COSC160: Data Structures

Jeremy Bolton, PhD
Assistant Teaching Professor

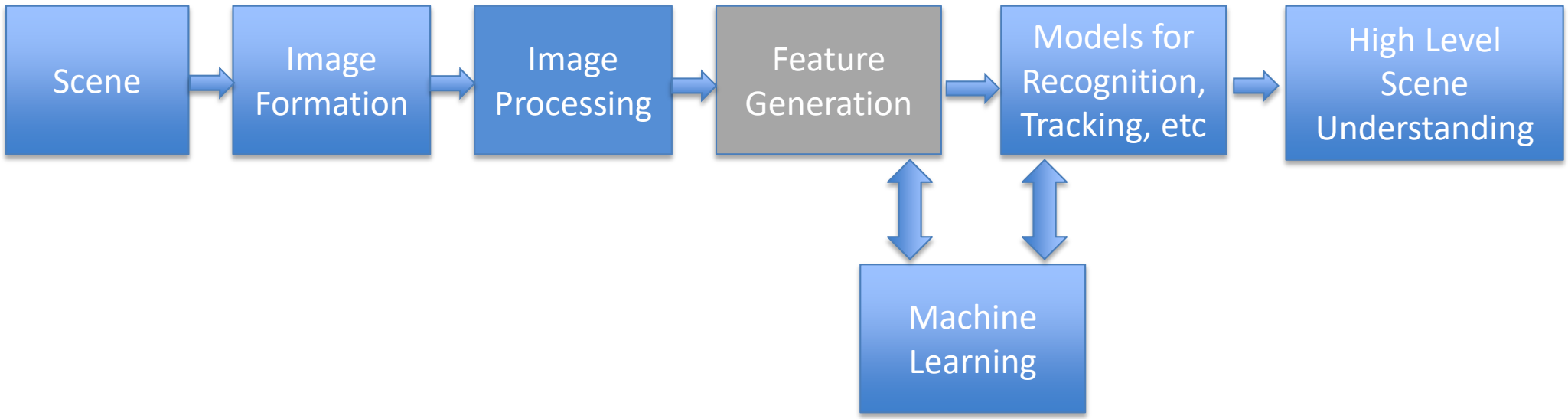
GEORGETOWN UNIVERSITY

Outline

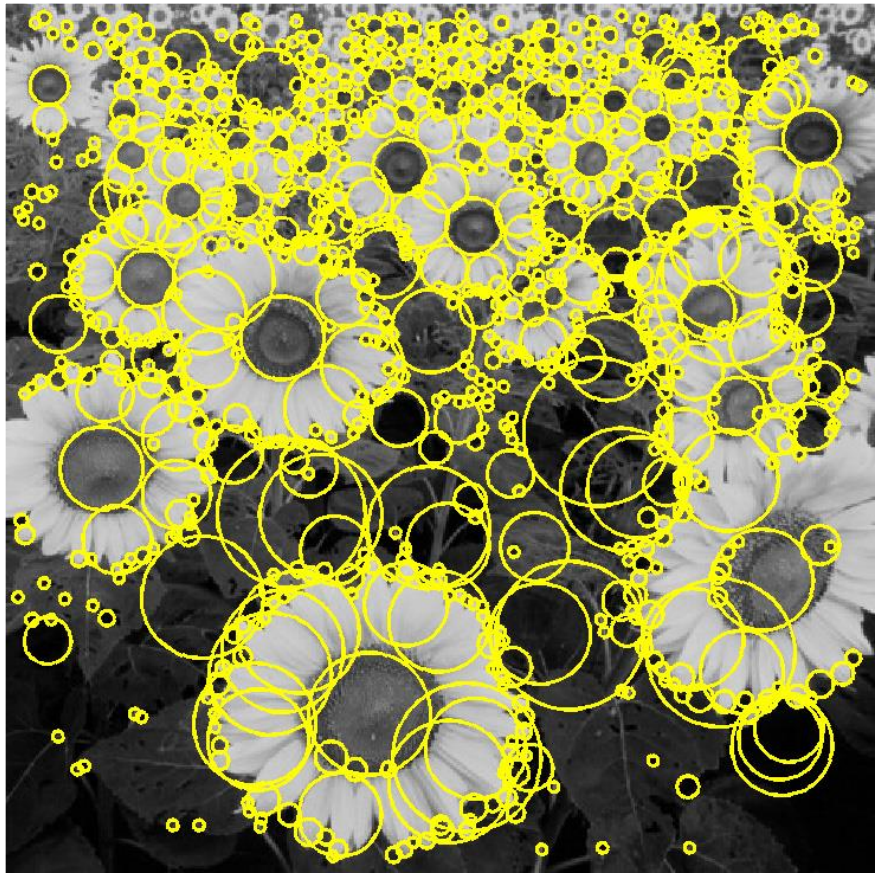
I. Why Features?
II. What makes a “good” feature?
    I. Invariance
III. Common Features
IV. Features for Classification

