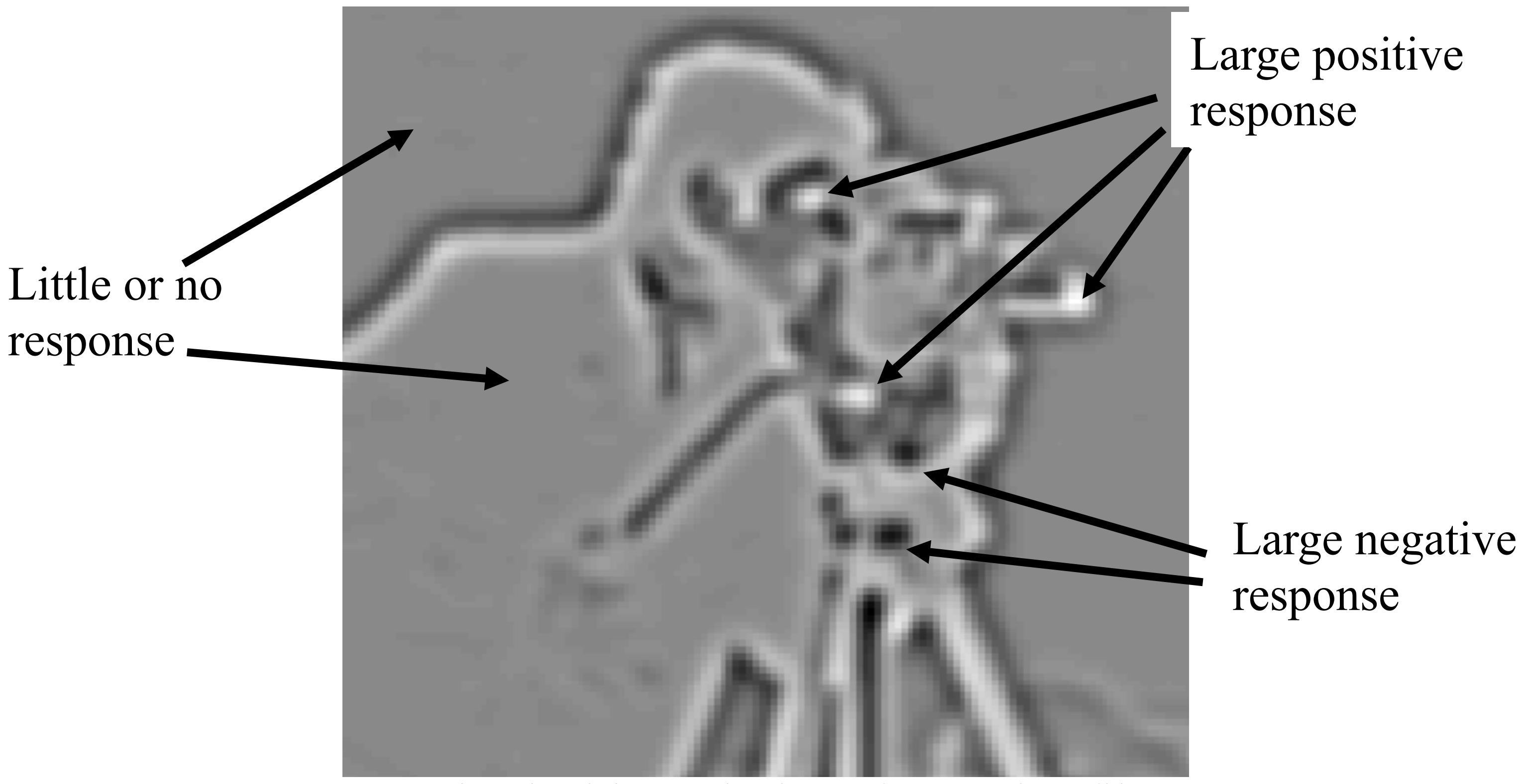
Simple Patch Matching

Background Reading:
T&V Section 7.2
Szeliski, Section 8.1

Summary: Filter Responses

Rough characterization of filter responses:

- little/no response (low magnitude values)
- large positive response (large positive values)
- large negative response (large negative values)



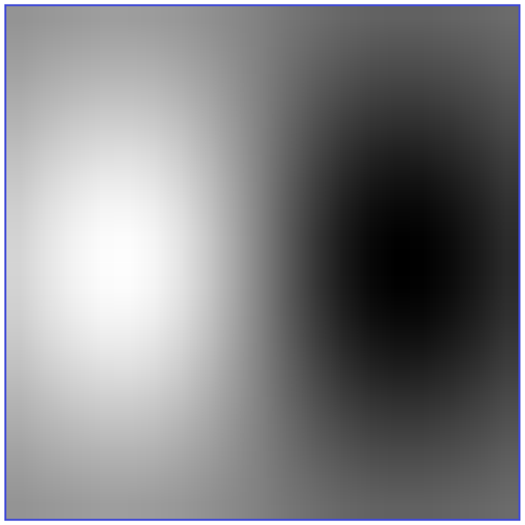
convolved with Laplacian of Gaussian filter

Summary: Filter Responses

Rough characterization of filter responses:

- little/no response (low magnitude values)
- large positive response (large positive values)
- large negative response (large negative values)

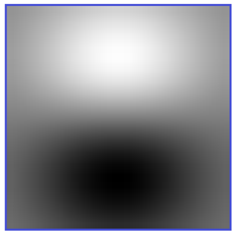
Deriv of Gaussian



Little/no response:
constant / slowly varying patches

Large positive response:
vertical edge

Large negative response:
vertical edge (opposite sign of contrast)



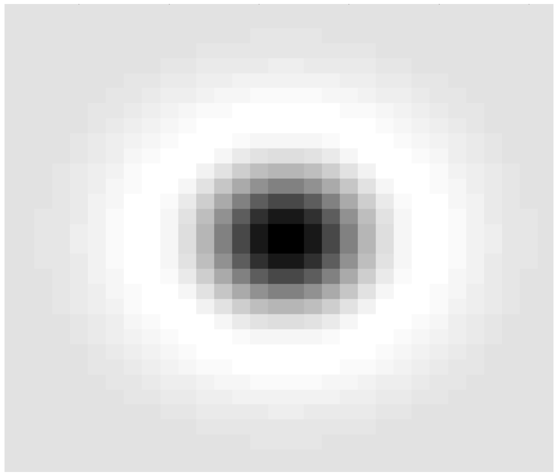
For horizontal edges

Summary: Filter Responses

Rough characterization of filter responses:

- little/no response (low magnitude values)
- large positive response (large positive values)
- large negative response (large negative values)

LoG


formed by second derivs of Gaussian

Little/no response:
constant / slowly varying patches
zero crossings (edges)

Large positive response:
dark blob on light background

Large negative response:
light blob on dark background

Derivative Filters

- Note that our first and second derivative filters have both positive and negative coefficient values in them.

e.g.

-1	1

1	-2	1

0	1	0
1	-4	1
0	1	0

- In fact, looking more closely, the positive and negative values perfectly balance each other -- the sum of filter coefficient values sums to zero.
- This is a not a coincidence, it is a consequence of the fact that derivatives of a constant image patch must = 0.

Convolution/Correlation with Constant Image Patch

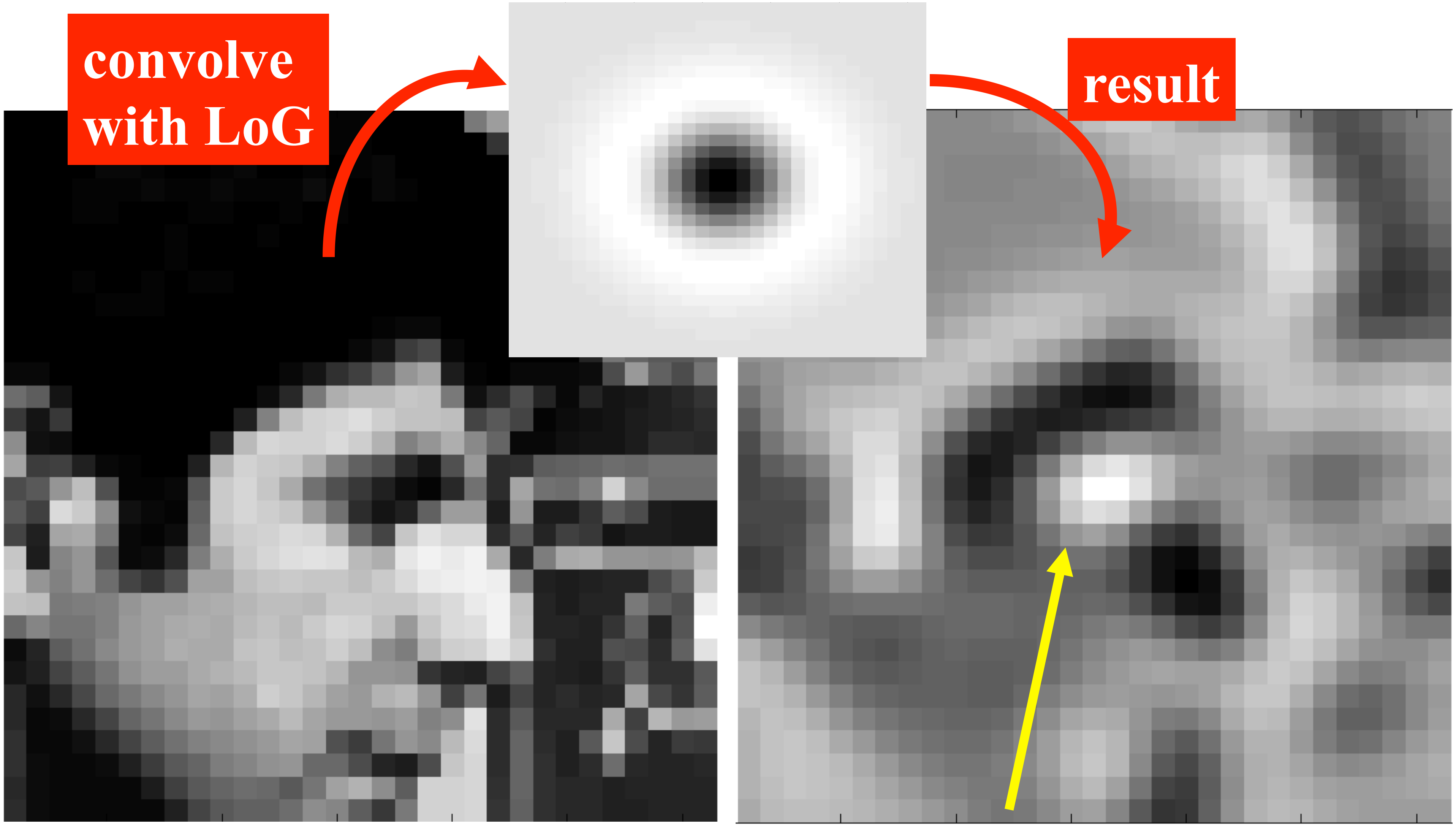
Consider correlation of general 3x3 filter with an image of constant grey value:

$$\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline\end{array} \otimes \begin{array}{|c|c|c|}\hline v & v & v \\ \hline v & v & v \\ \hline v & v & v \\ \hline\end{array}$$

Result: v*(a+b+c+d+e+f+g+h+i)

For this result to be zero, regardless of v, implies that the sum of filter coefficients must be zero.

