ECE276A: Sensing & Estimation in Robotics
Lecture 12: Visual Features and Optical Flow

Lecturer:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Siwei Guo: s9guo@eng.ucsd.edu

Anwesan Pal: a2pal@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 $y_1 \in \mathbb{R}^2$ in the first image in the sense of being the same point in physical space
- **Idea**: look for a pixel $y_2 \in \mathbb{R}^2$ such that $I_2(y_2) \approx I_1(y_1)$

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 $W(y_1)$, $W(y_2)$ 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 y_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 $W(y_1)$ and image values $I_1(z)$, $z \in W(y_1)$, associated with a pixel y_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 $x \in \mathbb{R}^3$ that emits light with the same energy in all directions (Lambertian) and radiance distribution $\mathcal{R}(x)$
- Suppose the camera is calibrated (i.e., $K = I_{3\times 3}$) and the two camera frames are related by the rigid-body transformation $(R, p) \in SE(3)$.
- Let I_1 and I_2 be two images and $y_1, y_2 \in \mathbb{R}^2$ be the two pixels corresponding to x:

$$I_2(y_2) = I_1(y_1) \sim \mathcal{R}(x)$$

- From the projection equations, the point y_1 in image I_1 corresponds to the point y_2 in image I_2 if:

$$y_2 = h(y_1) := \frac{1}{\lambda_2}(\lambda_1 R y_1 + p)$$

  where λ_1, λ_2 are the **unknown** scales/depths of the observed point x.
- **Brightness constancy constraint**: $\boxed{I_1(y_1) = I_2(h(y_1))}$

Local Deformation Models

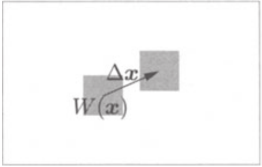
- The transformation h undergone by the entire image is determined by the scales λ_1, λ_2 of the visible surface and hence estimating h is as difficult as estimating the shape of the visible objects!
- Instead, we model the transformation only locally in a region $W(y)$:
  - **Translational model**: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

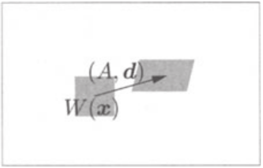
    $$h(z) \approx z + d, \quad \forall z \in W(y)$$

    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 $d \in \mathbb{R}^2$:

    $$h(z) \approx Az + d, \quad \forall z \in W(y)$$





Matching Point Features

- Requiring that $I_1(y_1) = I_2(h(y_1))$ is too much to ask for due to the approximation of h 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 h:

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

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

$$I_1(y) = I(y(t), t) \underbrace{\approx}_{\text{brightness constancy}} I_2(h(y)) \underbrace{\approx}_{\text{translation model}} I(y(t) + \nu dt, t + dt)$$

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

Continuous-Time Brightness Constancy

- **Brightness Constancy** (for the affine model):
  $I(y,t) \approx I(Ay + \nu dt, t + dt)$

- Linearizing the right-hand side around (y,t):

  $$I(Ay + \nu dt, t + dt) \approx I(y,t) + \nabla_y I(y,t)^T (Ay + \nu dt - y) + \frac{\partial I}{\partial t}(y,t) dt$$

  leads to:
  - Translational: $\min_{\nu} \sum_{z \in W(y)} \left\| \nabla_y I(z,t)^T \nu + \frac{\partial I}{\partial t}(z,t) \right\|_2^2$
  - Affine: $\min_{A,\nu} \sum_{z \in W(y)} \left\| \nabla_y I(z,t)^T \left(\frac{(A-I)}{dt} z + \nu \right) + \frac{\partial I}{\partial t}(z,t) \right\|_2^2$

- **Aperture problem**: The brightness constancy equation $\left(\frac{\partial I}{\partial y}\nu + \frac{\partial I}{\partial t} = 0\right)$ provides only one constraint for the two unknowns $\nu \in \mathbb{R}^2$.