GEORGETOWN UNIVERSITY

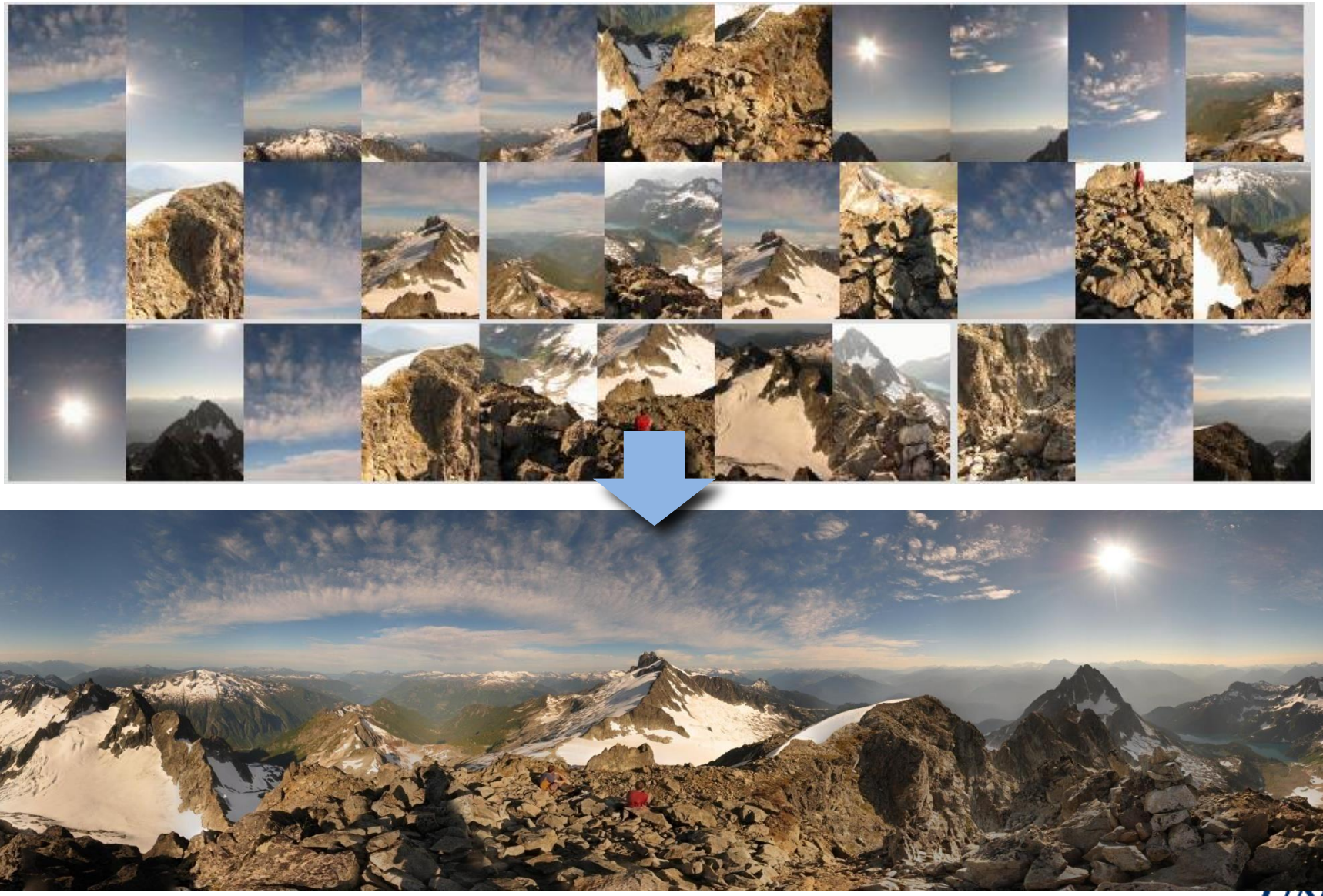
Big Picture: Computer Vision



Feature extraction: Corners and blobs



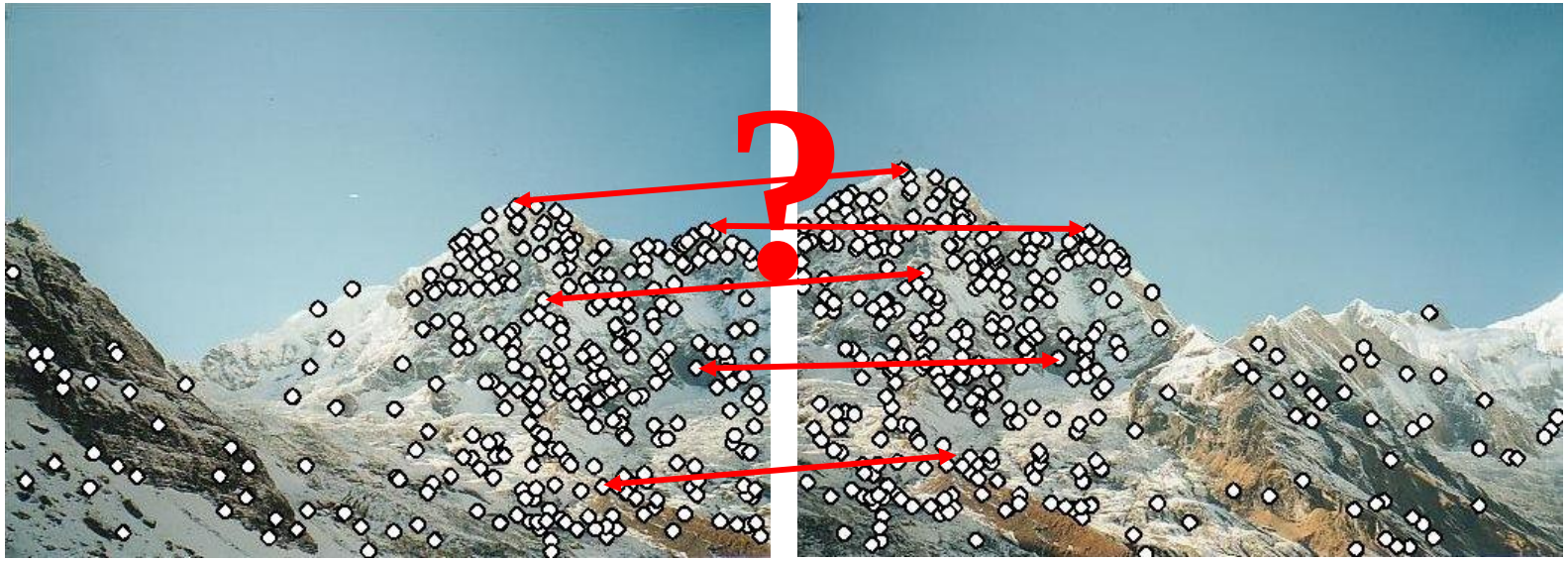
Motivation: Automatic panoramas (stitching)



Credit: Matt Brown

Why extract features?

We know how to detect good points
Next question: **How to match them?**



