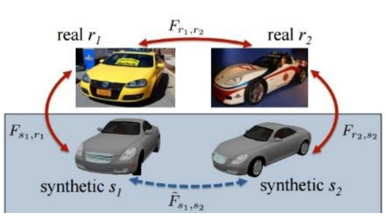
CS 395T
Lecture 6: Image Primitives and Correspondences



Qixing Huang

September 19th 2018

Overview

- Geometry reconstruction is generally impossible from a single image

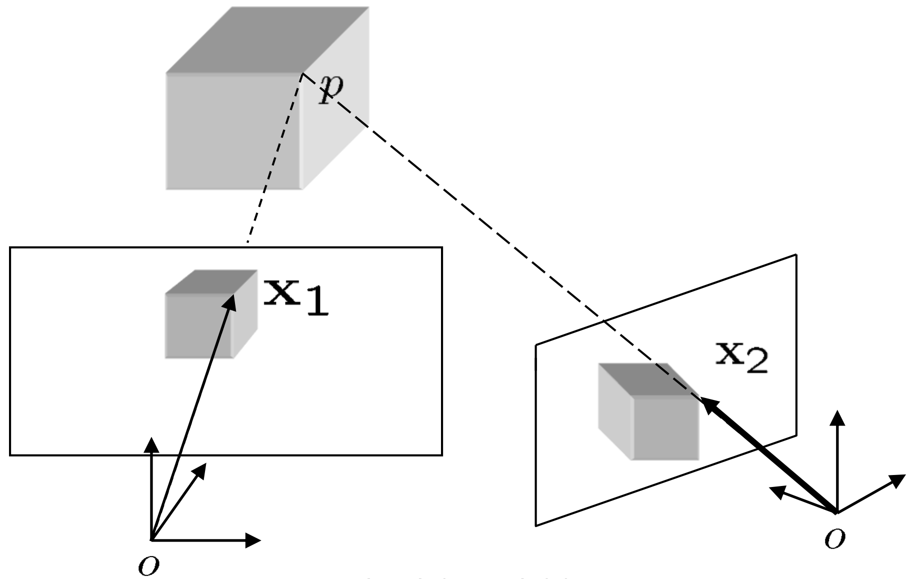
- It is possible to perform geometry reconstruction from multiple images
  - Critical issue: Which points correspond to which?

A harder problem

- Cases that are impossible to solve
  - White marble sphere rotating
  - Still mirror sphere but the light is changing
  - Objects may have anisotropic reflectance property

- This lecture:
  - What conditions the correspondence problem can be solved, and
  - Can be solved easily

Matching-correspondence



Lambertian assumption $$I_1(\mathbf{x}_1) = \mathcal{R}(p) = I_2(\mathbf{x}_2)$$

Rigid body motion $$\mathbf{x}_2 = h(\mathbf{x}_1) = \frac{1}{\lambda_2(\mathbf{X})}(R\lambda_1(\mathbf{X})\mathbf{x}_1 + T)$$

Correspondence $$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1))$$

Correspondence between images

- Simplified model

$$
\begin{aligned}
I_1 :\ \Omega \subset \mathbb{R}^2 \quad &\rightarrow \quad \mathbb{R}_+ \\
\mathbf{x} \quad &\mapsto \quad I(\mathbf{x})
\end{aligned}
\qquad\qquad
\begin{aligned}
I_2 : \Omega \subset \mathbb{R}^2 &\longrightarrow \mathbb{R}_+ \\
\mathbf{x} &\mapsto I_2(\mathbf{x})
\end{aligned}
$$

$$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1))$$

Usually either a translation or an affine transformation

- General models

$$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1)) + n(h(\mathbf{x}_1))$$

$$I_1(\mathbf{x}_1) = f_o(\mathbf{X}, \mathbf{x}) I_2(h(\mathbf{x}_1)) + n(h(\mathbf{x}_1))$$

