ECE276A: Sensing & Estimation in Robotics
Lecture 14: Visual Features

Instructor:
Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:
Qiaojun Feng: qif007@eng.ucsd.edu
Arash Asgharivaskasi: aasghari@eng.ucsd.edu
Thai Duong: tduong@eng.ucsd.edu
Yiran Xu: y5xu@eng.ucsd.edu

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering

From Photometry to Geometry

- Suppose that instead of a lidar (which measures the positions of points in the world), we would like to use a camera to localize our robot and build a map of the environment
- **Image**: an array of positive numbers that measure the amount of light incident on the sensor
- How do we go from measurements of light (**photometry**) to measurements of positions of points in the world?

Correspondence


- **Corresponding points** in two views are image projections of the same geometric point in space
- **Correspondence problem**: establish which point in the second image corresponds to a given point $\mathbf{z}_1 \in \mathbb{R}^2$ in the first image in the sense of being the same point in physical space
- **Idea**: look for a pixel $\mathbf{z}_2 \in \mathbb{R}^2$ such that $I_2(\mathbf{z}_2) \approx I_1(\mathbf{z}_1)$

Correspondence

- **Matching windows**: a much more robust process of establishing correspondence is to compare not the brightness of individual pixels but that of small windows $W(\mathbf{z}_1)$, $W(\mathbf{z}_2)$ around the points
- **Aperture problem**: the brightness profile within the selected windows is not rich enough to allow us to recover the transformation of the pixel z_1 uniquely (e.g., blank wall)
- **Features**: points whose local regions are rich enough to allow solving the correspondence problem. Features establish a link between photometric measurements and geometric primitives.
- The window shape $W(\mathbf{z}_1)$ and image values $I_1(\mathbf{y})$, $\mathbf{y} \in W(\mathbf{z}_1)$, associated with a pixel $\mathbf{z}_1$ in the first image undergo a *nonlinear transformation* as a consequence of the change of viewpoint

Brightness constancy constraint

- Suppose we are imaging a point $\mathbf{m} \in \mathbb{R}^3$ that emits light with the same energy in all directions (Lambertian) and radiance distribution $\mathcal{R}(\mathbf{m})$
- Suppose the camera is calibrated (i.e., $K = I_{3\times 3}$) and the two camera frames are related by the rigid-body transformation $(R, \mathbf{p}) \in SE(3)$.
- Let I_1 and I_2 be two images and $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^2$ be the two pixels corresponding to $\mathbf{m}$:

$$I_2(\mathbf{z}_2) = I_1(\mathbf{z}_1) \sim \mathcal{R}(\mathbf{m})$$

- From the projection equations, the point $\mathbf{z}_1$ in image I_1 corresponds to the point $\mathbf{z}_2$ in image I_2 if:

$$\mathbf{z}_2 = g(\mathbf{z}_1) := \frac{1}{\lambda_2}\left(\lambda_1 R \mathbf{z}_1 + \mathbf{p}\right)$$

  where λ_1, λ_2 are the **unknown** scales/depths of the observed point $\mathbf{m}$.
- **Brightness constancy constraint**: $\boxed{I_1(\mathbf{z}_1) = I_2(g(\mathbf{z}_1))}$

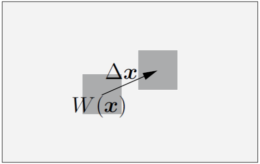
Local Deformation Models

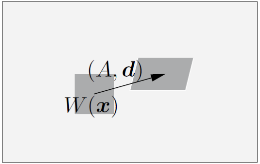
- The transformation g undergone by the entire image is determined by the scales λ_1, λ_2 of the visible surface and hence estimating g is as difficult as estimating the shape of the visible objects!
- Instead, we model the transformation only locally in a region $W(\mathbf{z})$:
  - **Translational model**: each point in the window undergoes the exact same translational motion $d \in \mathbb{R}^2$:

    $$g(\mathbf{y}) \approx \mathbf{y} + \mathbf{d}, \quad \forall \mathbf{y} \in W(\mathbf{z})$$