Lots of possibilities (this is a popular research area)

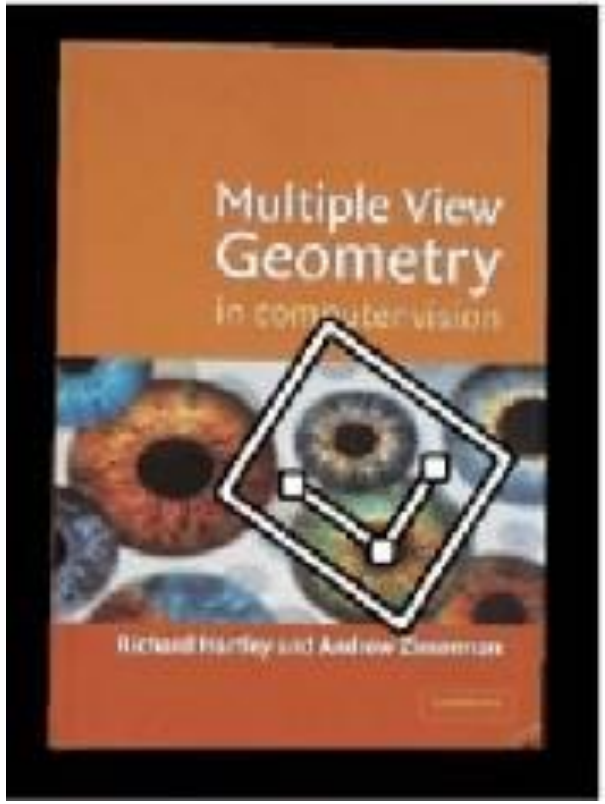
- Simple option: match square windows around the point (tailor-made, matched filter)
- State of the art approach: SIFT
  - David Lowe, UBC http://www.cs.ubc.ca/~lowe/keypoints/

Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?

