Today’s Topics

9. Unifying view of derivative estimation, template matching, smoothing, interpolation & Laplacian computation

10. The SIFT keypoint detector & descriptor

Announcements

• A4 is due April 4

• Marks for A2 are starting to come. Will be available by Friday (apologies from our side, lateness due to a sick TA)

• Grace-days left will be posted on Blackboard tomorrow

• May use grace-days after April 4, keep an eye on the final exam (worth 4 times the value of the assignment).
Topic 9: A Unifying View:

1. Template matching ⇔ Derivatives via WLS fitting
2. Image smoothing ⇔ Template matching
3. Image interpolation ⇔ Convolution w/ continuous smoothing function
4. Image differentiation ⇔ Convolution w/ derivative of a smoothing function
5. Image Laplacian ⇔ Difference of two Gaussian-smoothed versions of an image
Convolution

Template Matching

I, δ₁, δ²₁,... via WLS

Correlation

Convolution

Image Smoothing

Image Interpolation

Image Differentiation

Image Laplacian

Unifying View

WLS Estimation ↔ Cross Correlation

Patch (2w+1 pixels)
WLS Estimation $\iff$ Cross Correlation

$W_\omega$ weights

$X$: $x, x^2, 1/6x^3,$...

d: derivatives

I: image contents
WLS Estimation $\Leftrightarrow$ Cross Correlation

\[
\text{Patch (2w+1 pixels)}
\]

\[
\begin{bmatrix}
\mathbf{a}_1 & 0 \\
0 & \mathbf{2}_{2w+1}
\end{bmatrix}
\mathbf{I} =
\begin{bmatrix}
\mathbf{a}_1 & 0 \\
0 & \mathbf{2}_{2w+1}
\end{bmatrix}
\mathbf{X} \mathbf{d}
\]

\[\mathbf{WXd} = \mathbf{WI} \implies (\mathbf{WX})^T \mathbf{WX} \mathbf{d} = (\mathbf{WX})^T \mathbf{W} \mathbf{I} \]
\[\implies \mathbf{d} = [(\mathbf{WX})^T (\mathbf{WX})]^{-1} (\mathbf{WX})^T \mathbf{W} \mathbf{I}
\]

\[
\text{Solution: } \mathbf{d} = 
\begin{bmatrix}
\mathbf{f}(0) \\
\mathbf{f}'(0) \\
\vdots \\
\mathbf{f}^{(n)}(0)
\end{bmatrix} = 
\begin{bmatrix}
\mathbf{I}_1 \\
\mathbf{I}_2 \\
\vdots \\
\mathbf{I}_{2w+1}
\end{bmatrix} \cdot 
\begin{bmatrix}
\mathbf{I}_1 \\
\mathbf{I}_2 \\
\vdots \\
\mathbf{I}_{2w+1}
\end{bmatrix}^{-1}
\]
WLS Estimation $\Leftrightarrow$ Cross Correlation

\[
\begin{bmatrix}
\rho_1 & 0 \\
0 & \rho_{2_{\text{var}}} & \\
\end{bmatrix}
\begin{bmatrix}
\Gamma_1 \\
\Gamma_{2_{\text{var}}} \\
\end{bmatrix}
\begin{bmatrix}
X_d \\
\end{bmatrix}
\]

\[
WXd = WI \Rightarrow (\mathbf{WX})^T(\mathbf{WX})d = (\mathbf{WX})^TWI
\]

\[
d = \left(\mathbf{WX}^T(\mathbf{WX})\right)^{-1}(\mathbf{WX})^TWI
\]

One row of this matrix

Solution: 

\[
d = 
\begin{bmatrix}
\mathbf{I}_1 \\
\vdots \\
\mathbf{I}_{n_{\text{var}}} \\
\end{bmatrix}
\begin{bmatrix}
(t_0^T) \\
(t_1^T) \\
\vdots \\
(t_{n_{\text{var}}}^T) \\
\end{bmatrix}
\begin{bmatrix}
\mathbf{I}_1 \\
\vdots \\
\mathbf{I}_{n_{\text{var}}} \\
\end{bmatrix}
\]

WLS Estimation $\Leftrightarrow$ Cross Correlation

\[
\begin{bmatrix}
\rho_1 & 0 \\
0 & \rho_{2_{\text{var}}} & \\
\end{bmatrix}
\begin{bmatrix}
\Gamma_1 \\
\Gamma_{2_{\text{var}}} \\
\end{bmatrix}
\begin{bmatrix}
X_d \\
\end{bmatrix}
\]

\[
WXd = WI \Rightarrow (\mathbf{WX})^T(\mathbf{WX})d = (\mathbf{WX})^TWI
\]

\[
d = \left(\mathbf{WX}^T(\mathbf{WX})\right)^{-1}(\mathbf{WX})^TWI
\]