    This model is valid only in small windows and over short time durations but it is at the core of many feature matching and tracking algorithms.
  - **Affine model**: each point in the window undergoes an affine transformation with parameters $A \in \mathbb{R}^{2\times 2}$ and $\mathbf{d} \in \mathbb{R}^2$:

    $$g(\mathbf{y}) \approx A\mathbf{y} + \mathbf{d}, \quad \forall \mathbf{y} \in W(\mathbf{z})$$





Matching Point Features

- Requiring that $I_1(\mathbf{z}_1) = I_2(g(\mathbf{z}_1))$ is too much to ask for due to the approximation of g and the presence of noise and occlusions

- **Correspondence problem**: an optimization problem that aims to determine the (translation or affine) parameters of the local transformation model of g:

$$\min_{\mathbf{d}} \sum_{\mathbf{y} \in W(\mathbf{z})} \|I_1(\mathbf{y}) - I_2(\mathbf{y} + \mathbf{d})\|_2^2 \quad \text{OR} \quad \min_{A,\mathbf{d}} \sum_{\mathbf{y} \in W(\mathbf{z})} \|I_1(\mathbf{y}) - I_2(A\mathbf{y} + \mathbf{d})\|_2^2$$

- Our approximations of g are valid only locally in space and **time** so consider the continuous version of the brightness constancy constraint:

$$I_1(\mathbf{z}) = I(\mathbf{z}(t), t) \underbrace{\approx}_{\text{brightness constancy}} I_2(g(\mathbf{z})) \underbrace{\approx}_{\text{translation model}} I(\mathbf{z}(t) + \boldsymbol{\nu} dt, t + dt)$$

where dt is small and $\boldsymbol{\nu} \in \mathbb{R}^2$ is the velocity of $\mathbf{z}$

Continuous-Time Brightness Constancy

- **Brightness Constancy** (for the affine model):
  $I(\mathbf{z}, t) \approx I(A\mathbf{z} + \boldsymbol{\nu} dt, t + dt)$
- Linearizing the right-hand side around (z, t):
  $$I(A\mathbf{z} + \boldsymbol{\nu} dt, t + dt) \approx I(\mathbf{z}, t) + \nabla_{\mathbf{z}} I(\mathbf{z}, t)^\top (A\mathbf{z} + \boldsymbol{\nu} dt - \mathbf{z}) + \frac{\partial I}{\partial t}(\mathbf{z}, t) dt$$
  leads to:
  - Translational: $\min_{\boldsymbol{\nu}} \sum_{\mathbf{y} \in W(\mathbf{z})} \left\| \nabla_{\mathbf{z}} I(\mathbf{y}, t)^\top \boldsymbol{\nu} + \frac{\partial I}{\partial t}(\mathbf{y}, t) \right\|_2^2$
  - Affine: $\min_{A, \boldsymbol{\nu}} \sum_{\mathbf{y} \in W(\mathbf{z})} \left\| \nabla_{\mathbf{z}} I(\mathbf{y}, t)^\top \left(\frac{(A - I)}{dt} \mathbf{y} + \boldsymbol{\nu} \right) + \frac{\partial I}{\partial t}(\mathbf{y}, t) \right\|_2^2$
- **Aperture problem**: The brightness constancy equation $(\frac{\partial I}{\partial \mathbf{z}} \boldsymbol{\nu} + \frac{\partial I}{\partial t} = 0)$ provides only one constraint for the two unknowns $\boldsymbol{\nu} \in \mathbb{R}^2$.
- There are enough constraints on $\boldsymbol{\nu}$ only when the brightness constancy constraint is applied to each $\mathbf{y}$ in a region $W(\mathbf{z})$ that contains "sufficient texture" and the motion $\boldsymbol{\nu}$ is assumed constant in the region.