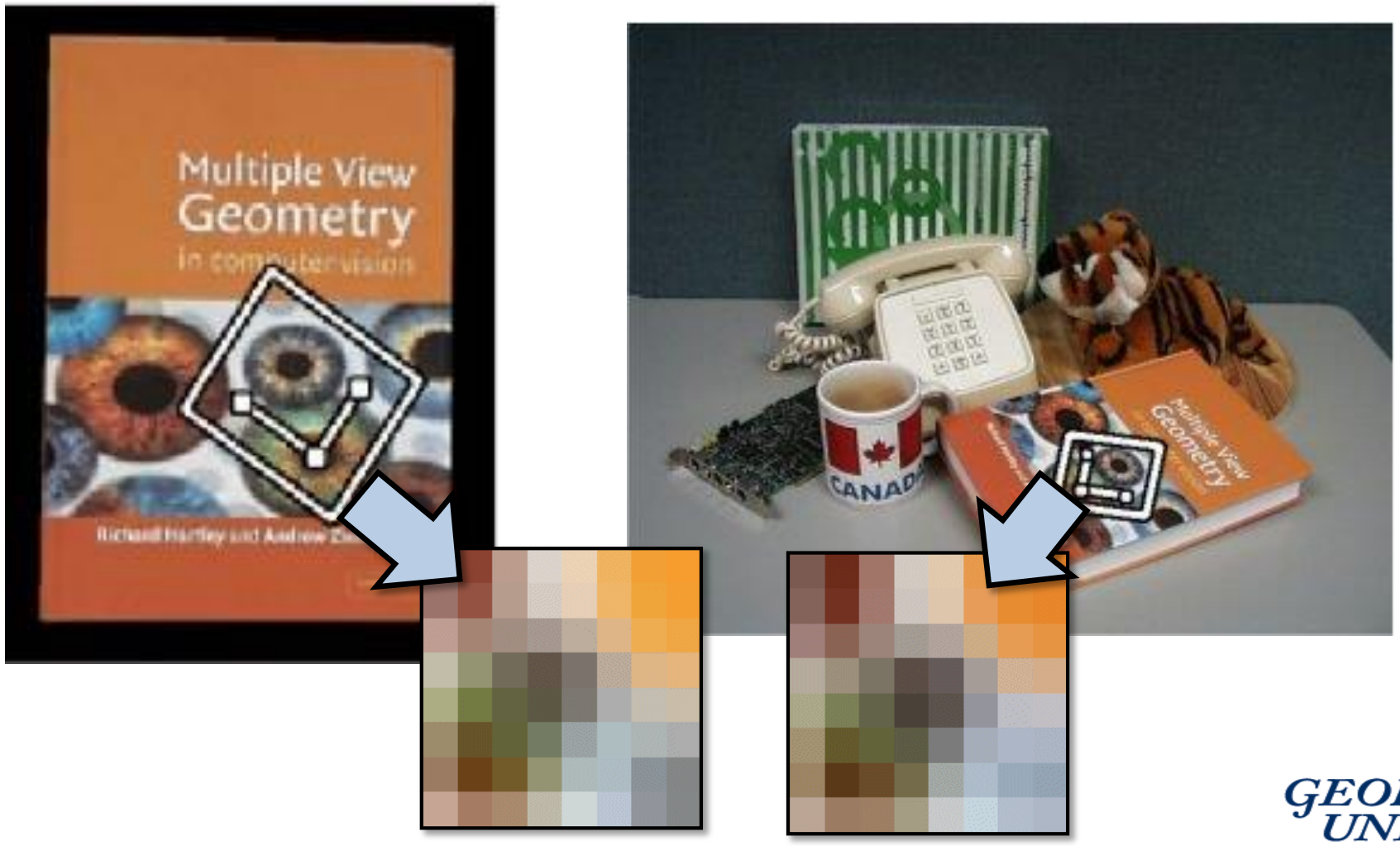
Step 1: extract features
Step 2: match features
Step 3: align images

Why Features? Feature matching for Recognition



GEORGETOWN UNIVERSITY

Feature Matching



GEORGETOWN UNIVERSITY

More motivation…

Feature points are used for:

- Image alignment (e.g., mosaics)
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
- … other

Local vs. Global Features: Advantages of local features

Locality

- features are local, so robust to occlusion and clutter

Quantity

- hundreds or thousands in a single image

Distinctiveness:

- can differentiate a large database of objects

Efficiency

- real-time performance achievable

Feature Generation

- Some common features
  - Edge kernels (Previously Discussed)
  - Texture Features (Previously Discussed)
  - Orientation
    - Histogram of Gradients (HoG)
    - Edge Histogram Descriptors (EHD)
    - Hough Transform (Note: Often Global Feature)
  - Invariant Features
    - SIFT
    - MOPS
    - Harris (Corner)
  - Tailor-made features
    - design your own kernel that works for a specific problem

What makes a good feature?

Depends on a few factors

- Purpose:
  - Classification (models used)
  - Feature (Image) Matching

- Data Characteristics (Do you want to capture changes in)
  - Scale
  - Intensity
  - Rotations
  - color

Invariance

Suppose we are comparing two images I_1 and I_2

- I_2 may be a transformed version of I_1
- What kinds of transformations are we likely to encounter in practice?