- There are enough constraints on ν only when the brightness constancy constraint is applied to each z in a region $W(y)$ that contains "sufficient texture" and the motion ν is assumed constant in the region

Feature Tracking and Optical Flow

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

Feature Tracking and Optical Flow

- To compute the velocity ν we need to solve:

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

- Letting $y = (u, v)$ and setting the gradient to zero results in:

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

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

Point Feature Selection

- For $G(y)$ to be invertible, the region $W(y)$ must have nontrivial gradients along independent directions, therefore resembling a "corner" structure.

- **Corner feature**: a pixel y such that the smallest singular value of $G(y)$ (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 $G(y) := \begin{bmatrix} \sum_{z \in W(\mathbf{y})} I_u^2(z) & \sum_{z \in W(\mathbf{y})} I_u(z) I_v(z) \\ \sum_{z \in W(\mathbf{y})} I_u(z) I_v(z) & \sum_{z \in W(\mathbf{y})} I_v^2(z) \end{bmatrix}$ at every pixel $y = (u, v)$

(Feature tracking) select point features $y_1, y_2, \ldots$ such that $G(y_i)$ is invertible

(Optical flow) select y_i on a fixed grid

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

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

Else $\nu(y) = 0$.

(Feature tracking) at time $t + 1$, repeat the operation at $y + \nu(y)$

(Optical flow) at time $t + 1$, repeat the operation at y

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.

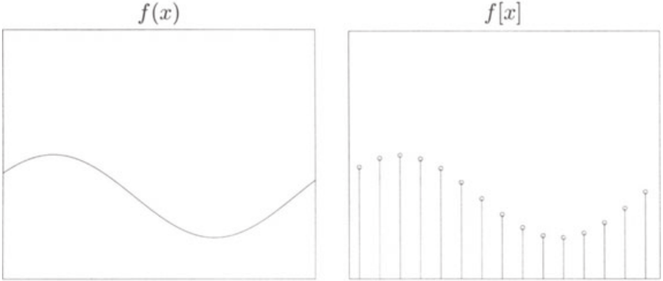
Image Gradient

- 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.,:

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

- 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 $\omega_n(f) < \frac{\pi}{T}$:

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

Derivative of a Sampled Signal

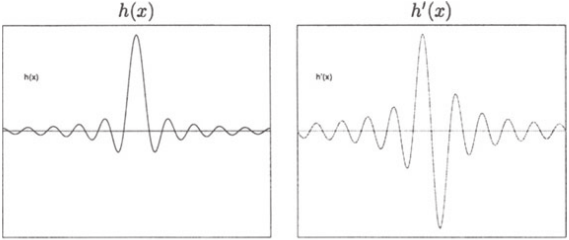
- Differentiating $f(x) = f[x] * h(x)$:

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

- 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 $h'[x]$:

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

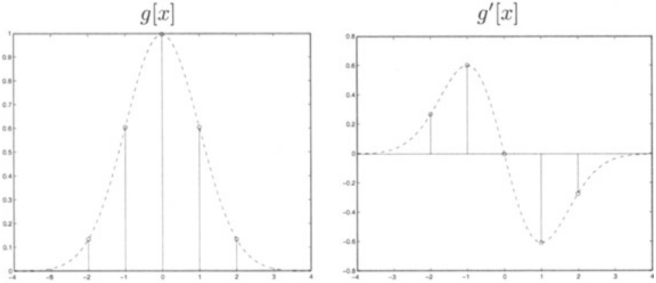
$$h'(x) = \frac{(\pi^2 x/T^2)\cos(\pi x/T) - \pi/T \sin(\pi x/T)}{(\pi x/T)^2}, \quad x \in \mathbb{R}$$



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:

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



$$g[x] = \begin{bmatrix} 0.1353 & 0.6065 & 1.0000 & 0.6065 & 0.1353 \end{bmatrix} \quad g'[x] = \begin{bmatrix} 0.2707 & 0.6065 & 0 & -0.6065 & -0.2707 \end{bmatrix}$$