Feature Tracking and Optical Flow

- The brightness constancy equation ($\frac{\partial I}{\partial \mathbf{z}}\boldsymbol{\nu} + \frac{\partial I}{\partial t} = 0$) can be used to compute optical flow or track photometric features in a sequence of moving images
- **Optical flow**: the velocity $\boldsymbol{\nu}$ of particle flowing through a given image location $\mathbf{z}$
- **Feature tracking**: the computation of the velocity $\boldsymbol{\nu}$ of a particle $\mathbf{z}(t)$ moving through the image domain so that $\mathbf{z}(t + dt) = \mathbf{z}(t) + \boldsymbol{\nu} dt$ (translational model)
- The only difference between optical flow and feature tracking is at the conceptual level, whether the vector $\boldsymbol{\nu}$ is computed at fixed locations in the image or at moving points $\mathbf{z}(t)$

Feature Tracking and Optical Flow

- To compute the velocity $\boldsymbol{\nu}$ we need to solve:

$$\min_{\boldsymbol{\nu}} \sum_{\mathbf{y} \in W(\mathbf{z})} \left\| \nabla_{\mathbf{z}} I(\mathbf{y}, t)^\top \boldsymbol{\nu} + \frac{\partial I}{\partial t}(\mathbf{y}, t) \right\|_2^2$$

- Letting $\mathbf{z} = (u, v)$ and setting the gradient to zero results in:

$$\begin{aligned} 0 &= 2 \sum_{\mathbf{y} \in W(\mathbf{z})} \left(\nabla_{\mathbf{z}} I(\mathbf{y}, t)^\top \boldsymbol{\nu} + \frac{\partial I}{\partial t}(\mathbf{y}, t) \right) \nabla_{\mathbf{z}} I(\mathbf{y}, t) \\ &= 2 \sum_{\mathbf{y} \in W(\mathbf{z})} \left(\begin{bmatrix} I_u^2(\mathbf{y}) & I_u(\mathbf{y}) I_v(\mathbf{y}) \\ I_u(\mathbf{y}) I_v(\mathbf{y}) & I_v(\mathbf{y})^2 \end{bmatrix} \boldsymbol{\nu} + \begin{bmatrix} I_u(\mathbf{y}) I_t(\mathbf{y}) \\ I_v(\mathbf{y}) I_t(\mathbf{y}) \end{bmatrix} \right) \\ &= 2 \left(\underbrace{\begin{bmatrix} \sum_{\mathbf{y}} I_u^2(\mathbf{y}) & \sum_{\mathbf{y}} I_u(\mathbf{y}) I_v(\mathbf{y}) \\ \sum_{\mathbf{y}} I_u(\mathbf{y}) I_v(\mathbf{y}) & \sum_{\mathbf{y}} I_v(\mathbf{y})^2 \end{bmatrix}}_{G(\mathbf{z})} \boldsymbol{\nu} + \underbrace{\begin{bmatrix} \sum_{\mathbf{y}} I_u(\mathbf{y}) I_t(\mathbf{y}) \\ \sum_{\mathbf{y}} I_v(\mathbf{y}) I_t(\mathbf{y}) \end{bmatrix}}_{b(\mathbf{z})} \right) \end{aligned}$$

- The optimal estimate of the image velocity at $\mathbf{z}$ is $\boxed{\boldsymbol{\nu}^* = -G(\mathbf{z})^{-1} b(\mathbf{z})}$

Point Feature Selection

- For $G(\mathbf{z})$ to be invertible, the region $W(\mathbf{z})$ must have nontrivial gradients along independent directions, therefore resembling a "corner" structure.
- **Corner feature**: a pixel $\mathbf{z}$ such that the smallest singular value of $G(\mathbf{z})$ (equal to the eigenvalues for a symmetric matrix) is larger than some threshold τ
- **Harris corner detector**: A variation of the corner detector that thresholds the quantity:

  $$\det(G) + k\,\mathrm{tr}^2(G) = \sigma_1\sigma_2 + k(\sigma_1+\sigma_2)^2 = (1+2k)\sigma_1\sigma_2 + k(\sigma_1+\sigma_2)^2,$$

  where $k \in \mathbb{R}$ is a small scalar and σ_1, σ_2 are the singular values of G. Since k is small, both singular values of G need to be sufficiently large to pass the threshold.
- More sophisticated techniques that utilize contours (or edges) and search for high curvature points in the detected contours are used in practice