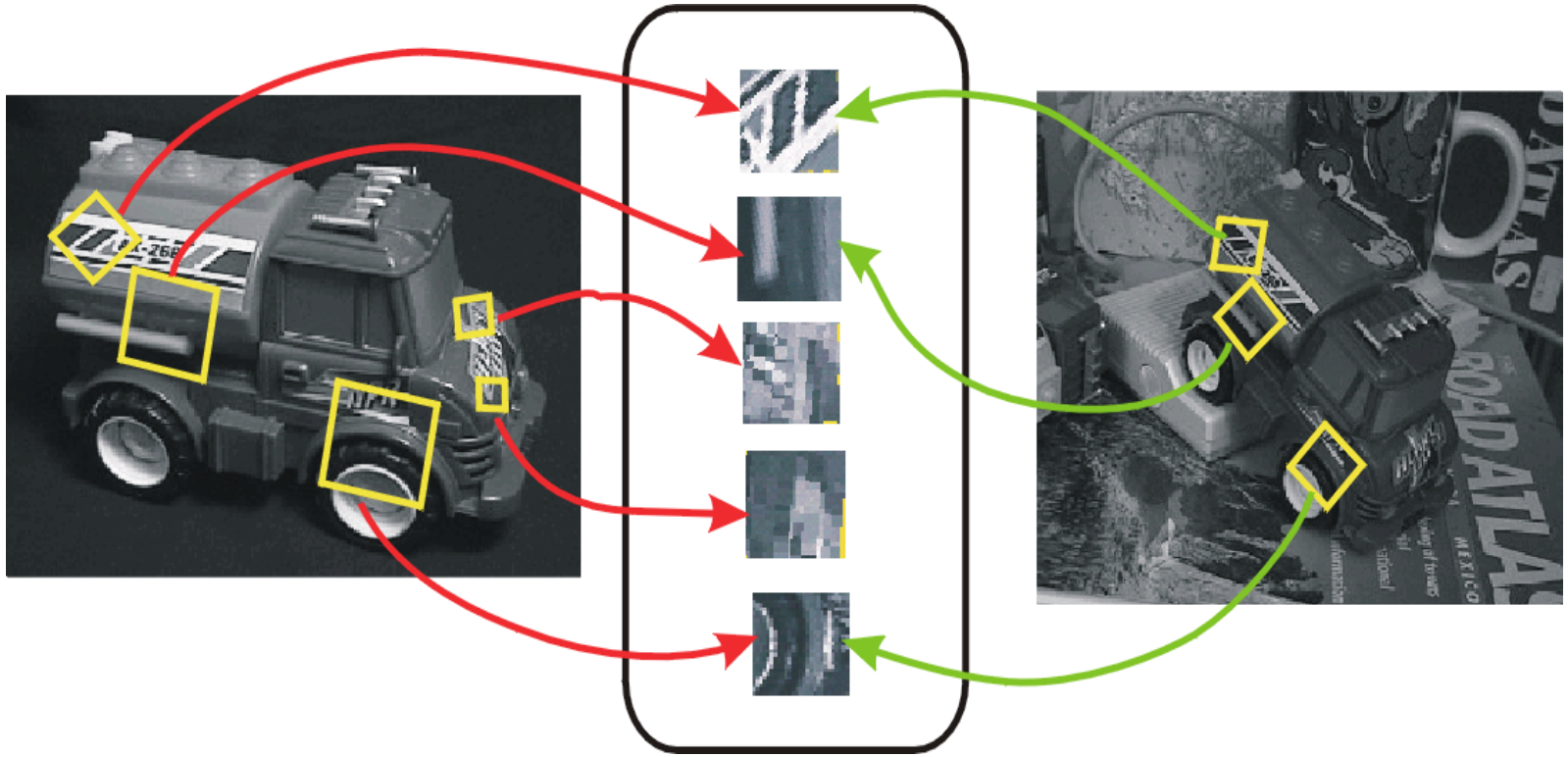
We'd like to find the same features regardless of the transformation

- This is called transformational ***invariance***
- Most feature methods are designed to be invariant in some way …
  - Translation, 2D rotation, scale
- Some can also handle
  - Limited 3D rotations (SIFT works up to about 60 degrees)
  - Limited affine transformations
  - Limited illumination/contrast changes

Invariant local features

Find features that are invariant to transformations

- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, …



Feature Descriptors

GEORGETOWN UNIVERSITY

How to achieve invariance

Need both of the following:

1. Make sure your detector is invariant
   - EG invariant to translation and rotation
   - Scale is trickier
     - common approach is to detect features at many scales using a Gaussian pyramid (e.g., MOPS)
     - Other methods find “the best scale” to represent each feature (e.g., SIFT)
2. Design an invariant feature *descriptor*
   - A descriptor captures the information in a region around the detected feature point
   - The simplest descriptor: a matched filter (fixed size kernel)
     - What’s this invariant to?
   - Let’s look at some better approaches…

Some "Invariant" Features

- When orientation matters …
  - HOG
  - Hough

- Highly Invariant
  - MOPS
  - Harris
  - SIFT

Orientations - I

- Gradient magnitude is affected by illumination changes
  - but gradient direction isn't