Photometric and geometric features

- Try to characterize the locations in the image where correspondences can be established
- Associate each point with a region
  - Matches should be consistent between the corresponding regions
- Consistency is with respect to a transformation h that is parameterized by some parameter alpha (e.g., translational model and affine model)

$$I_1(\mathbf{x}) = I_2(h(\mathbf{x}, \alpha)) \ \forall \ \mathbf{x} \in W(\mathbf{x})$$

$$\hat{\alpha} = \arg\min_{\alpha} \phi(I_1(\mathbf{x}), I_2(h(\mathbf{x}, \alpha)))$$

Optical flow and feature tracking

Translational model

- Invariance assumption

$$I_1(\mathbf{x}_1) = I_2(h(\mathbf{x}_1)) = I_2(\mathbf{x}_1 + \mathbf{u})$$

$$\Big\downarrow \text{Make it continuous, like a video}$$

$$I(\mathbf{x}(t), t) = I(\mathbf{x}(t) + \mathbf{u}(t), t + dt)$$

- Image brightness constant constraint:

Derivative computation: $\nabla I(\mathbf{x}(t), t)^T \mathbf{u} + I_t(\mathbf{x}(t), t) = 0$

where

$$\nabla I(\mathbf{x}, t) \doteq \begin{bmatrix} \frac{\partial I}{\partial x}(\mathbf{x}, t) \\ \frac{\partial I}{\partial y}(\mathbf{x}, t) \end{bmatrix}, \quad \text{and} \quad I_t(\mathbf{x}, t) \doteq \frac{\partial I}{\partial t}(\mathbf{x}, t).$$

Optical flow and the aperture problem

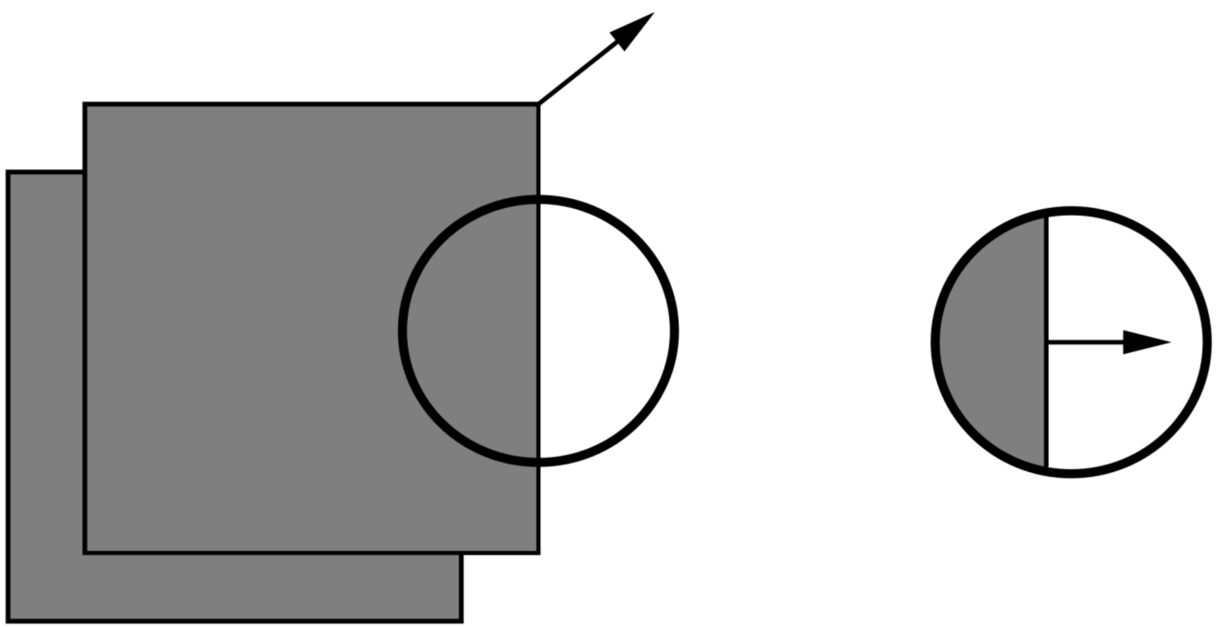
- Simplified notation

$$\boxed{\nabla I^T \mathbf{u} + I_t = 0}$$

- Eulerian view:
  - Fix our attention at a particular image location and compute the velocity of “particles flowing” through that pixel
  - **u** is called a optical flow
- Lagrangian view:
  - Fix our attention at a particular particle **x**(t)
  - This is called feature tracking

