Image Patches
Image Patch Features

• We often want to find distinctive points in the image
  – Could match these points to a 3D model, for object recognition
  – Or track them from one image to another, for motion or structure estimation

• A point can be described by the appearance of a small image “patch” surrounding the point

Patches are 15x15 pixels
We’ll see that the best patches to use are locally unique
  – Good types of features: bright dots, corners
  – Bad types of features: regions with constant value, or long straight edges

In general, there are 4 steps:
  – Feature detection
  – Feature description
  – Feature matching
  – Feature tracking

First let’s look at a simple feature matching method
  – This will help give motivation for the first two steps
Sum of Squared Differences (SSD)

- Say we want to match a patch from image $I_0$ to image $I_1$
  - We’ll assume that intensities don’t change, and there is only translational motion within the image
  - So we expect that
    $$I_1(x_i + u) = I_0(x_i)$$
  - In practice we will have noise, so they won’t be exactly equal
- But we can search for the displacement $u=(x,y)$ that minimizes the sum of squared differences
  $$E(u) = \sum_i w(x_i) \left[ I_1(x_i + u) - I_0(x_i) \right]^2$$

$w(x_i)$ is an optional weighting function, that weights the center of the patch higher than the periphery
Cross Correlation

• Expanding the expression for SSD:

\[ E(u) = \sum_i [I_1(x_i + u) - I_0(x_i)]^2 \]

\[ = \sum_i I_1(x_i + u)^2 - 2\sum_i I_1(x_i + u)I_0(x_i) + \sum_i I_0(x_i)^2 \]

• The middle term is the cross-correlation of \( I_1 \) and \( I_0 \) ... it is high when \( I_1 \) matches \( I_0 \)

• Under certain conditions minimizing the SSD is equivalent to maximizing the cross-correlation
  – Cross-correlation can be done very efficiently (in the frequency domain)
Vector representation of correlation

- Correlation is a sum of products of corresponding terms
  \[ c(x, y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} w(s, t) f(x + s, y + t) = w(x, y) \cdot f(x, y) \]

- We can think of correlation as a dot product of vectors \( \mathbf{w} \) and \( \mathbf{f} \)

\[ c = w_1 f_1 + w_2 f_2 + \ldots + w_{mn} f_{mn} = \mathbf{w} \cdot \mathbf{f} \]

- If images \( w \) and \( f \) are similar, their vectors (in \( mn \)-dimensional space) are aligned

\[ c = |\mathbf{w}| |\mathbf{f}| \cos \theta \]
Correlation can do matching

- Let \( w(x, y) \) be a template sub-image
- Try to find an instance of \( w \) in image \( f(x, y) \)
- The correlation score \( c(x, y) \) is high where \( w \) matches \( f \)

\[
c(x, y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} w(s, t) f(x + s, y + t) = w(x, y) \otimes f(x, y)
\]
Template Matching (continued)

• Since \( f \) is not constant everywhere, we need to normalize

\[
c = \frac{\mathbf{w} \cdot \mathbf{f}}{\|\mathbf{w}\| \|\mathbf{f}\|} = \cos \theta
\]

• We can get better precision by subtracting off means

\[
c(x, y) = \sum_{s,t} \left[ w(s, t) - \bar{w} \right] \left[ f(x + s, y + t) - \bar{f} \right]
\]

\[
c(x, y) = \frac{\sum_{s,t} \left[ w(s, t) - \bar{w} \right] \sum_{s,t} \left[ f(x + s, y + t) - \bar{f} \right]}{\left( \sum_{s,t} \left[ w(s, t) - \bar{w} \right]^2 \sum_{s,t} \left[ f(x + s, y + t) - \bar{f} \right]^2 \right)^{1/2}}
\]

This is the normalized cross correlation coefficient

Range: \(-1.0 \ldots +1.0\)

Perfectly opposite

Perfect match
Matlab example

- Track a point from one image to the next
  - `imcrop` – get template
  - `normxcorr2` – normalized cross correlation
Note on normxcorr2

• The Matlab function normxcorr2 correlates a template $T$ with image $I$
  – To find the maximum, can do
    
    $C = \text{normxcorr2}(T,I); \quad % \text{Do normalized cross correlation}$
    $c_{\text{max}} = \max(C(:));$
    $[y_2 \ x_2] = \text{find}(C == c_{\text{max}});$

• Assume $T$ is size $(2*M+1)x(2*M+1)$
  – Then the resulting score image is bigger than $I$, by $M$ rows and $M$ columns along each side and top and bottom
  – You should subtract off $M$ from the coordinates
    
    $y_2 = y_2 - M;$
    $x_2 = x_2 - M;$
Example - Track a point across a sequence

- Sequence of images “building.avi”

Which points are the best to track?
Pick point to track using mouse

• \([x, y] = \text{ginput}(1);\)
  – Allows you to pick one point with the mouse (crosshairs)
  – Returns \(x, y\) location of picked point
% Track a point
clear all
close all

movieObj = VideoReader('building.avi'); % open file
nFrames = movieObj.NumberOfFrames;

I0 = read(movieObj,1); % Get first image
I0 = rgb2gray(I0);
imshow(I0, []);
[x, y] = ginput(1);
x = round(x);
y = round(y);

% Size of window
M = 9;
rectangle('Position', [x-M y-M 2*M+1 2*M+1], 'EdgeColor', 'r');

% Extract template
T = I0(y-M:y+M, x-M:x+M);
figure, imshow(T, []);
pause

% Read images one at a time:
for i=2:nFrames
    img = read(movieObj,i); % get one RGB image
    img = rgb2gray(img);
    imshow(img, []);

    C = normxcorr2(T, img);
cmax = max(C(:));
[y2 x2] = find(C==cmax);

    y2 = y2-M;
x2 = x2-M;
    rectangle('Position', [x2-M y2-M 2*M+1 2*M+1], 'EdgeColor', 'r');
    fprintf('Correlation score = %f\n', cmax);
pause
end

• This program lets you pick a point in the first image
• It then extracts a patch of size (2*M+1)x(2*M+1) surrounding that point
• It then tries to match that patch to subsequent images