Feature Tracking and Optical Flow

Algorithm 1 Basic Feature Tracking and Optical Flow

Input: Image I at time t

Compute the image gradient (I_u, I_v)

Compute $G(\mathbf{z}) := \begin{bmatrix} \sum_{\mathbf{y} \in W(\mathbf{z})} I_u^2(\mathbf{y}) & \sum_{\mathbf{y} \in W(\mathbf{z})} I_u(\mathbf{y}) I_v(\mathbf{y}) \\ \sum_{\mathbf{y} \in W(\mathbf{z})} I_u(\mathbf{y}) I_v(\mathbf{y}) & \sum_{\mathbf{y} \in W(\mathbf{z})} I_v^2(\mathbf{y}) \end{bmatrix}$ at every pixel $\mathbf{z} = (u, v)$

(Feature tracking) select point features $\mathbf{z}_1, \mathbf{z}_2, \ldots$ such that $G(\mathbf{z}_i)$ is invertible

(Optical flow) select $\mathbf{z}_i$ on a fixed grid

Compute $b(\mathbf{z}) := \begin{bmatrix} \sum_{\mathbf{y} \in W(\mathbf{z})} I_u(\mathbf{y}) I_t(\mathbf{y}) \\ \sum_{\mathbf{y} \in W(\mathbf{z})} I_v(\mathbf{y}) I_t(\mathbf{y}) \end{bmatrix}$

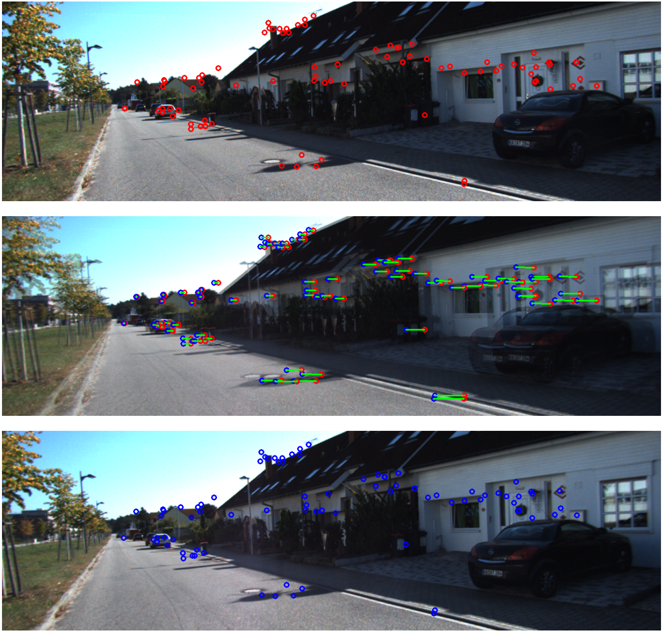
If $G(\mathbf{z})$ is invertible (guaranteed for point features), compute $\boldsymbol{\nu}(\mathbf{z}) = -G(\mathbf{z})^{-1} b(\mathbf{z})$

Else $\boldsymbol{\nu}(z) = 0$.

(Feature tracking) at time $t + 1$, repeat the operation at $\mathbf{z} + \boldsymbol{\nu}(z)$

(Optical flow) at time $t + 1$, repeat the operation at $\mathbf{z}$

Feature Tracking and Optical Flow



Feature Tracking and Optical Flow

- The feature tracking/optical flow algorithm is very efficient when we use the translational deformation model
- When features are tracked over extended periods of time, however, the estimation error accumulates
- Instead of matching image regions between adjacent frames, one could match image regions between an initial frame and the current frame
- The simple translational deformation model is no longer accurate and we should use the affine deformation model
- Further reading:
  - J. Shi and C. Tomasi, "Good features to track," IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 593-600, 1994.