Aperture problem

- A single constraint does not uniquely specify the motion
  - We cannot differentiate diagonal motion and horizontal motion



Local constancy

- Motion is the same for all points in a window W(**x**)
- This is equivalent to assuming a purely translational deformation model:

$$h(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\lambda\mathbf{x}) = \mathbf{x} + \mathbf{u}(\lambda\mathbf{x}) \quad \text{for all } \mathbf{x} \in W(\mathbf{x})$$

$$\Big\downarrow \text{Optimization formulation}$$

$$E_b(\mathbf{u}) = \sum_{W(x,y)} \left(\nabla I^T(x,y,t) \cdot \mathbf{u}(x,y) + I_t(x,y,t)\right)^2$$

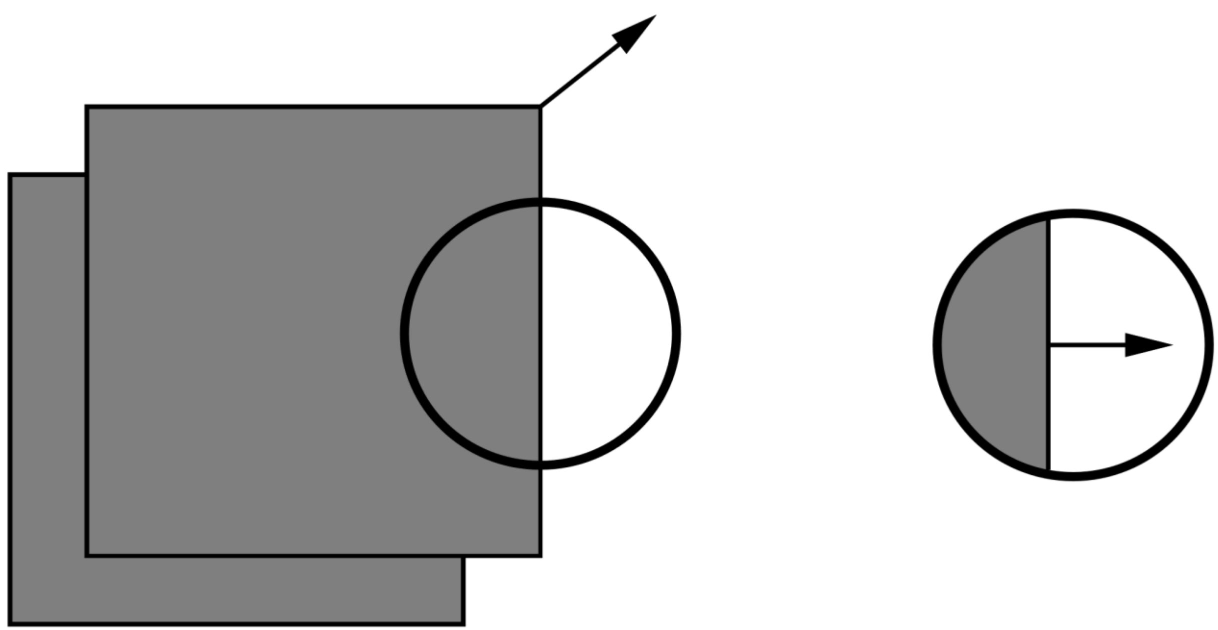
$$\Big\downarrow \text{Least square solution}$$

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \mathbf{u} + \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} = 0$$

G may be degenerate

- The intensity variation in a local image window varies only along one dimension or vanishes

$I_x = 0$ and/or $I_y = 0$



Affine deformation model

- Translational model is too rigid in many settings
- A more flexible model is that affine deformation of image regions that support point features $I_1(x)= I_2(h(x))$, where the function h has the following form:

$$h(\mathbf{x}) = A\mathbf{x} + \mathbf{d} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \mathbf{x} + \begin{bmatrix} a_5 \\ a_6 \end{bmatrix}$$