One row of this matrix

Solution: 

\[
d = 
\begin{bmatrix}
\mathbf{I}_1 \\
\vdots \\
\mathbf{I}_{n_{\text{var}}} \\
\end{bmatrix}
\begin{bmatrix}
(t_0^T) \\
(t_1^T) \\
\vdots \\
(t_{n_{\text{var}}}^T) \\
\end{bmatrix}
\begin{bmatrix}
\mathbf{I}_1 \\
\vdots \\
\mathbf{I}_{n_{\text{var}}} \\
\end{bmatrix}
\]
I(w=0) can be computed as the cross-correlation of the image patch (I) and the “template” t₀.

\[
: d = \begin{bmatrix} f(0) \\ \frac{df}{dx}(0) \\ \frac{df}{dy}(0) \end{bmatrix} = \begin{bmatrix} (t^0) \top \\ (t^1) \top \end{bmatrix} \cdot \begin{bmatrix} I_0 \\ I_1 \\ \vdots \\ I_{2w+1} \end{bmatrix}
\]

\[
I^\top \cdot t^0 = CC(I, t^0)
\]
One can think of the estimation of $\frac{d}{dx} (c)$ as the “similarity” value between the image patch $x_i$ and the template $t^i$. 

**Similarity function**

Cross- Correlation

$CC (x_i, t^i) = X_i^\top \cdot t^i$
Unifying View

- Image Smoothing
- Template Matching
- Correlation
- Convolution
- Image Interpolation
- Image Differentiation
- Image Laplacian

Gaussians in 1D and 2D

1D Gaussian

\[ G(x) = \frac{1}{\sqrt{2\pi}\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right) \]
Gaussians in 1D and 2D

1D Gaussian

\[ G_x(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-m)^2}{2\sigma^2}} \]

2D Gaussian

\[ G_y(r, c) = G_x(r)G_z(c) = \frac{1}{2\pi\sigma^2} e^{-\frac{(r-m_x)^2 + (c-m_y)^2}{2\sigma^2}} \]

Example: Applying Gaussian Smoothing

No smoothing
Example: Applying Gaussian Smoothing

No smoothing  \( \sigma = 2 \)

Example: Applying Gaussian Smoothing

No smoothing  \( \sigma = 2 \)  \( \sigma = 4 \)
Gaussian Smoothing ⇔ Cross-Correlation

Gaussian smoothing: estimate the Image intensity term as the similarity between the image patch and a Gaussian template

\[
\text{Gaussian Template } G_6
\]

\[
\text{Image patch } X_i
\]

Prove this:

\[
CC(X_i, G_6) = \frac{1}{\sum G_6(x)} \cdot \frac{1}{G_6(x)}
\]

For smoothing operations, we want to compute a weighted average, so we use the scaled template

\[
\overline{G_6} = \frac{1}{\sum G_6(x)} \cdot G_6(x)
\]

\[
X_i
\]
Averaging vs. Gaussian Smoothing

Result of Cross-Correlation with Gaussian Mask

Unifying View

Image Smoothing
Template Matching
I, δI, δ²I,... via WLS
Image Interpolation
Correlation
Convolution
Image Differentiation
Image Laplacian
Image Interpolation: Definition

Given a discrete set of (intensity) values $I_0, I_1, ..., I_{M-1}$ (also called “samples”), define a continuous function $I(x)$ that can be evaluated at any real $x \in [0, M-1]$. 

![Diagram of Image Interpolation]

Intensity

Pixel (x)

Image Interpolation: Definition

Given a discrete set of (intensity) values $I_0, I_1, ..., I_{M-1}$ (also called “samples”), define a continuous function $I(x)$ that can be evaluated at any real $x \in [0, M-1]$. 

![Diagram of Image Interpolation]

Intensity

Pixel (x)
Interpolation applications:

Image warping
Image Interpolation: Applications

Interpolation applications:

- Image warping

Design of differentiation templates

\[ (I * T)(x) = \sum_{x=0}^{M-1} I_k T(x - k) \]

The result of interpolation is a function that can be evaluated at non-integer values of \( x \).
Interpolation: General Expression

\[(I \ast T)(x) = \sum_{x=0}^{M-1} I_k T(x - k)\]

The result of interpolation is a function that can be evaluated at non-integer values of \(x\).