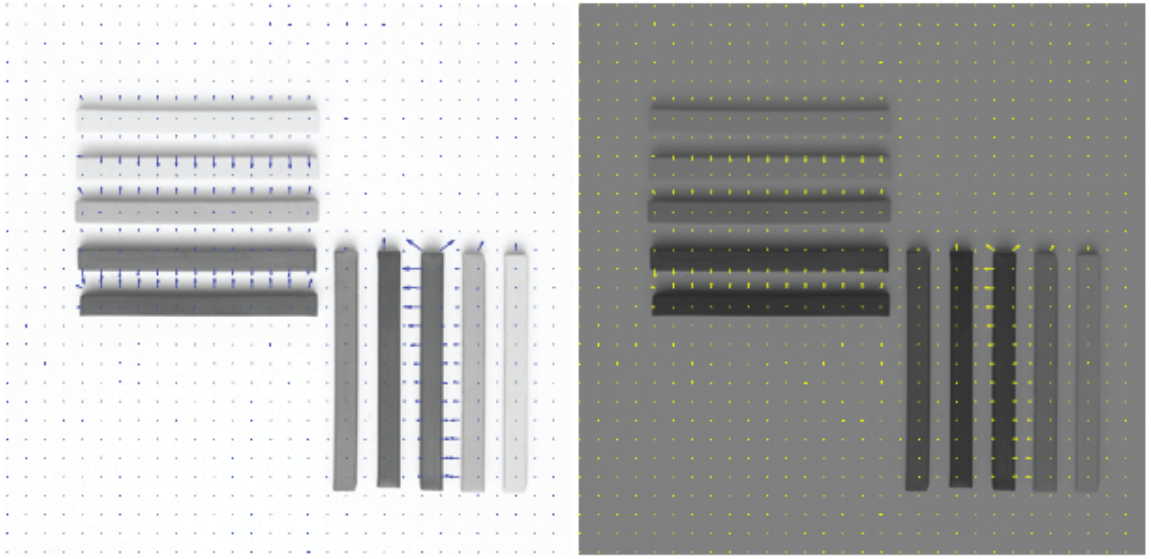


FIGURE 5.7: The magnitude of the image gradient changes when one increases or decreases the intensity. The orientation of the image gradient does not change; we have plotted every 10th orientation arrow, to make the figure easier to read. Note how the directions of the gradient arrows are fixed, whereas the size changes. *Philip Gatward © Dorling Kindersley, used with permission.*

GEORGETOWN UNIVERSITY

Orientations - II

- Notice larger gradients are “better”
  - we know the orientation “better”; associated image points are “more interesting”

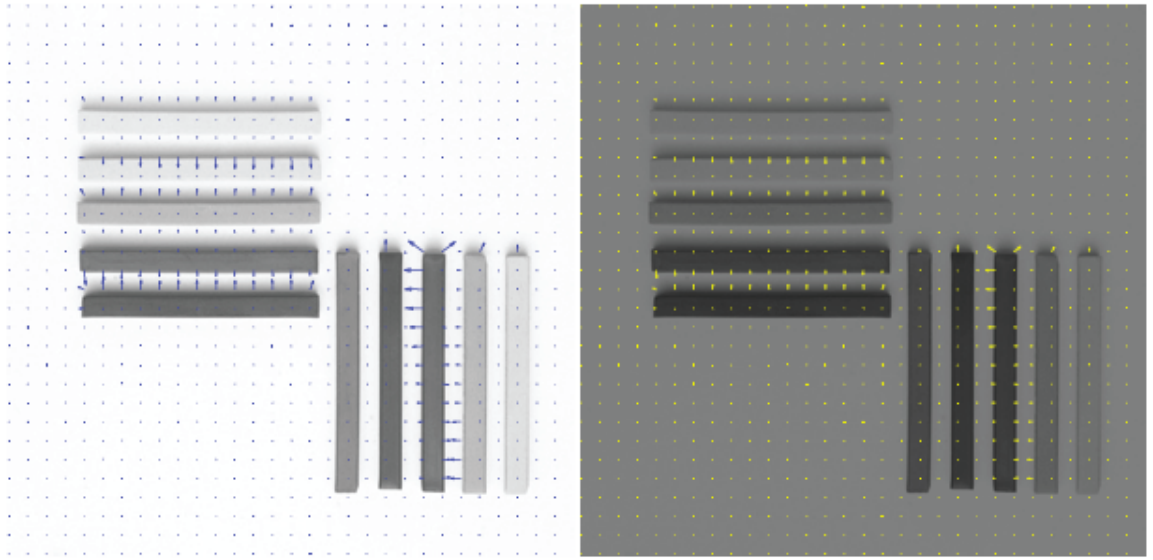


FIGURE 5.7: The magnitude of the image gradient changes when one increases or decreases the intensity. The orientation of the image gradient does not change; we have plotted every 10th orientation arrow, to make the figure easier to read. Note how the directions of the gradient arrows are fixed, whereas the size changes. *Philip Gatward © Dorling Kindersley, used with permission.*

Orientation Histograms (at different scales)