- We can estimate A and d via the following adapted objective function:

$$E_a(A, \mathbf{d}) = \sum_{W(\mathbf{x})} (I(\mathbf{x}, t) - I(A\mathbf{x} + \mathbf{d}, t + dt))^2$$

Feature detection algorithms

Computing image gradient

- A standard approach

$$I_x(x,y) = \frac{\partial I}{\partial x}(x,y), \quad I_y(x,y) = \frac{\partial I}{\partial y}(x,y).$$

- A more robust approach

$$\tilde{I}(x,y) = \int_u \int_v I(u,v) g_\sigma(x-u) g_\sigma(y-v)\ dudv.$$

$$\downarrow$$

$$I_x(x,y) = \int_u \int_v I(u,v) g'_\sigma(x-u) g_\sigma(y-v)\ dudv$$

$$I_y(x,y) = \int_u \int_v I(u,v) g_\sigma(x-u) g'_\sigma(y-v)\ dudv$$

Approximation

Line features: edges

Algorithm 4.1 (Canny edge detector). *Given an image $I(x, y)$, follow the steps to detect if a given pixel (x, y) is an edge:*

- *set a threshold $\tau > 0$ and standard deviation $\sigma > 0$ for the Gaussian function,*
- *compute the gradient vector $\nabla I = [I_x, I_y]^T$ according to*

$$\boxed{\begin{array}{rcl} I_x(x,y) & = & \sum_{u,v} I(u,v) g'_\sigma(x-u) g_\sigma(y-v) \\ I_y(x,y) & = & \sum_{u,v} I(u,v) g_\sigma(x-u) g'_\sigma(y-v) \end{array}} \tag{4.34}$$

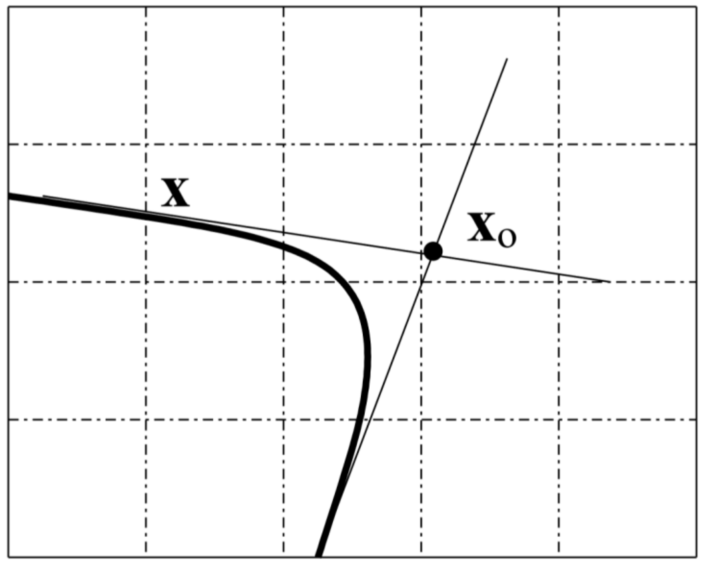
- *if $\|\nabla I(x,y)\|^2 = \nabla I^T \nabla I$ is a local maximum along the gradient and larger than the prefixed threshold τ, then mark it as an edge pixel.*

An example

Figure 4.4. Original image.

Figure 4.5. Edge pixels from the Canny edge detectors.