The general expression above allows \(T\) to be any arbitrary function defined for \(x-k \in [-m, m]\).

In practice, \(T\) is chosen to satisfy additional properties (e.g. differentiability and \(\int T(x) dx = 1\)).
Example #1: Interpolation Using Gaussian Kernel

\[(I * G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x - k)\]

\(I_k\): intensity at k-th pixel
\(x - k\): distance between x and k pixel
\(G_\sigma(x - k)\): contribution of the k-th pixel

\[(I * G_\sigma)(x) = \text{weighted combination of } I[0], \ldots, I[M-1]\]
Example #2: Interpolation Using Linear Kernel

\[(I \ast L)(x) = \sum_{k=0}^{M-1} I_k L(x - k)\]

\[(I \ast L)(x) = \text{weighted combination of }\]
\[I[0], I[1], ..., I[M-1]\]
Example #2: Interpolation Using Linear Kernel

\[(I * L)(x) = \sum_{k=0}^{M-1} I_k L(x - k)\]

\[ (I * L)(x) = \text{weighted combination of} \]
\[ I[0], \ldots, I[M-1] \]

\[ (I * L)(x) = I[5] \cdot L(x - 5) + I[6] \cdot L(x - 6) \]

Unifying View

Image Smoothing
Template Matching
\[ I, \partial I, \partial^2 I, \ldots \text{via} \ WLS \]
Image Interpolation
Image Differentiation
Image Laplacian
Image differentiation (derivatives computation)

Step 1: Interpolate to define a continuous function

\[(I * G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x - k)\]

Step 2: Take the derivative of this continuous function

\[\frac{\delta}{\delta x} (I * G_\sigma)(x)\]

---

Step #1: Interpolate Using Gaussian Kernel

\[(I * G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x - k)\]

To interpolate, we evaluate the expression at continuous values \(x\).
Step #1: Interpolate Using Gaussian Kernel

\[ (I * G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x - k) \]

To interpolate, we evaluate the expression at continuous values \( x \).

Because \( (I * G_\sigma)(x) \) is a weighted sum of Gaussians, we can compute its derivative analytically.

Step #2: Differentiate the Interpolated Image

\[ \frac{d}{dx} (I * G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x - k) \]

\( \sigma \) is continuous
\( k \) is discrete
Step #2: Differentiate the Interpolated Image

\[
(I \ast G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x-k)
\]

\[
\frac{d}{dx} (I \ast G_\sigma)(x) = \frac{d}{dx} \left[ \sum_{\kappa = \infty}^{M-1} I_{\kappa} \cdot G_\sigma(x-\kappa) \right]
\]

\[
\sum_{\kappa = \infty}^{M-1} I_{\kappa} \cdot \frac{d}{dx} G_\sigma(x-\kappa) \iff \frac{d}{dx} (I \ast G_\sigma)(x) = \left[ I \ast \left( \frac{d}{dx} G_\sigma \right) \right](x)
\]

Step #2: Differentiate the Interpolated Image

\[
(I \ast G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x-k)
\]

\[
\frac{d}{dx} (I \ast G_\sigma)(x) = \frac{d}{dx} \left[ \sum_{\kappa = \infty}^{M-1} I_{\kappa} \cdot G_\sigma(x-\kappa) \right]
\]

\[
\sum_{\kappa = \infty}^{M-1} I_{\kappa} \cdot \frac{d}{dx} G_\sigma(x-\kappa) \iff \frac{d}{dx} (I \ast G_\sigma)(x) = \left[ I \ast \left( \frac{d}{dx} G_\sigma \right) \right](x)
\]
Image Differentiation $\Leftrightarrow$ Convolution w/ Gaussian Derivative

$$(I * G_\sigma)(x) = \sum_{x=0}^{M-1} I_k G_\sigma(x - k)$$

We can compute derivatives by applying a template that is the derivative of the Gaussian function.

$$\frac{d}{dx} (I * G_\sigma)(x) = \left[ I * \left( \frac{d}{dx} G_\sigma \right) \right](x)$$

Convolution with the Derivative of a Gaussian

Gaussian

$$G_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}}$$
Convolution with the Derivative of a Gaussian

Gaussian

\[ G_{\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} \]

First derivative

\[ G'_{\sigma}(x) = \frac{\delta}{\delta x} \left( \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} \right) \]
\[ = -\frac{2x}{2\sigma^2} \left( \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} \right) \]
\[ = -\frac{2x}{2\sigma^2} G_{\sigma}(x) \]

Familiar?