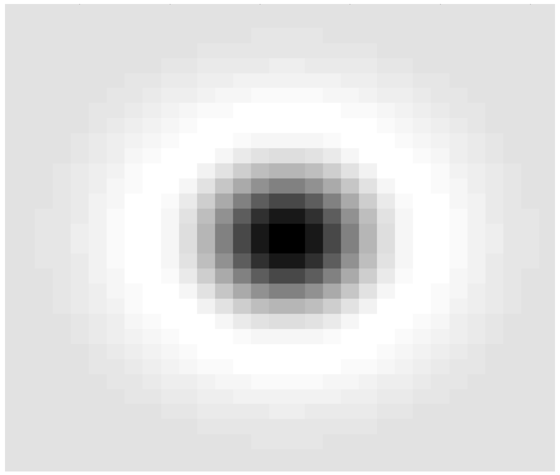
Observe and Generalize



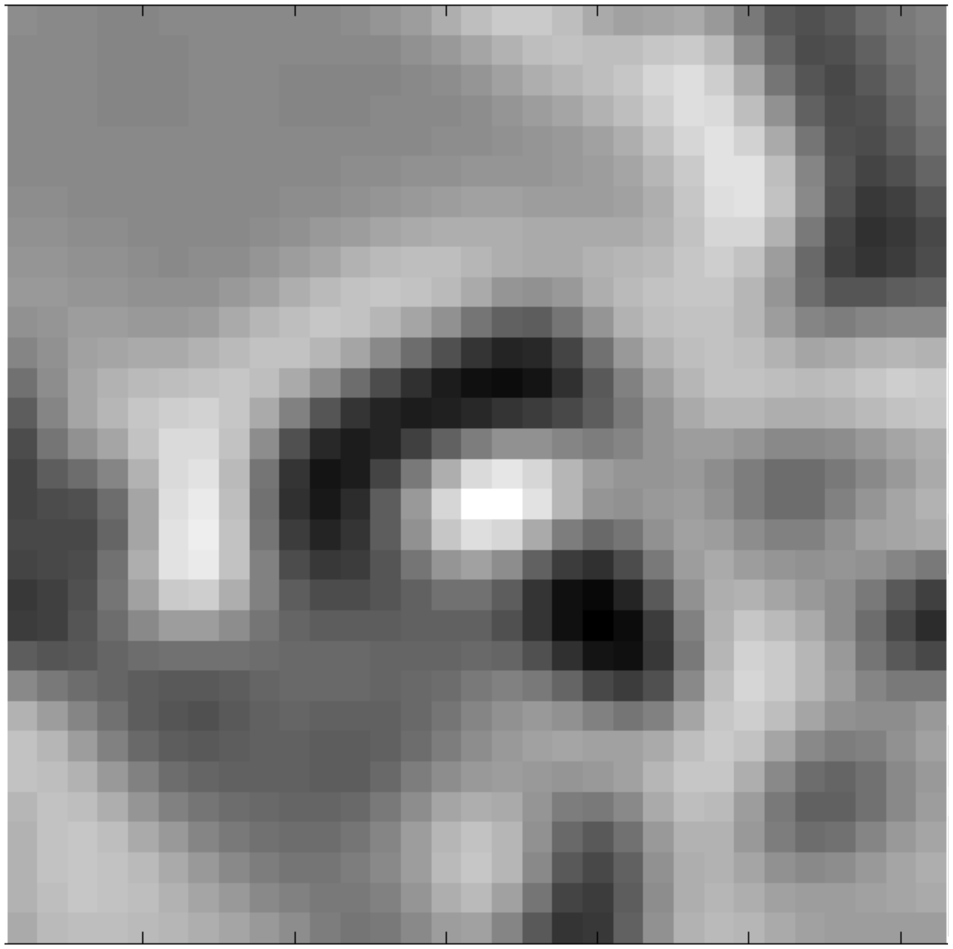
Large positive response

Observe and Generalize

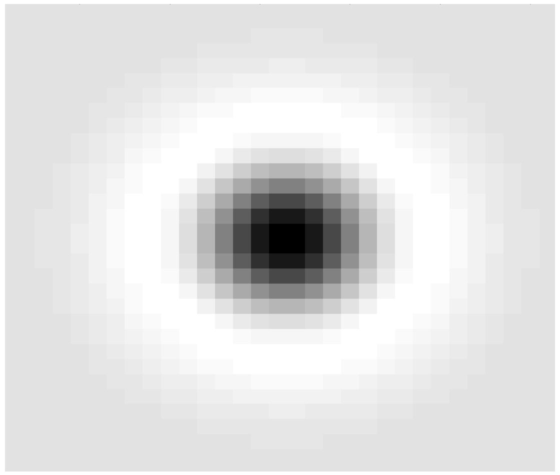
LoG looks a bit like an eye.



Observe and Generalize



LoG looks a bit like an eye.

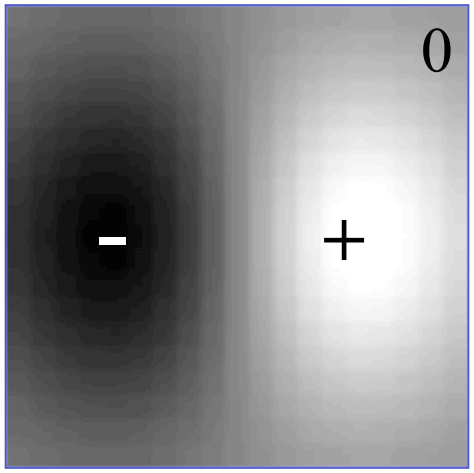


and it responds maximally in the eye region!

Observe and Generalize

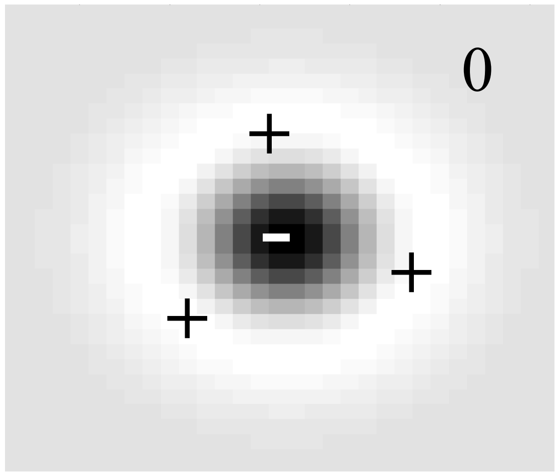
Key idea: Cross correlation with a filter can be viewed as comparing a little "picture" of what you want to find against all local regions in the image.

Deriv of Gaussian filter



Large positive response:
vertical edge; lighter on right
Large negative response:
vertical edge; lighter on left

LoG filter



Large positive response:
dark blob on light background
Large negative response:
light blob on dark background

Observe and Generalize

Key idea: Cross correlation with a filter can be viewed as comparing a little "picture" of what you want to find against all local regions in the image.

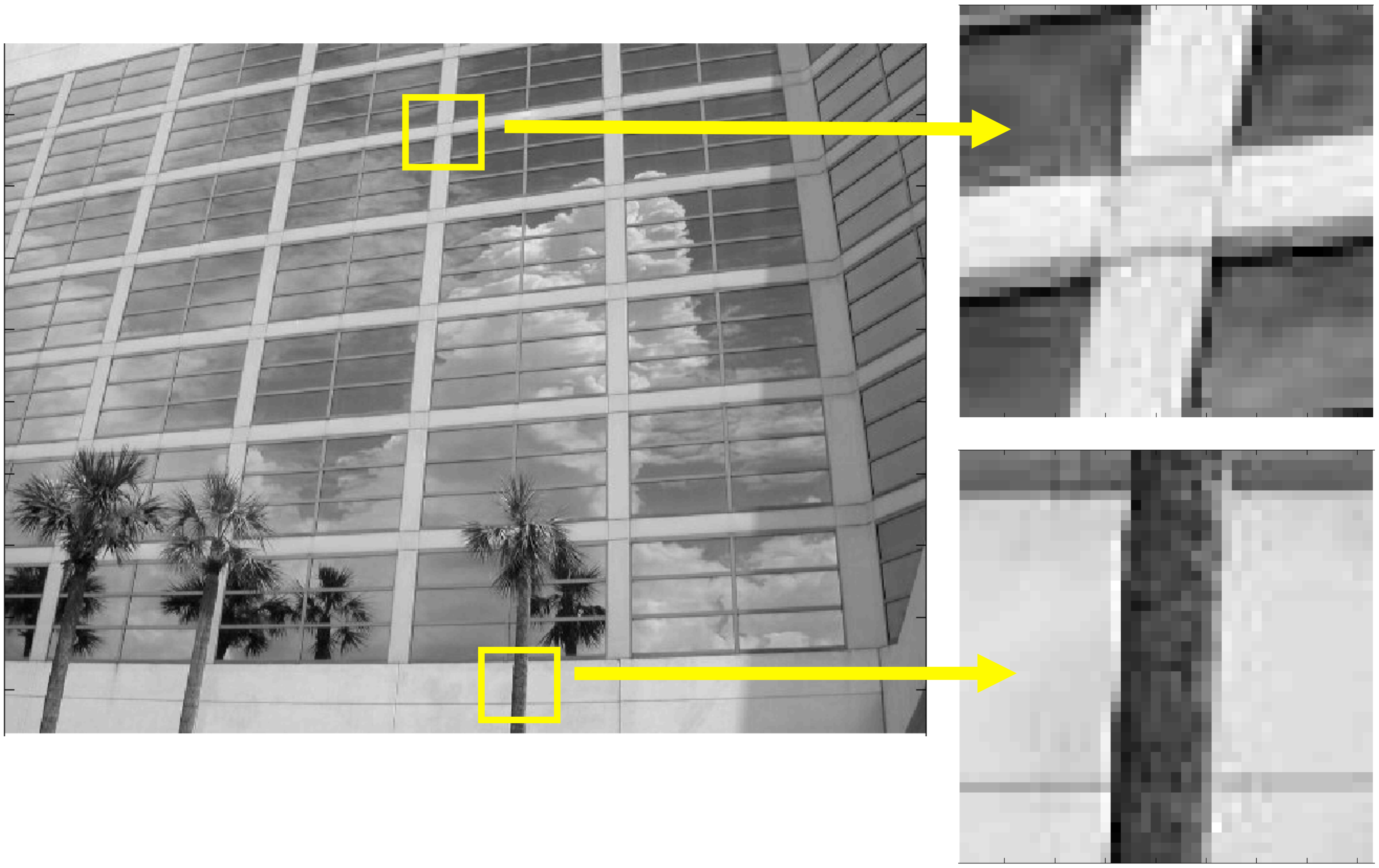
For this reason, it is sometimes called "matched filtering"