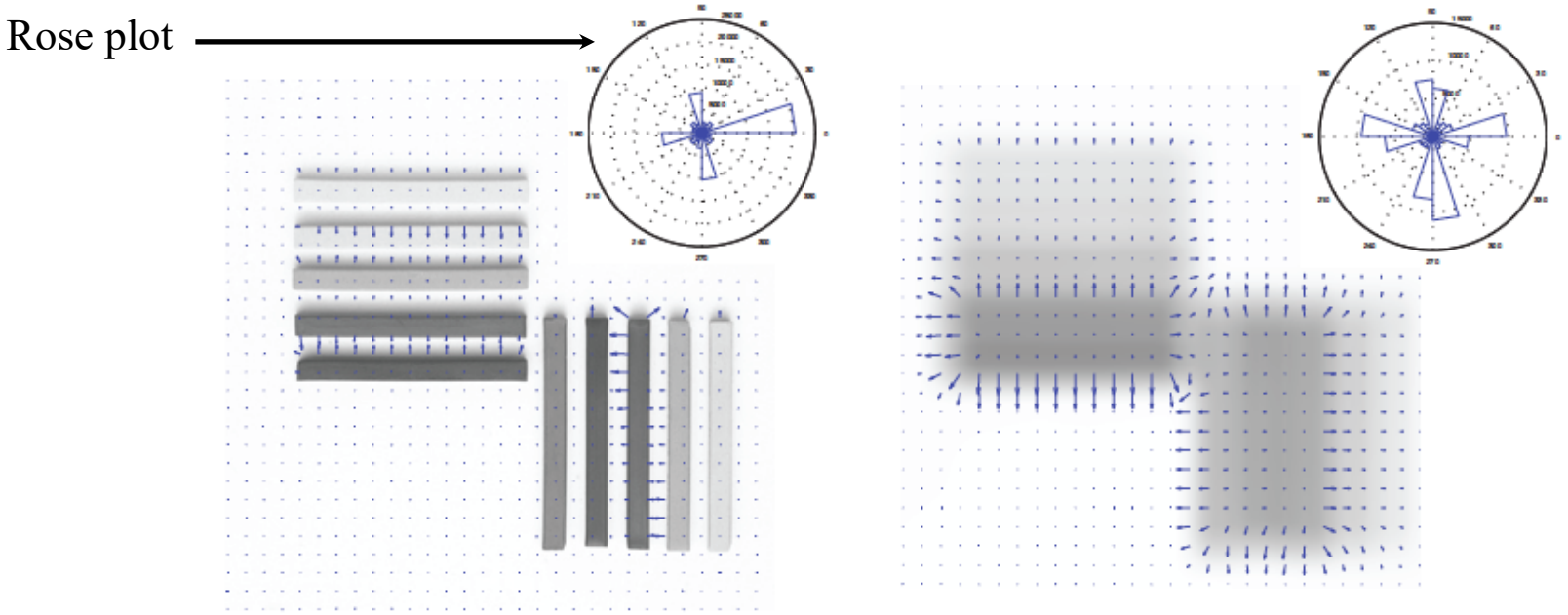


FIGURE 5.8: The scale at which one takes the gradient affects the orientation field. We show the overall trend of the orientation field by plotting a rose plot, where the size of a wedge represents the relative frequency of that range of orientations. **Left** shows an image of artists pastels at a fairly fine scale; here the edges are sharp, and so only a small set of orientations occurs. In the heavily smoothed version on the **right**, all edges are blurred and corners become smooth and blobby; as a result, more orientations appear in the rose plot. *Philip Gatward © Dorling Kindersley, used with permission.*

Orientation Histograms Vary

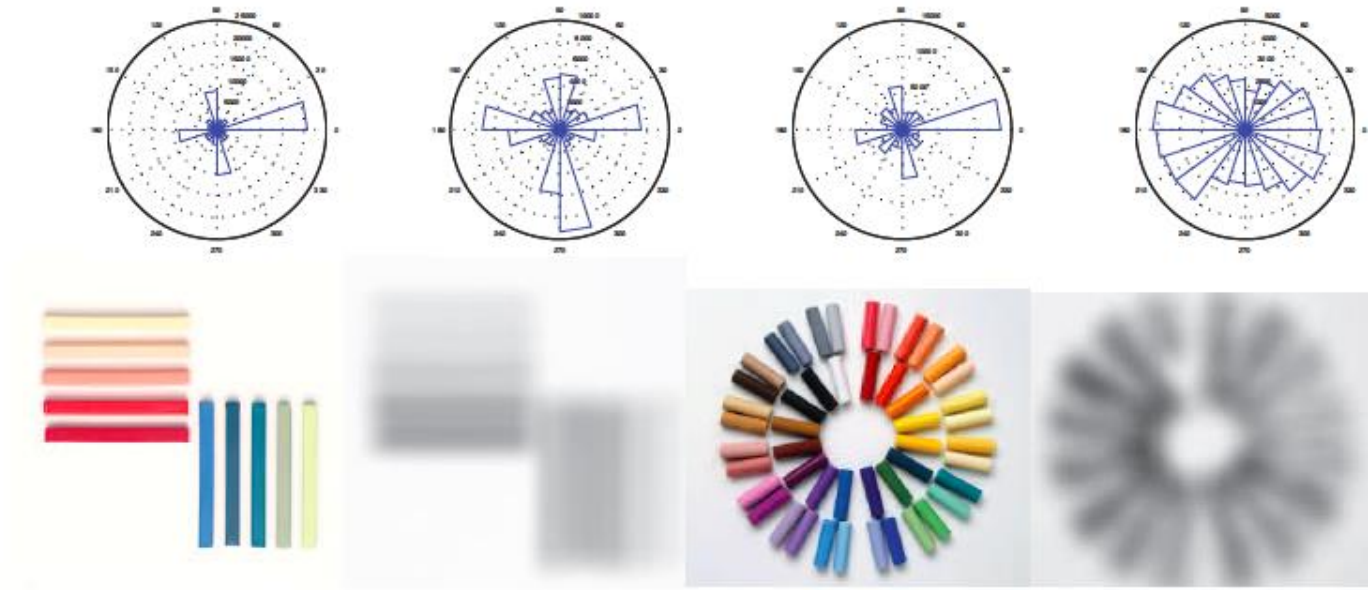


FIGURE 5.9: Different patterns have quite different orientation histograms. The **left** shows rose plots and images for a picture of artists pastels at two different scales; the **right** shows rose plots and images for a set of pastels arranged into a circular pattern. Notice how the pattern of orientations at a particular scale, and also the changes across scales, are quite different for these two very different patterns. *Philip Gatward © Dorling Kindersley, used with permission.*