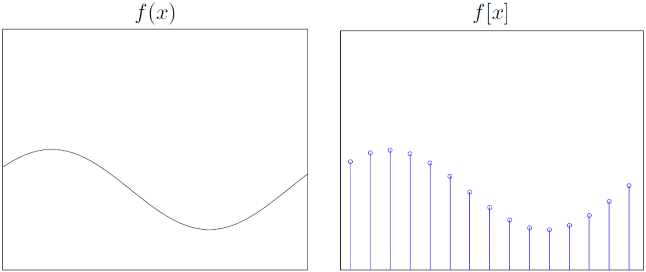
Image Gradient

- How do we compute the gradients $I_u(u,v,t)$, $I_v(u,v,t)$, and $I_t(u,v,t)$ needed for feature tracking/optical flow?
- We could approximate the derivatives using finite differences, e.g.,:

$$I_t(u,v,t) = I(u,v,t) - I_t(u,v,t-1) \quad \text{OR} \quad I_t(u,v,t) = \frac{1}{2}\left(I(u,v,t+1) - I_t(u,v,t-1)\right)$$

- To derive a more accurate approach we need to understand the relationship between a continuous signal $f(x)$ and its sampled version with period T:

$$f[x] = f(xT), \quad x \in \mathbb{Z}$$



Nyquist-Shannon Sampling Theorem

- If $f(x)$ is band limited, i.e., its Fourier transform satisfies $|F(\omega)| = 0$ for all $\omega > \omega_n$ (**Nyquist frequency**), it can be reconstructed exactly from a set of discrete samples at sampling frequency $\omega_s := \frac{2\pi}{T} > 2\omega_n$.
- The continuous signal $f(x)$ can be reconstructed by multiplying its sampled version $f[x]$ in the frequency domain with an ideal reconstruction filter $h(x)$ with Fourier transform:

$$H(\omega) = \begin{cases} 1, & \omega \in \left[-\frac{\pi}{T}, \frac{\pi}{T}\right] \\ 0, & \text{else} \end{cases} \qquad h(x) = \textbf{sinc}\left(\frac{\pi x}{T}\right), \quad x \in \mathbb{R}$$

- Multiplication in the frequency domain corresponds to convolution in the spatial domain, thus as long as $\omega_n < \frac{\pi}{T}$:

$$f(x) = f[x] * h(x), \qquad x \in \mathbb{R}$$

Derivative of a Sampled Signal

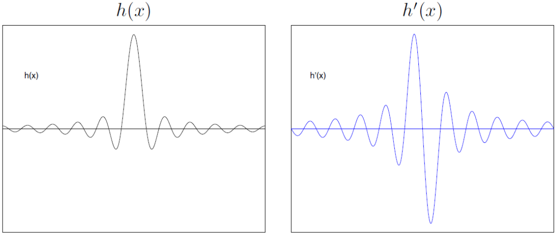
- Differentiating $f(x) = f[x] * h(x)$:

$$\frac{d}{dx}f(x) = \sum_{k=-\infty}^{\infty} f[k]\frac{d}{dx}h(x-k) = f[x] * \frac{dh}{dx}(x)$$

- Sampling the above result shows that the derivative of the sampled function $f'[x]$ can be computed as a convolution of the sampled signal $f[x]$ with the sampled derivative of the sync function $h'[x]$:

$$f'[x] = f[x] * h'[x]$$

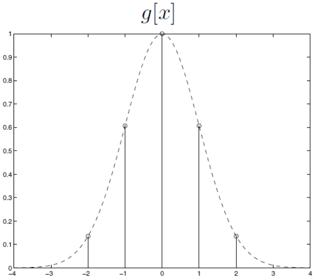
$$h'(x) = \frac{(\pi^2 x/T^2)\cos(\pi x/T) - \pi/T\sin(\pi x/T)}{(\pi x/T)^2}, \quad x \in \mathbb{R}$$

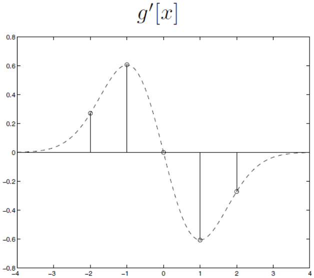


Five-tap Gaussian Filter

- The sync function has infinite support and falls off very slowly away from the origin. Hence, the sync convolution is not practically feasible and simple truncation yields undesirable artifacts.
- The derivative computation can be approximated by convolving with a Gaussian since it drops to zero much faster than the sync:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-x^2}{2\sigma^2}} \qquad g'(x) = -\frac{x}{\sigma^2\sqrt{2\pi\sigma^2}} e^{\frac{-x^2}{2\sigma^2}}$$