In fact, you can prove that the best linear operator for finding an image patch is essentially the patch itself

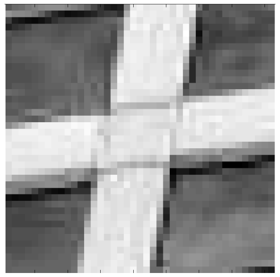
(using variational calculus, outside scope of our course).

Template Matching

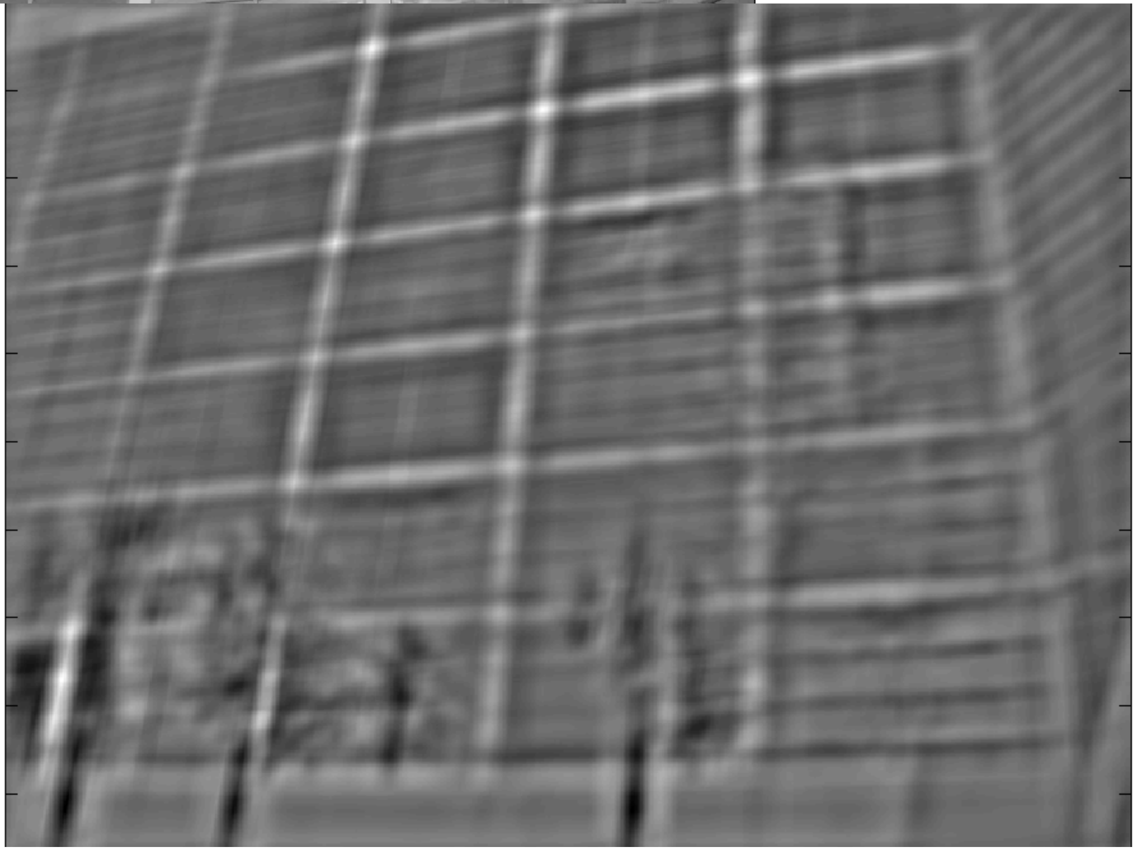
What if we cut little pictures out from the image, then cross-correlated them with the same or other images?