Image Smoothing
Template Matching
Image Interpolation
Image Differentiation
Image Laplacian
1, \delta I, \delta^2 I, ... via WLS
**Convolution with the Derivative of a Gaussian**

This is the plot of $G'_\sigma(x)$

\[
\frac{\delta I}{\delta x}(0) = \frac{1}{\sum_{i=-5}^{5} G'(i)} [G'(-5), \ldots, G'(5)] * [I_1, I_2, \ldots, I_{11}]
\]

**Convolution with the Derivative of a Gaussian**

Comparing the 1D kernel for estimating $\frac{dI}{dx}$

1. **using WLS polyfit with Gaussian weights**
2. **using convolution with $\frac{dG}{dx}$**

\[
\frac{dI}{dx}(t) = [t, t', \ldots, t^n] \begin{bmatrix} I_1 & I_2 & \cdots & I_{11} \end{bmatrix}
\]

\[
\frac{\delta I}{\delta x}(0) = \frac{1}{\sum_{i=-5}^{5} G'(i)} [G'(-5), \ldots, G'(5)] * [I_1, I_2, \ldots, I_{11}]
\]
Convolution with the Derivative of a Gaussian

\[(I \ast G_\sigma)(x) = \sum_{\kappa=0}^{M-1} I[\kappa] \cdot G_\sigma(x - \kappa)\]

We can compute derivatives by applying a template that is the derivative of the Gaussian function:

\[\frac{\partial}{\partial x} (I \ast G_\sigma)(x) = \left[ I \ast \left( \frac{d}{\partial x} G_\sigma \right) \right](x)\]

Convolution with the Derivative of a Gaussian

\[G''_\sigma(x) = \frac{\delta^2}{\delta x^2} \left( \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \right)\]

\[= \left( \frac{x^2}{\sigma^2} - 1 \right) \left( \frac{1}{\sigma^2} \right) \left( \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \right)\]

\[= \left( \frac{x^2}{\sigma^2} - 1 \right) \left( \frac{1}{\sigma^2} \right) G_\sigma(x)\]

\[\frac{d}{\partial x} (I \ast G_\sigma)(x) = \left[ I \ast \left( \frac{d}{\partial x} G_\sigma \right) \right](x)\]
The advantages of estimating derivatives using a convolution with a Derivative of a Gaussian template are:

- The Derivative of $G_{\sigma}$ can be computed analytically.
- One can control the size of $\sigma$ (the scale of the smoothing).
- One can compute higher order derivatives by convolving with $\frac{d^n}{dx^n} G_{\sigma}(x)$.
- Computations are always dot products (i.e. efficient).
Principle #5

Subtracting Gaussian-smoothed versions of an image $I$ at nearby scales $\sigma_1$ and $\sigma_2$.

Computing the Laplacian of $I$. 

$I$, $\delta I$, $\delta^2 I$, ... via WLS
What does smoothing take away?

$I \ast G_{e_1}$

What Does Smoothing Take Away?

$I \ast G_{e_2}$
The Difference-Of-Gaussians (DOG) Filter

\[ I * G_{\sigma_1} - I * G_{\sigma_2} = I * (G_{\sigma_1} - G_{\sigma_2}) \]

This is called a DOG filter (Diferencie of Gaussians)
Consider $G$ to be a function of both $x$ and $\sigma$.

Equivalence of DOG and 2\textsuperscript{nd} Derivative Filter

What is $I \ast (G_{\sigma_1} - G_{\sigma_2})$?
What is $G_{\sigma_1} - G_{\sigma_2}$?

$G_{\sigma_1} - G_{\sigma_2} = (\sigma_2 - \sigma_1) \frac{\delta G_\sigma}{\delta \sigma} (x, \sigma_1)$
Equivalence of DOG and 2nd Derivative Filter

What is \( I \ast (G_{\sigma_1} - G_{\sigma_2}) \)?
What is \( G_{\sigma_1} - G_{\sigma_2} \)?

Consider \( G \) to be a function of both \( x \) and \( \sigma \).

\[
G_{\sigma_1} - G_{\sigma_2} = (\sigma_2 - \sigma_1) \frac{\delta G}{\delta \sigma}(x, \sigma_1)
\]

Compare to the 2nd derivative of \( G \) with respect to \( x \).

The DOG filter is just a scaled version of the Gaussian 2nd derivative filter.

Equivalence of DOG and 2nd Derivative Filter

What is \( I \ast (G_{\sigma_1} - G_{\sigma_2}) \)?
What is \( G_{\sigma_1} - G_{\sigma_2} \)?