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:

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

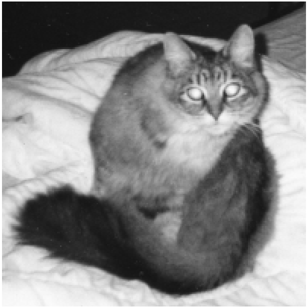
- The computation of the image derivatives is then accomplished as a pair of 1-D convolutions with filters obtained by sampling a continuous Gaussian function and its derivative:

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

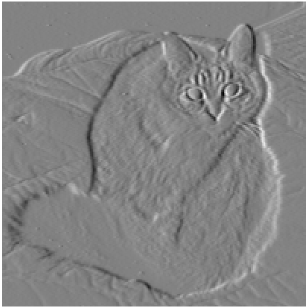
$$I_v[u,v] = I[u,v] * g[u] * g'[v] = \sum_{k=-\omega/2}^{\omega/2} \sum_{l=-\omega/2}^{\omega/2} I[k,l] g[u-k] g'[v-l]$$

- 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.

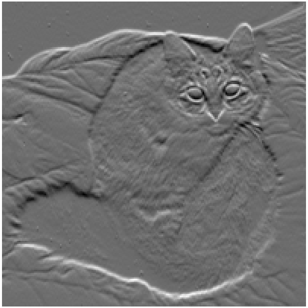
Image Gradient



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

Epipolar Geometry

- Let $x \in \mathbb{R}^3$ (world frame) be observed by two calibrated cameras
- Without loss of generality assume that the first camera frame coincides with the world frame. Let the position and orientation of the second camera be $p \in \mathbb{R}^3$ and $R \in SO(3)$
- The images of x in the two camera frames are:

$$\begin{aligned} \lambda_1 y_1 &= x, & \lambda_1 &= \text{unknown scale} \\ \lambda_2 y_2 &= R^T(x - p), & \lambda_2 &= \text{unknown scale} \end{aligned}$$

- We obtain the following relationship between the image points:

$$\lambda_1 y_1 = R\lambda_2 y_2 + p$$

- To eliminate the unknown depths λ_i:
  - pre-multiply with $\hat{p}$
  - note that $\hat{p}y_1$ is perpendicular to y_1

$$\underbrace{\lambda_1 y_1^T \hat{p} y_1}_{0} = \lambda_2 y_1^T \hat{p} R y_2 + \underbrace{y_1^T \hat{p} p}_{0}$$

Essential Matrix

- Thus, $\lambda_2 y_1^T \hat{p} R y_2 = 0$ and since $\lambda_2 > 0$, we arrive at the following result
- **Epipolar constraint**: Consider two images y_1, y_2 of the same point x from two calibrated cameras with relative pose (R, p). Then:

$$0 = y_1^T \hat{p} R y_2 = y_1^T E y_2$$

where $E := \hat{p} R \in \mathbb{R}^{3\times3}$ is the **essential matrix**.

- **Essential matrix characterization**: a non-zero $E \in \mathbb{R}^{3\times3}$ is an essential matrix iff its singular value decomposition is $E = U\mathbf{diag}(\sigma, \sigma, 0)V^T$ for some $\sigma \geq 0$ and $U, V \in SO(3)$
- **Pose recovery from the Essential matrix**: There are exactly two relative poses corresponding to a non-zero essential matrix E:

$$(\hat{p}, R) = \left(UR_z\left(\frac{\pi}{2}\right)\mathbf{diag}(\sigma, \sigma, 0)U^T, UR_z^T\left(\frac{\pi}{2}\right)V^T\right)$$

$$(\hat{p}, R) = \left(UR_z\left(-\frac{\pi}{2}\right)\mathbf{diag}(\sigma, \sigma, 0)U^T, UR_z^T\left(-\frac{\pi}{2}\right)V^T\right)$$