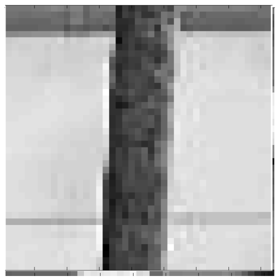
Template Matching



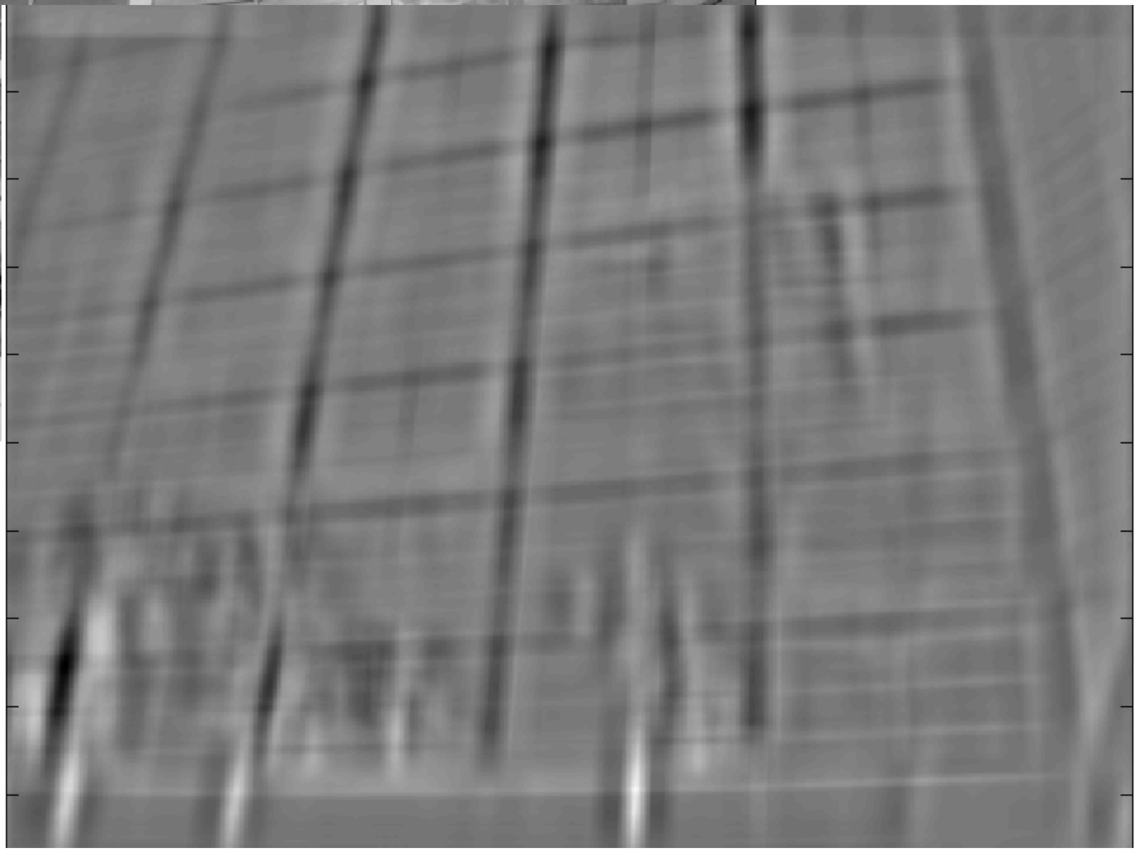
*



Template Matching



*



Confession

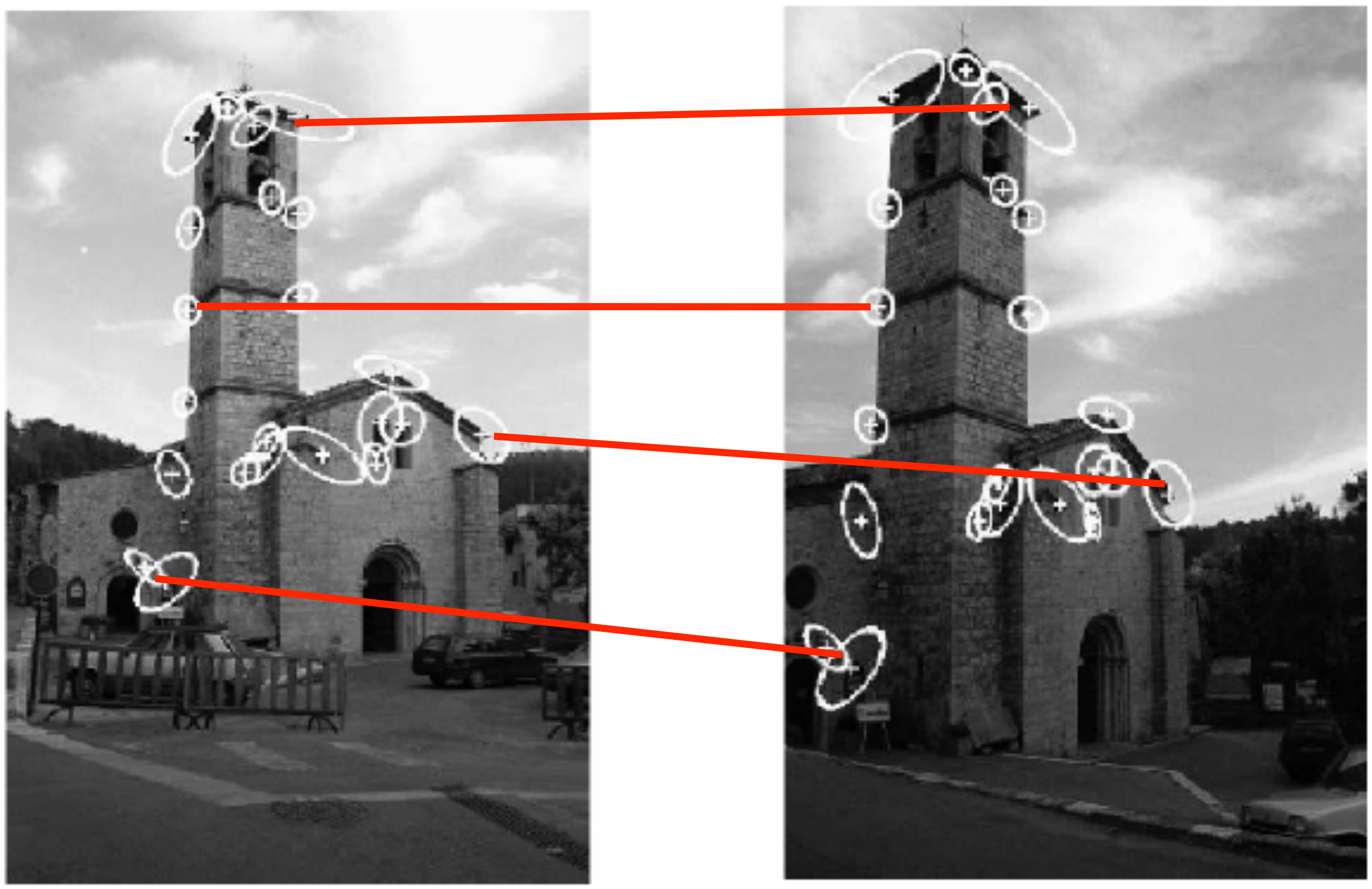
I've cheated a little bit.

I subtracted the mean grey value from the template before doing cross correlation.

Why? (we will discuss later, but think about it first)

Correspondence Problem

Vision tasks such as stereo and motion estimation require finding corresponding features across two or more views.



The Correspondence Problem

- Basic assumptions:
  - Most scene points are visible in both images
  - Corresponding image regions are similar
- These assumptions hold if:
  - The distance of points from the cameras is much larger than the distance between cameras

The Correspondence Problem

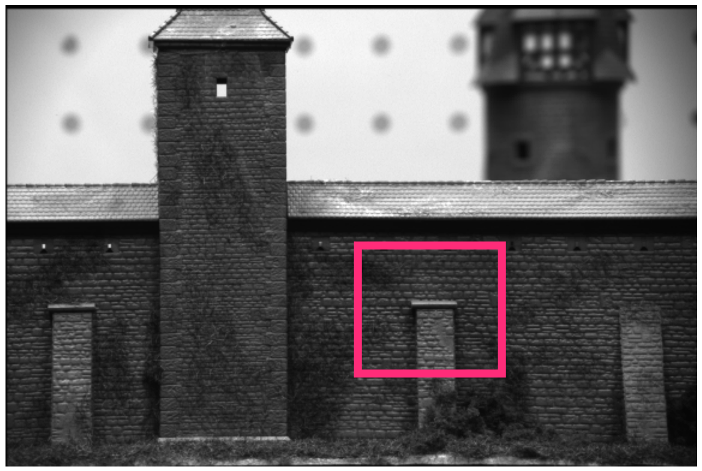
- Is a “search” problem:
  - Given an element in the left image, search for the corresponding element in the right image.
  - We will typically need geometric constraints to reduce the size of the search space
- We must choose:
  - Elements to match
  - A similarity measure to compare elements

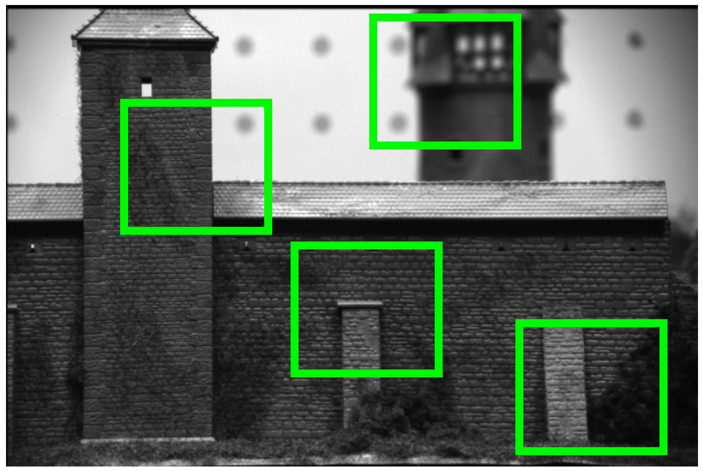
Correspondence Problems

- Two classes of algorithms:
  - Dense : compute a set of correspondences at (nearly) every pixel
  - Sparse : compute correspondences between a discrete set of key points / features

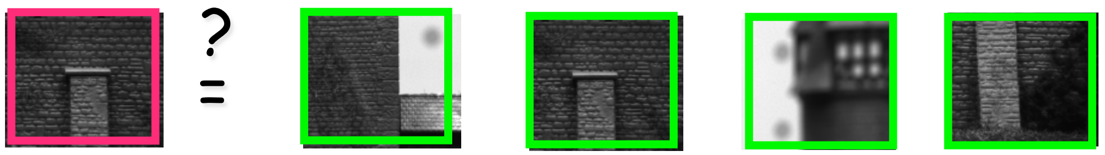
Correlation-based Algorithms

Elements to be matched are image patches of fixed size



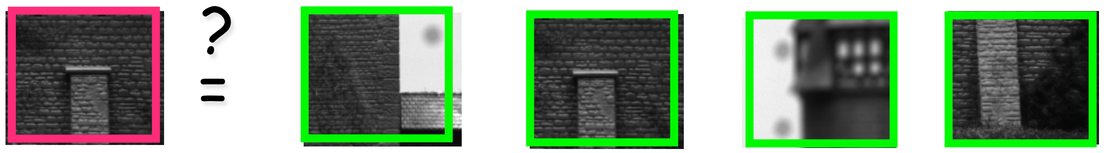


Task: what is the corresponding patch in a second image?



Correlation-based Algorithms

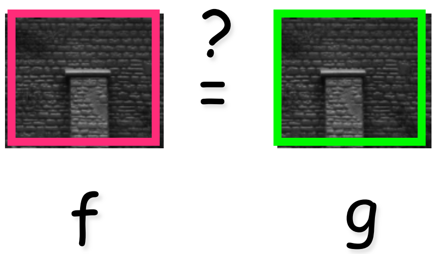
Task: what is the corresponding patch in a second image?



1) **Need an appearance similarity function.**

2) **Need a search strategy to find location with highest similarity. Simplest (but least efficient) approach is exhaustive search.**

Comparing Windows:



Some possible measures:

$$\max_{[i,j]\in R} |f(i,j) - g(i,j)|$$

$$\sum_{[i,j]\in R} |f(i,j) - g(i,j)|$$

$$\left.\begin{aligned} SSD &= \sum_{[i,j]\in R} (f(i,j) - g(i,j))^2 \\ C_{fg} &= \sum_{[i,j]\in R} f(i,j)g(i,j) \end{aligned}\right\}$$

Most popular

Correlation C_{fg}

$$C_{fg} = \sum_{[i,j]\in R} f(i,j)g(i,j)$$

If we are doing exhaustive search over all image patches in the second image, this becomes cross-correlation of a template with an image. We have seen this before – imfilter(im,template,'corr' ,'same').