To answer, consider \( G \) to be a function of both \( x \) and \( \sigma \).

Using approximate differences, the derivative can be computed as:

\[
\frac{\delta G(x, \sigma)}{\delta \sigma} = \frac{G(x, \sigma_2) - G(x, \sigma_1)}{\sigma_2 - \sigma_1}
\]

From where:

\[
G_{\sigma_1} - G_{\sigma_2} = (\sigma_2 - \sigma_1) \frac{\delta G}{\delta \sigma}(x, \sigma_1)
\]

Compare \( \frac{\delta^2 G}{\delta \sigma^2} \) and \( \frac{\delta G}{\delta x^2} \).

\[
\frac{\partial^2 G}{\partial x^2} = \left( \frac{x^2}{\sigma^4} - \frac{1}{6} \right) \frac{\partial G}{\partial \sigma}
\]

From where:

\[
(G_{\sigma_1} - G_{\sigma_2}) \ast (\delta^2 G_{\sigma_1}) = \left( \delta^2 G_{\sigma_1} \right) \ast (G_{\sigma_1} - G_{\sigma_2})
\]
The Difference-Of-Gaussians (DOG) Filter

What is $I \ast (G_{G_1} - G_{G_2})$?

What is $G_{G_1} - G_{G_2}$?

In two dimensions:

$$G_{G_2}(x,y) - G_{G_1}(x,y) = G_1(G_{G_1} - G_{G_2}) \nabla^2 G_{G_1}(x,y)$$

Image Laplacian from the difference of two Gaussians:

$$\nabla^2 (I \ast G_{G_1}) = I \ast \nabla^2 G_{G_1} = (I \ast G_{G_2} - I \ast G_{G_1}) \frac{1}{G(G_{G_1} - G_{G_2})}$$

Unifying View

Image Smoothing  Template Matching  $I, \delta I, \delta^2 I, \ldots$ via WLS

Correlation  Convolution

Image Interpolation  Image Differentiation  Image Laplacian
Topic 10: Feature Detection & Image Matching

- Introduction to the image matching problem
- Image matching using SIFT features
- The SIFT feature detector
- The SIFT descriptor

SIFT

SIFT: Scale Invariant Feature Transform
Developed by David Lowe in 1999
One of the most powerful representations for feature detection and matching.
Widely used in applications that range from robotics to image retrieval and recognition, image stitching, camera calibration and video analysis.
The Image Matching Problem

Goal: to identify “features” or patches in image I, that appear in another image I’

Lines indicate a correspondence between location (x, y) in image I, and location (x’, y’) in image I’.
The Image Matching Problem

Is it possible to solve this problem by direct template matching between the two images?

The Image Matching Problem

Is it possible to solve this problem by direct template matching between the two images?
The Image Matching Problem

Is it possible to solve this problem by direct template matching between the two images?

Yes, but it would be impossibly inefficient.
(i.e. must search over all possible pairs of patches)

Feature-Based Image Matching

Feature detection & matching
Feature-Based Image Matching

Detect features in I → Match features across the two images
Detect features in I’

Errors in Feature-Based Image Matching

- In general, some/most of these correspondences may be incorrect.
  - Two types of error:
    1. False positive matches: algorithm returns a correspondence between 2 locations where none exists.
    2. False negative matches: algorithm fails to detect a correspondence between two instances of the same feature/patch.
Errors in Feature-Based Image Matching

Goal: Minimize false positives and false negatives across a wide range of imaging conditions.

1. False positive matches: Algorithm returns a correspondence between 2 locations where none exists.

2. False negative matches: Algorithm fails to detect a correspondence between two instances of the same feature/patch.

Evaluating a Feature Detector’s Performance

- Ideal performance
- Good performance
- Poor performance

Correct matches (as fraction of total) vs. incorrect matches (false positives as fraction of total)
Feature Matching & Transformation Invariance

A useful feature detector and matching algorithm must be insensitive to a wide range of image transformations.

Transformation-Invariant Feature Detectors

A feature detector is called invariant to a certain image transformation if it can reliably detect features in a transformed version of the source image.
A feature detector is called invariant to a certain image transformation if it can reliably detect features in a transformed version of the source image.
A feature detector is called invariant to a certain image transformation if it can reliably detect features in a transformed version of the source image.

**Topic 10:**

**Feature Detection & Image Matching**

- Introduction to the image matching problem
- Image matching using SIFT features
- The SIFT feature detector
- The SIFT descriptor
We’ll define the SIFT descriptor next class.