Point features: corners



A corner feature x_o is the virtual intersection of local edges

$$\min_{\mathbf{x}_o} \; E_c(\mathbf{x}_o) \doteq \sum_{\mathbf{x} \in W(\mathbf{x}_o)} \left(\nabla I^T(\mathbf{x})(\mathbf{x} - \mathbf{x}_o) \right)^2$$

Corner detection

Algorithm 4.2 (Corner detector). *Given an image $I(x, y)$, follow the steps to detect if a given pixel (x, y) is a corner feature:*

- *set a threshold $\tau \in \mathbb{R}$ and a window W of fixed size,*
- *for all pixels in the window W around (x, y) compute the matrix*

$$\boxed{G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}} \tag{4.38}$$

- *if the smallest singular value $\sigma_{min}(G)$ is bigger than the prefixed threshold τ, then mark the pixel as a feature (or corner) point.*

Corner definition: The irradiance change enough in two independent directions

Harris edge and corner detector

- Thresholding the following quantity:

$$C(G) = \det(G) + k \cdot \mathrm{trace}^2(G)$$

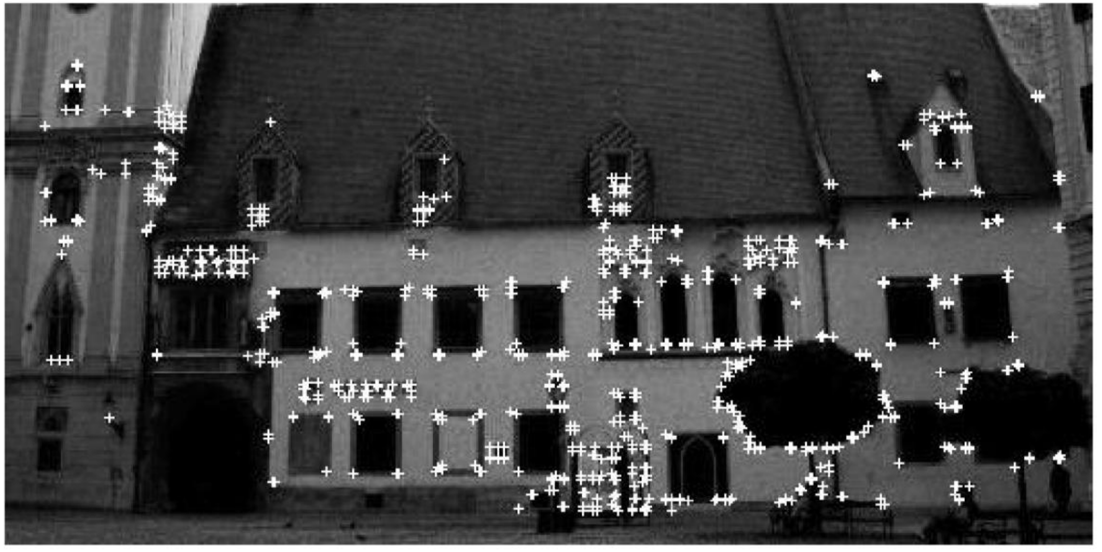


Figure 4.7. An example of the response of the Harris feature detector using 5×5 integration window and parameter $k = 0.04$.

$$C(G) = \sigma_1\sigma_2 + k(\sigma_1 + \sigma_2)^2 = (1 + 2k)\sigma_1\sigma_2 + k(\sigma_1^2 + \sigma_2^2).$$

Feature tracking

Wide baseline matching

Region based similarity metric

- Sum of squared differences

$$SSD(h) = \sum_{\tilde{\mathbf{x}} \in W(\mathbf{x})} \|I_1(\tilde{\mathbf{x}}) - I_2(h(\tilde{\mathbf{x}})\|^2$$

- Normalize cross-correlation

$$NCC(h) = \frac{\sum_{W(\mathbf{x})}(I_1(\tilde{\mathbf{x}})-\bar{I}_1)(I_2(h(\tilde{\mathbf{x}}))-\bar{I}_2))}{\sqrt{\sum_{W(\mathbf{x})}(I_1(\tilde{\mathbf{x}})-\bar{I}_1)^2 \sum_{W(\mathbf{x})}(I_2(h(\tilde{\mathbf{x}}))-\bar{I}_2)^2)}}$$

- Sum of absolute differences

$$SAD(h) = \sum_{\tilde{\mathbf{x}} \in W(\mathbf{x})} |I_1(\tilde{\mathbf{x}}) - I_2(h(\tilde{\mathbf{x}}))|$$

Questions?