Example

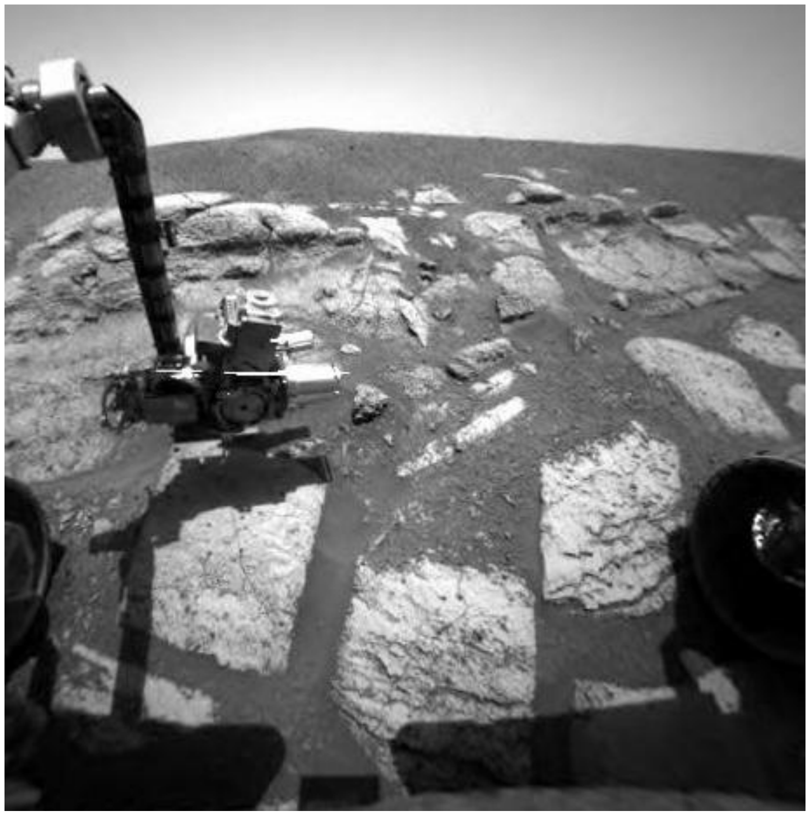


Image 1

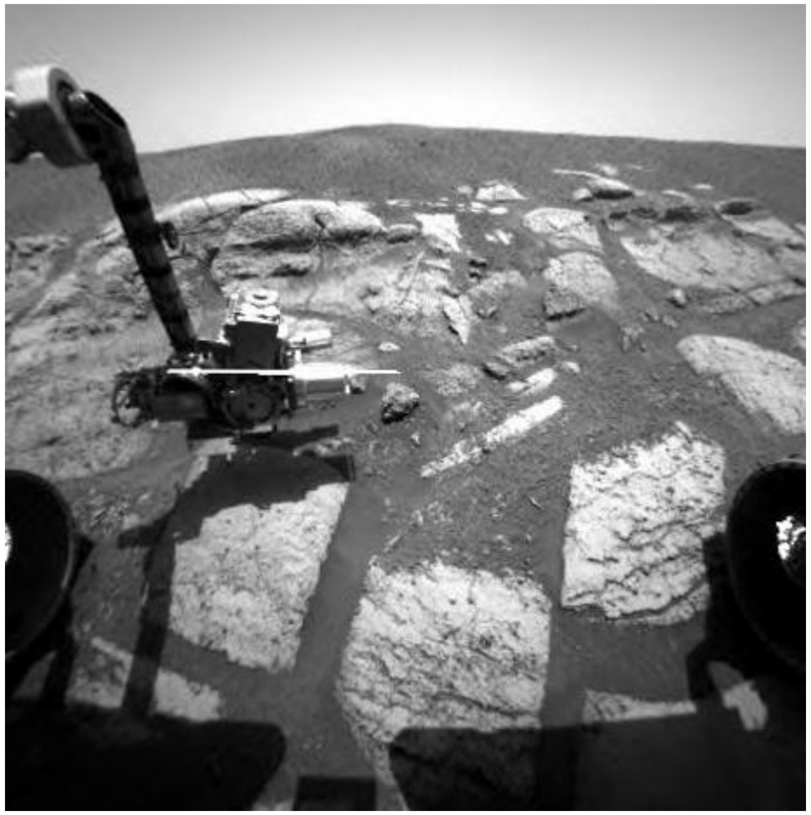


Image 2

Note: this is a stereo pair from the NASA mars rover. The rover is exploring the “El Capitan” formation.

Example

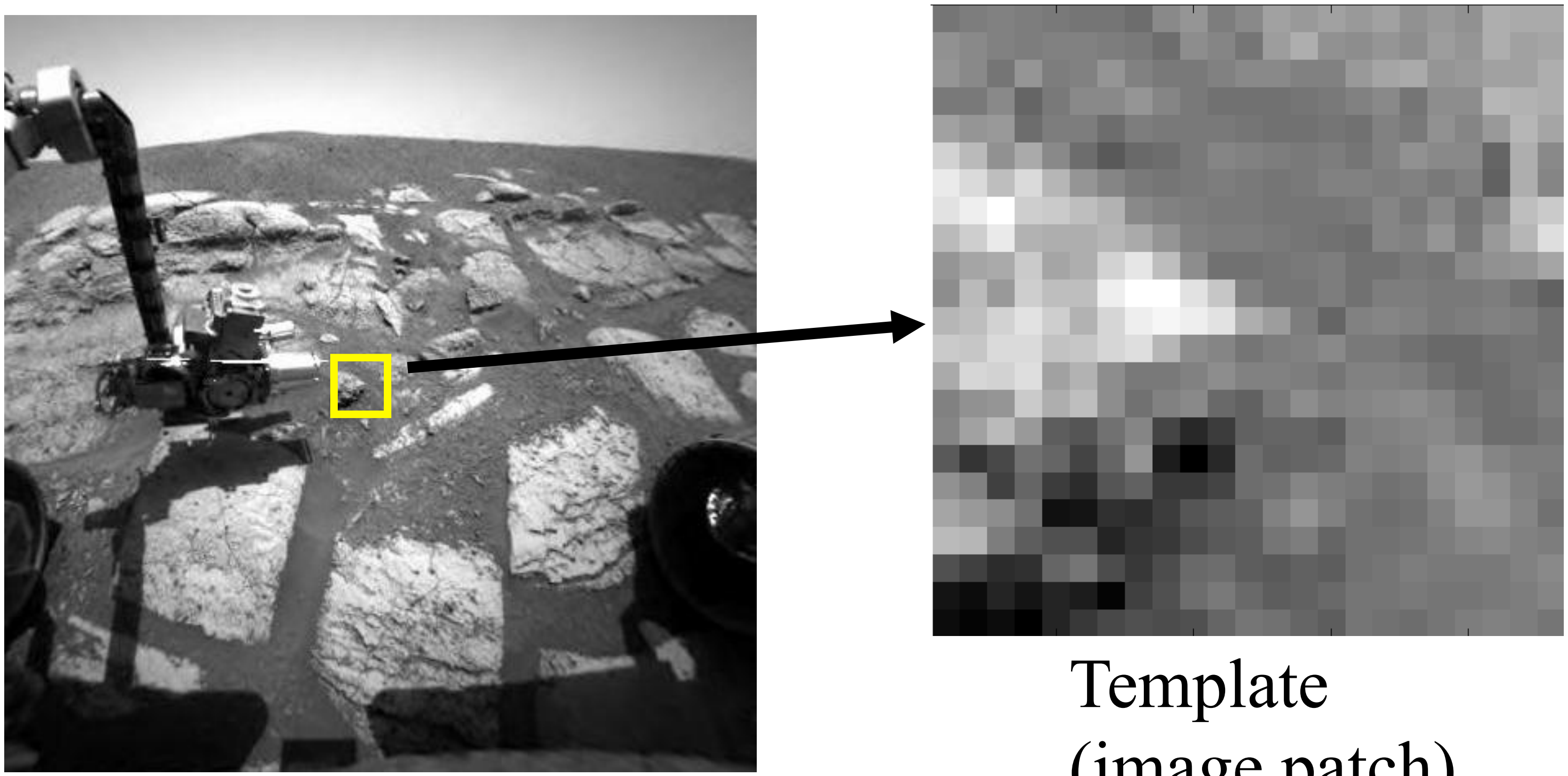
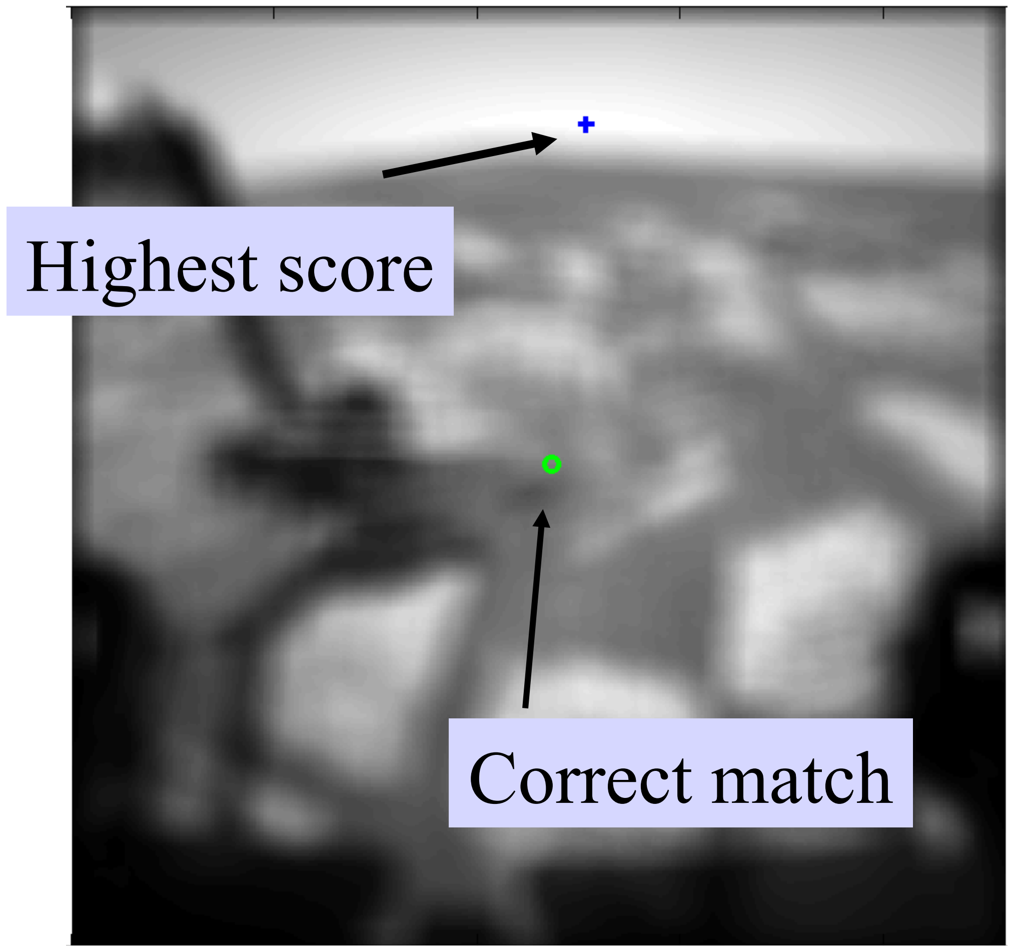


