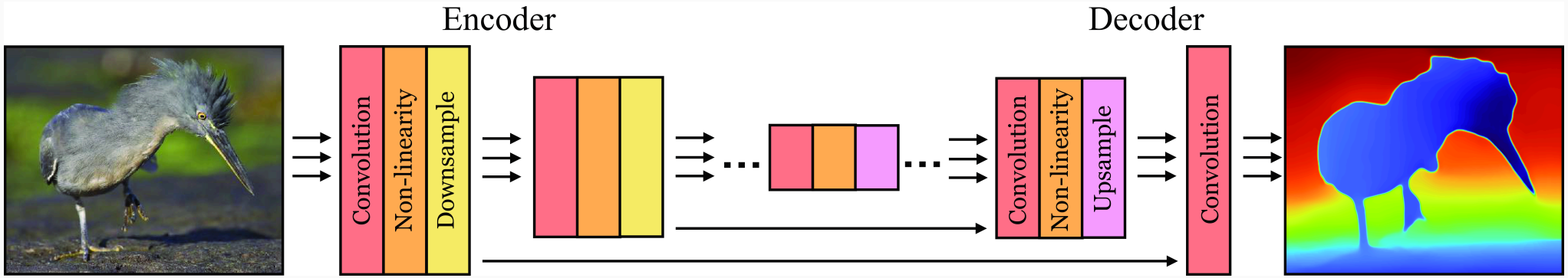


Figure 22: source: visionbook

Conclusions

Key Takeaways

- perspective cues allow to predict depth in single views
- in stereo vision, geometry constraints reduces correspondences between left and right views search from 2D to 1D (epipolar constraint)
  - rectification aligns the epipolar line with the pixel grid
  - matching left and right views plus triangulation allows us to compute the depth
- disparity is the key quantity in both classic and DNN-based methods

References

- visionbook Chapter 40
- evaluation of state-of-the-art depth prediction methods https://vision.middlebury.edu/stereo/eval3/