$g[x] = [0.1353 \quad 0.6065 \quad 1.0000 \quad 0.6065 \quad 0.1353]$ $\quad g'[x] = [0.2707 \quad 0.6065 \quad 0 \quad -0.6065 \quad -0.2707]$

Image Gradient

- In the case of images (2-D functions) the result is the same:

$$I(u,v) = I[u,v] * h(u,v) \qquad h(u,v) = h(u)h(v) = \frac{\sin(\pi u/T)\sin(\pi v/T)}{\pi^2 uv/T^2},$$

- Note that $h(u,v) = h(u)h(v)$ is separable which leads to:

$$I_u[u,v] = I[u,v] * h'[u] * h[v] \qquad I_v(u,v) = I[u,v] * h[u] * h'[v]$$

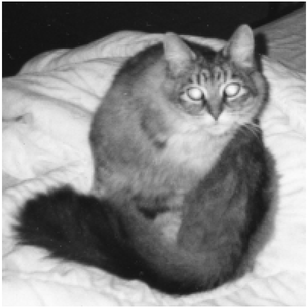
- The computation of the image derivatives is then accomplished as a pair of 1-D convolutions with filters obtained by sampling a continuous Gaussian function and its derivative:

$$I_u[u,v] = I[u,v] * g'[u] * g[v] = \sum_{k=-\omega/2}^{\omega/2} \sum_{l=-\omega/2}^{\omega/2} I[k,l] g'[u-k] g[v-l]$$

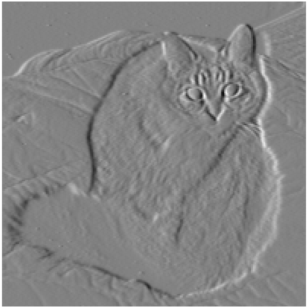
$$I_v[u,v] = I[u,v] * g[u] * g'[v] = \sum_{k=-\omega/2}^{\omega/2} \sum_{l=-\omega/2}^{\omega/2} I[k,l] g[u-k] g'[v-l]$$

- The number of samples is typically chosen as $\omega = 5\sigma$, imposing the fact that the window subtends 98.76% of the area under the Gaussian curve.

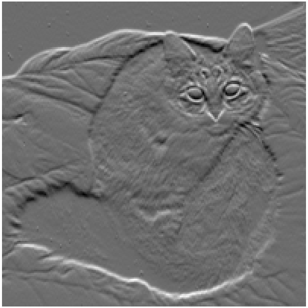
Image Gradient



I



I_u



I_v

Other Derivative Filters, Features, and Descriptors

- Other commonly used derivative filters:
  - **Interpolation filter**: $h[x] = \frac{1}{2}[1,\ 1]$ with derivative $h'[x] = \frac{1}{2}[1, -1]$
  - **Sobel filter**: $h[x] = \frac{1}{2+\sqrt{2}}[1, \sqrt{2}, 1]$ with derivative $h'[x] = \frac{1}{3}[1, 0, -1]$
  - **Gabor filter**: used for texture analysis
- Other features and descriptors (describe feature shape, color, texture):
  - **SIFT**: the Scale-Invariant Feature Transform (SIFT), introduced by David Lowe, is one of the most successful local image features/descriptors in the past decade. It makes the Harris corner scale invariant by using scale-space filtering via a Laplacian of Gaussian kernel (blob detector)
  - **SURF**: the Speeded-Up Robust Feature is a speeded-up version of SIFT which applies an approximate 2^{nd} derivative Gaussian filter at many scales along the axes and at 45° (more robust to rotation than Harris corners)
  - **FAST**: a Feature from Accelerated Segment Test detects corners by considering 16 pixels around the pixel y being tested and is several times faster than other corner detectors
  - **BRIEF**: a Binary Robust Independent Elementary Features speed up descriptor calculation and matching
  - **ORB**: Oriented FAST and Rotated BRIEF