Image 1

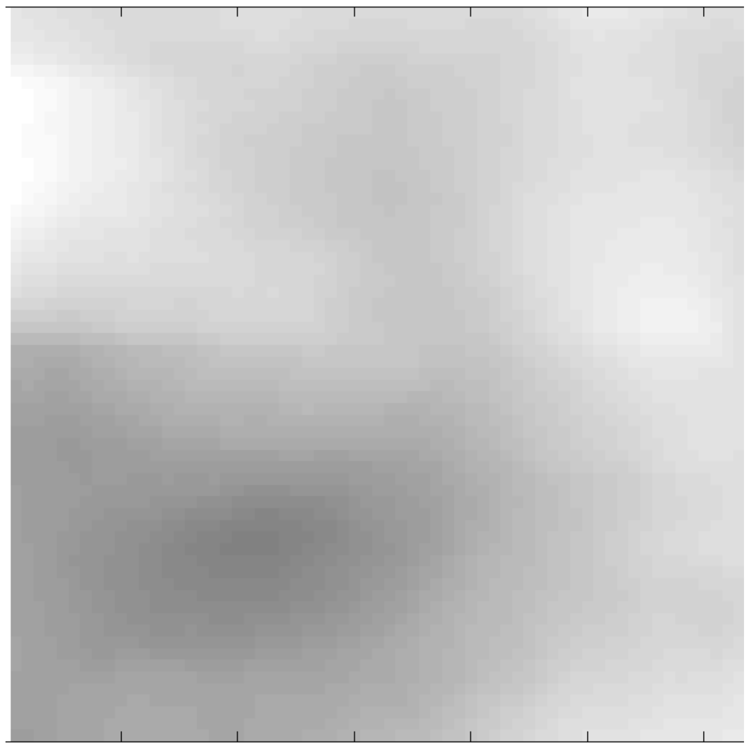
Template
(image patch)

Example: Raw Cross-correlation

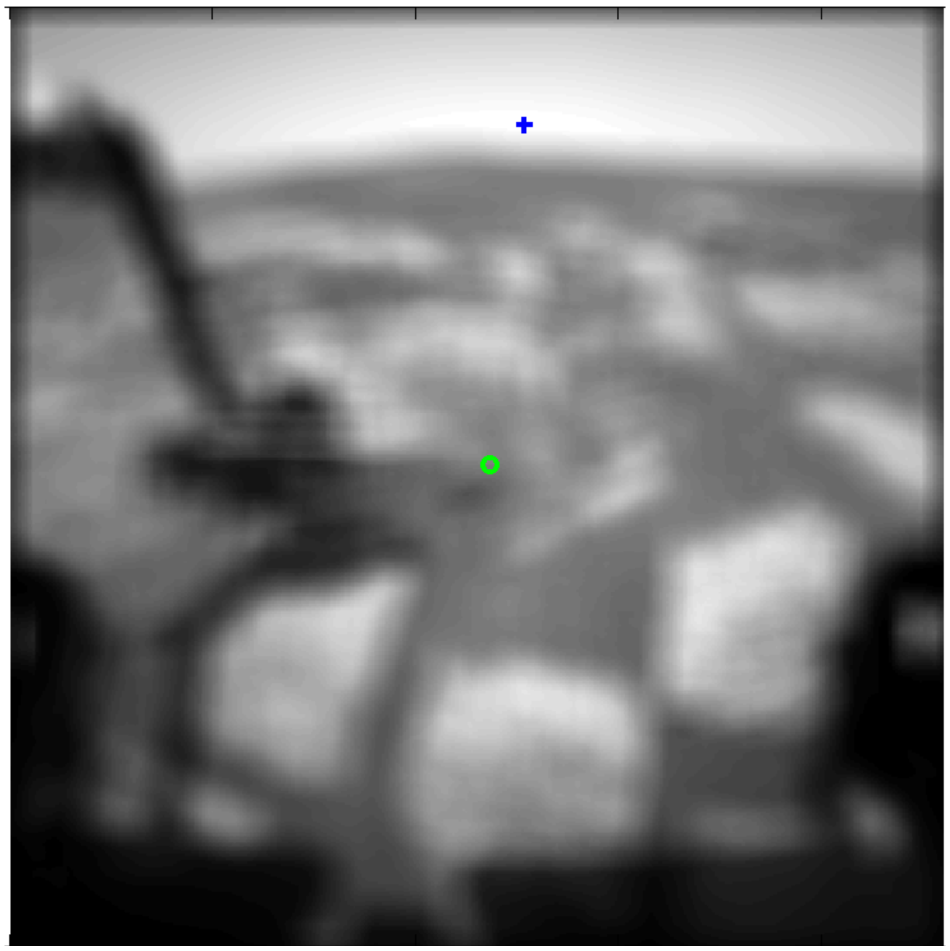
score = imfilter(image2,tmpl,'corr')



Score around correct match



Example: Cross-correlation



Note that score image looks a lot like a blurry version of image 2.

This clues us in to the problem with straight correlation with an image template.

Problem with Correlation of Raw Image Templates

Consider once again the correlation of template with an image of constant grey value:

$$\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline \end{array} \otimes \begin{array}{|c|c|c|}\hline v & v & v \\ \hline v & v & v \\ \hline v & v & v \\ \hline \end{array}$$

Result: v*(a+b+c+d+e+f+g+h+i)

Problem with Correlation of Raw Image Templates

Now consider correlation with a constant image that is twice as bright.

$$\begin{array}{|c|c|c|} \hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline \end{array} \otimes \begin{array}{|c|c|c|} \hline 2v & 2v & 2v \\ \hline 2v & 2v & 2v \\ \hline 2v & 2v & 2v \\ \hline \end{array}$$

$$\text{Result: } 2*v*(a+b+c+d+e+f+g+h+i)$$
$$> v*(a+b+c+d+e+f+g+h+i)$$

If a,b,...,i are all positive (e.g. pixel values), we get a larger score regardless of what patch they represent!

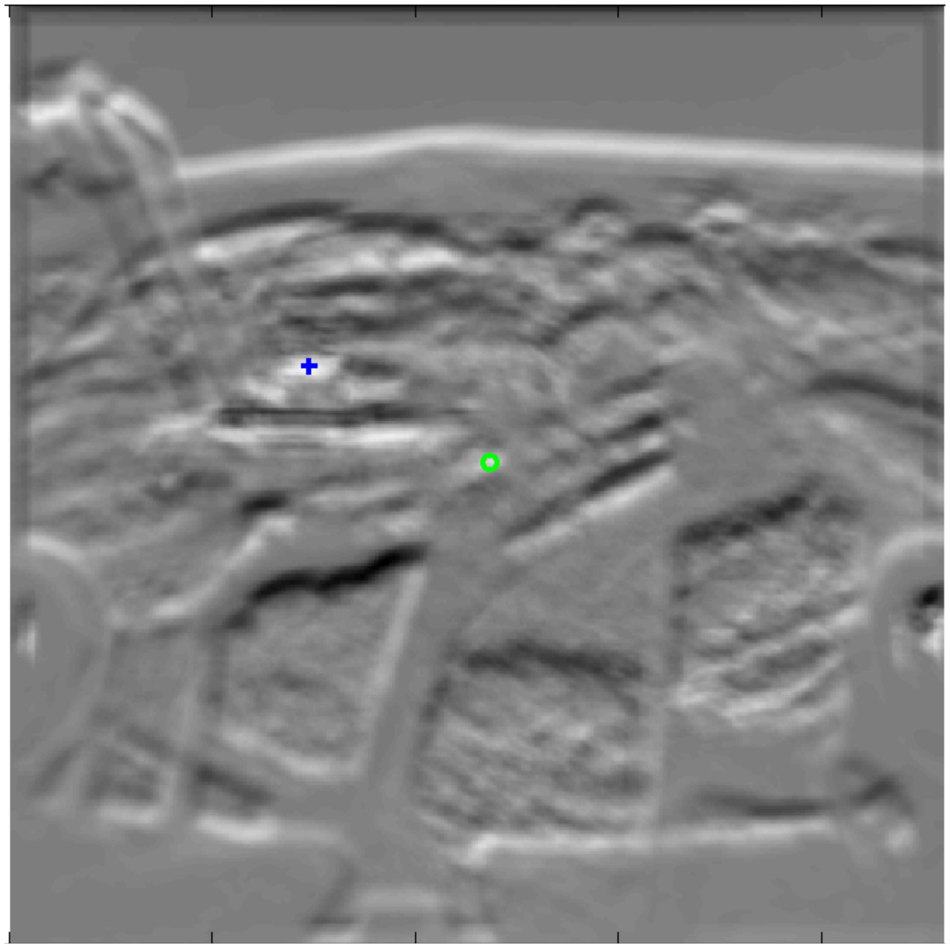
Solution

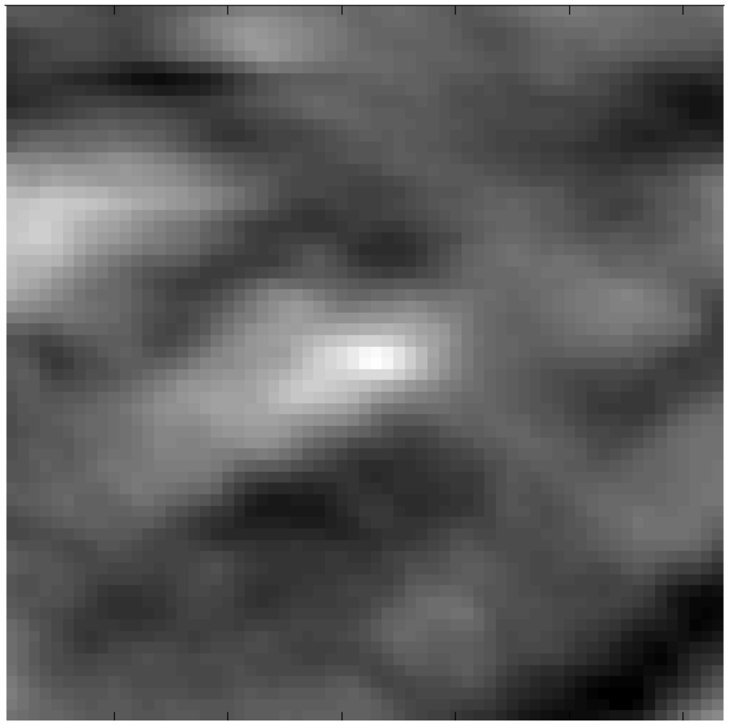
Subtract off the mean value of the template.

In this way, the correlation score is higher only when darker parts of the template overlap darker parts of the image, and brighter parts of the template overlap brighter parts of the image.

Note: this makes some coefficients in the template negative, and some positive, just like in our earlier examples of derivative of Gaussian and LoG filters.

Correlation, zero-mean template





Better! But highest score is still not the correct match.

However, note that the highest score in a local neighborhood of the correct match is the best match.

"SSD" or "block matching" (Sum of Squared Differences)

$$\sum_{[i,j]\in R} (f(i,j) - g(i,j))^2$$

1) Popular matching score
2) Easily differentiable (good for mathematical analysis)
3) Some claim it works better than cross-correlation
4) The basis for nearest-neighbor matching

Relation between SSD and Correlation

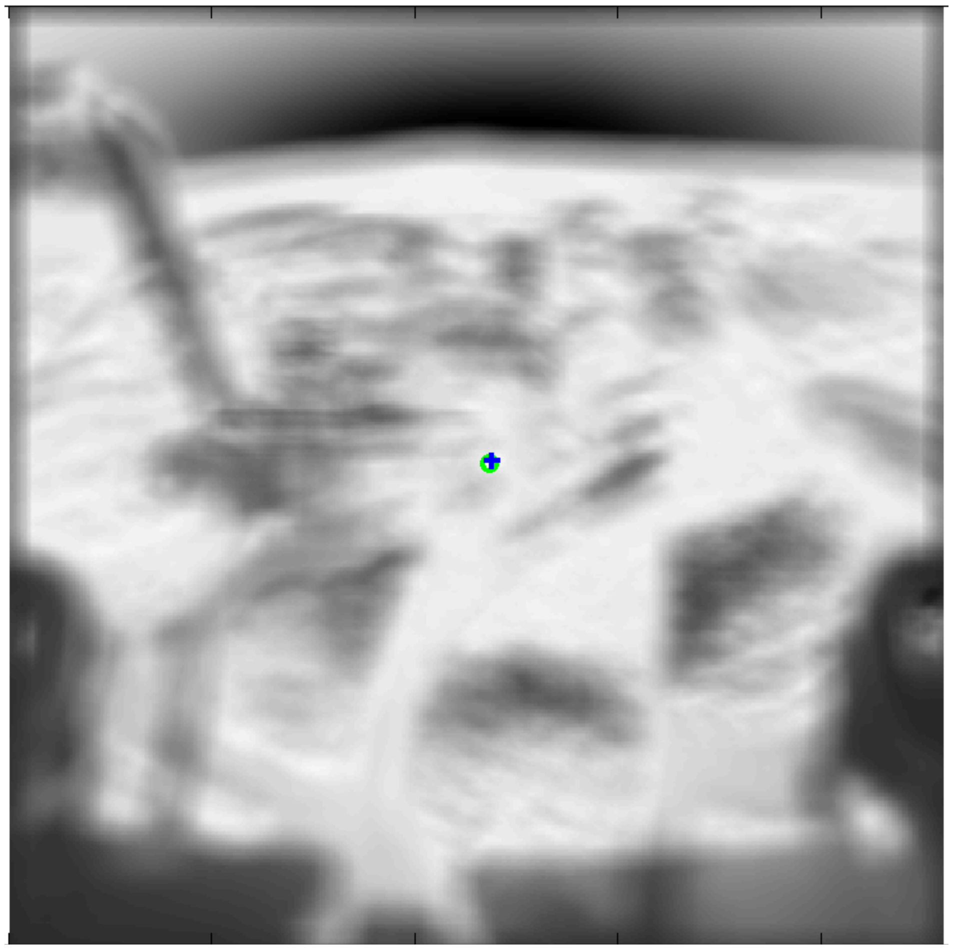
$$SSD = \sum_{[i,j]\in R} (f-g)^2$$

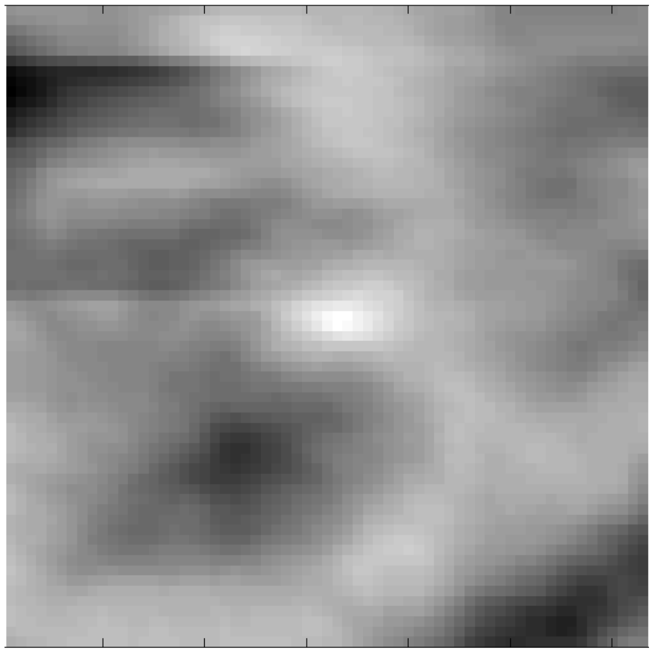
$$= \sum_{[i,j]\in R} f^2 + \sum_{[i,j]\in R} g^2 - 2\boxed{\sum_{[i,j]\in R} fg}$$

$$C_{fg} = \sum_{[i,j]\in R} f(i,j)g(i,j)$$

Correlation!

SSD





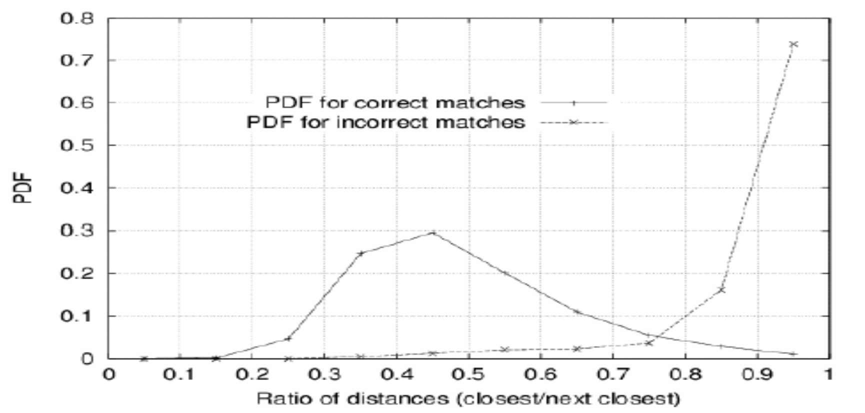
Best match (highest score) in image coincides with correct match in this case! But lots of “close” seconds.

SSD / Nearest-Neighbor Matching

Getting better matches from SSD

1) Choose closest match that does not exceed a threshold

2) Compare closest match distance with second closest match distance



3) Bidirectional matching – when matching two sets of features A1,A2,... and B1,B2,... declare a match between Ai and Bk if and only if Bk is closest match for Ai and Ai is closest match for Bk (mutually compatible)

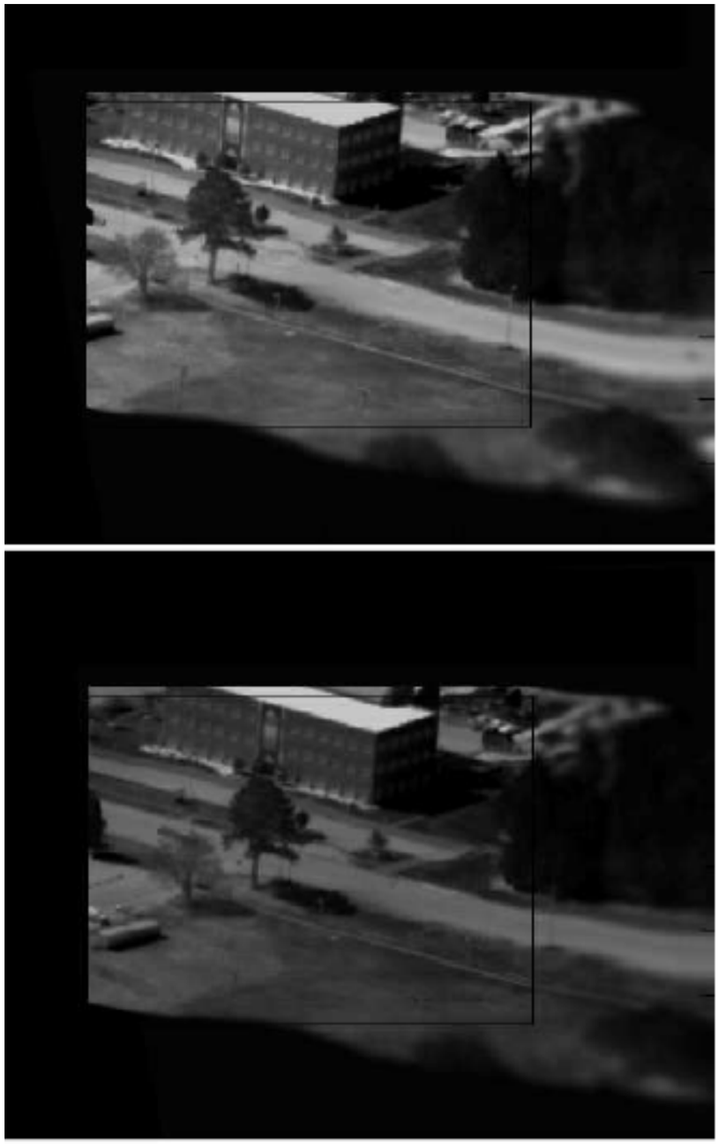
Normalized Correlation

Motivation: Dealing with Intensity Changes

•the camera taking the second image might have different intensity response characteristics than the camera taking the first image

•Illumination in the scene could change

•The camera might have auto-gain control set, so that it's response changes as it moves through the scene.



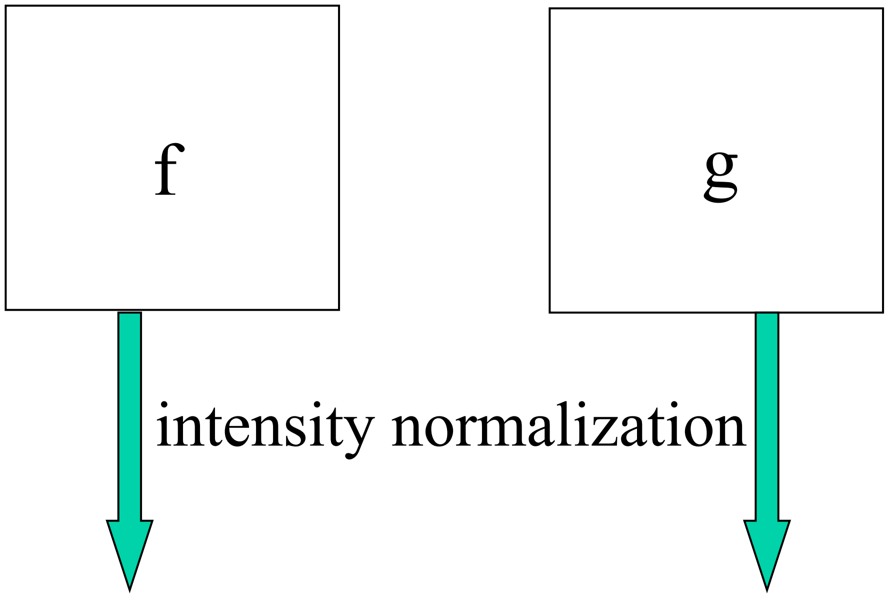
Intensity Normalization

•When a scene is imaged by different sensors, or under different illumination intensities, both SSD and C_{fg} can be misleading for patches representing the same area in the scene!

•A solution is to NORMALIZE the pixels in both patches before comparing them by subtracting the mean of the patch intensities and dividing by a constant proportional to the standard deviation.

$$\hat{f} = \frac{f - \bar{f}}{\sqrt{\sum (f - \bar{f})^2}} \qquad \hat{g} = \frac{g - \bar{g}}{\sqrt{\sum (g - \bar{g})^2}}$$

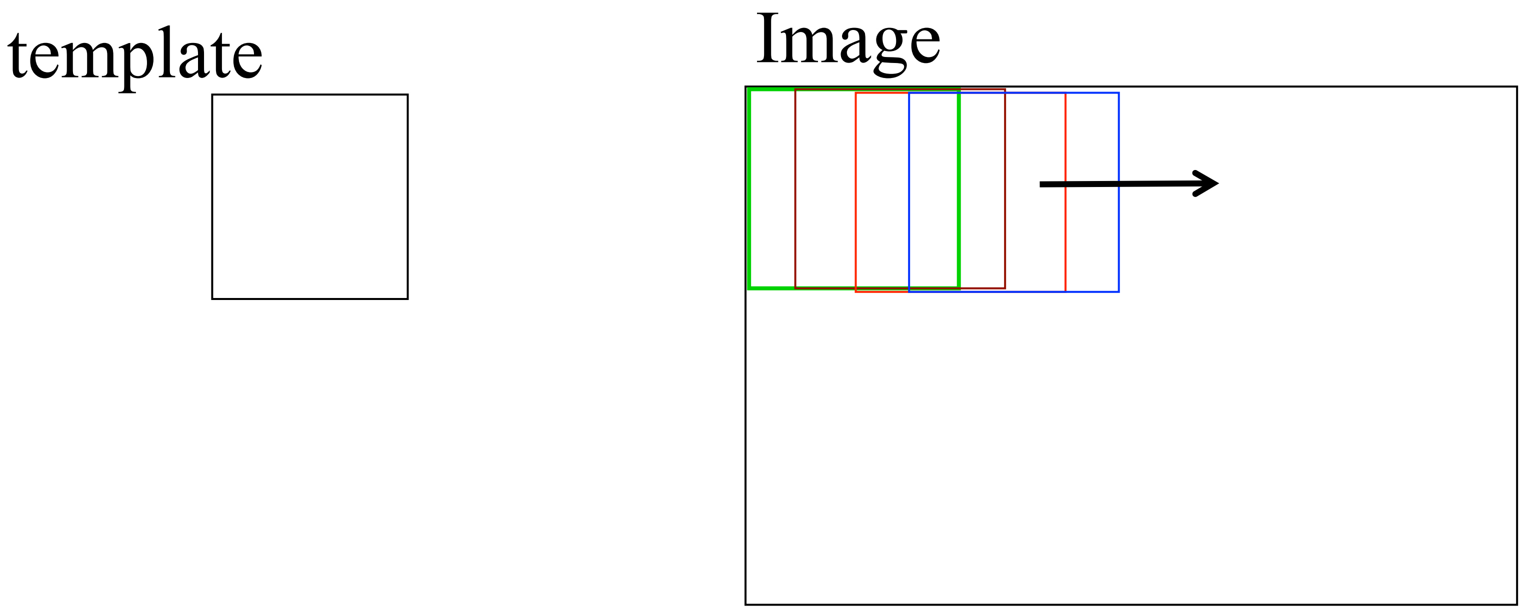
Normalized Correlation



$$\hat{f} = \frac{f - \bar{f}}{\sqrt{\sum (f - \bar{f})^2}} \qquad \hat{g} = \frac{g - \bar{g}}{\sqrt{\sum (g - \bar{g})^2}}$$

$$\boxed{\mathrm{NCC(f,g)} = C_{fg}(\hat{f}, \hat{g}) = \sum_{[i,j] \in R} \hat{f}(i,j)\hat{g}(i,j)}$$

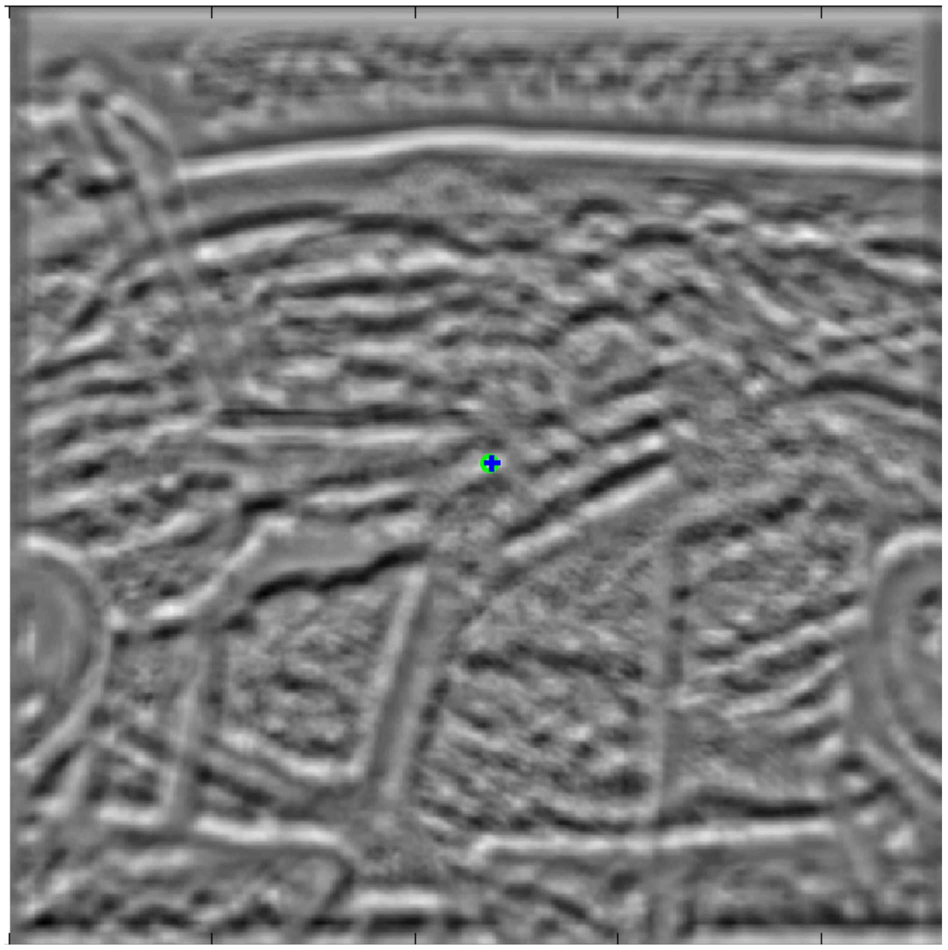
Normalized Cross Correlation

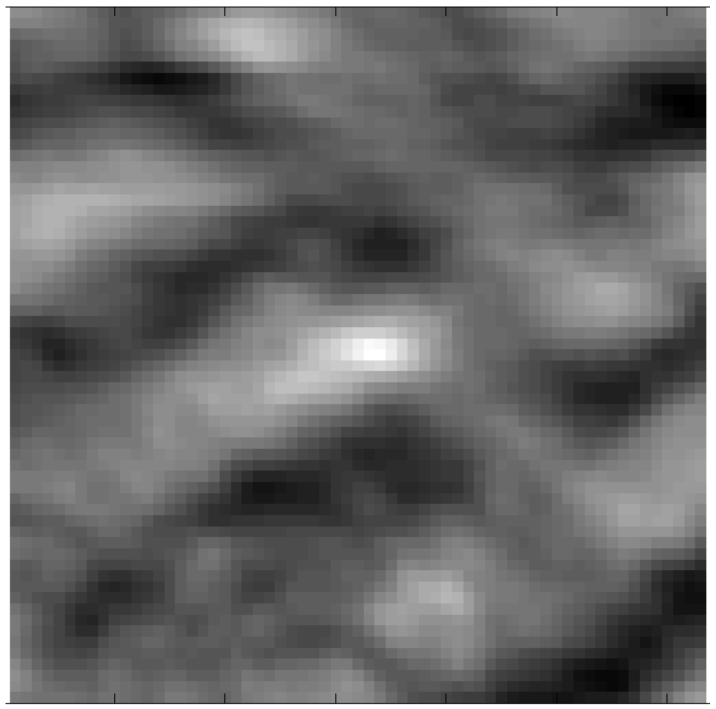


Like cross correlation of a template with all patches in an image, but using normalized correlation to compute the score between each pair of patches

In Matlab: normxcorr2(template, image)

Normalized Cross Correlation





Highest score also coincides with correct match.
Also, looks like less chances of getting a wrong match.

Normalized Cross Correlation

Important point about NCC:
Score values range from 1 (perfect match)
to -1 (completely anti-correlated)

Intuition: treating the normalized patches as vectors, we see they are unit vectors. Therefore, correlation becomes dot product of unit vectors, and thus must range between -1 and 1.

Consequence: determining whether a match is "good" or not becomes easier, because the range of score values is tightly bounded and well-understood.