GEORGETOWN UNIVERSITY

Building Orientation Representations

- GOAL: We would like to represent a pattern in an image patch
  - to detect things in images (recognition)
  - to match points in one image to corresponding points in another image (matching)
- Necessary properties
  - we have to know which (image property) patch to describe
  - think of this as knowing the center and size of an image window
- Desirable features (dependent upon feature!)
  - representation doesn't change much if the center is slightly wrong
  - representation doesn't change much if the size is slightly wrong
  - representation is distinctive
  - representation doesn't change much if the patch gets brighter/darker
  - large gradients are more important than small gradients

Histograms of Oriented Gradients

- Necessary properties
  - we have to know which patch to describe
  - think of this as knowing the center and size of an image window

assume window size is known

- Desirable features
  - representation doesn't change much if the center is slightly wrong
  - representation doesn't change much if the size is slightly wrong
  - representation is distinctive
  - representation doesn't change much if the patch gets brighter/darker
  - large gradients are more important than small gradients

Use histograms

Use orientations

Weight orientation histogram entries

Break window into boxes, describe each Separately (does order matter?)

GEORGETOWN UNIVERSITY

Histograms of Oriented Gradients

- Strategy:
  - break patch up into blocks
  - construct histogram representing gradient orientations in that block
    - which won't change much if the patch moves slightly
    - entries weighted by magnitude

- Variants
  - histogram of angles
  - histogram of gradient vectors, length normalized by block averages

HOG features

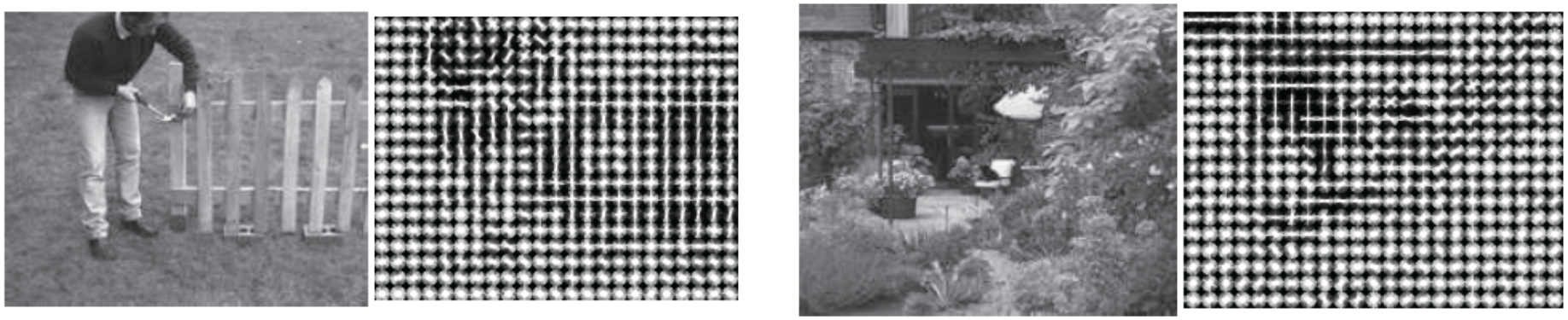


FIGURE 5.15: The HOG features for each the two images shown here have been visualized by a version of the rose diagram of Figures 5.7–5.9. Here each of the cells in which the histogram is taken is plotted with a little rose in it; the direction plotted is at right angles to the gradient, so you should visualize the overlaid line segments as edge directions. Notice that in the textured regions the edge directions are fairly uniformly distributed, but strong contours (the gardener, the fence on the **left**; the vertical edges of the french windows on the **right**) are very clear. This figure was plotted using the toolbox of Dollár and Rabaud. *Left:* © *Dorling Kindersley, used with permission. Right: Geoff Brightling* © *Dorling Kindersley, used with permission.*

HOG - Crucial Points

- Gradient orientations are *not affected by intensity*
- Orientations with *larger magnitude are more important*
  - *Use weighting scheme*
- Describe an image window of known location, size
  - Histograms reduce the effect of poor estimate of location, size
  - Break window into subwindows
    - for each, compute an orientation histogram, weighting orientations by magnitude
  - **Numerous variants available …**

Feature Patterns and the Hough Transform

- A line is the set of points (x, y) such that $(\sin\theta)x + (\cos\theta)y + d = 0$
  - Different choices of θ, d>0 give different lines
- For any (x, y) there is a one parameter family of lines through this point, given by $(\sin\theta)x + (\cos\theta)y + d = 0$
- Each point gets to vote for each line in the family; if there is a line that has lots of votes, that should be the line passing through the points

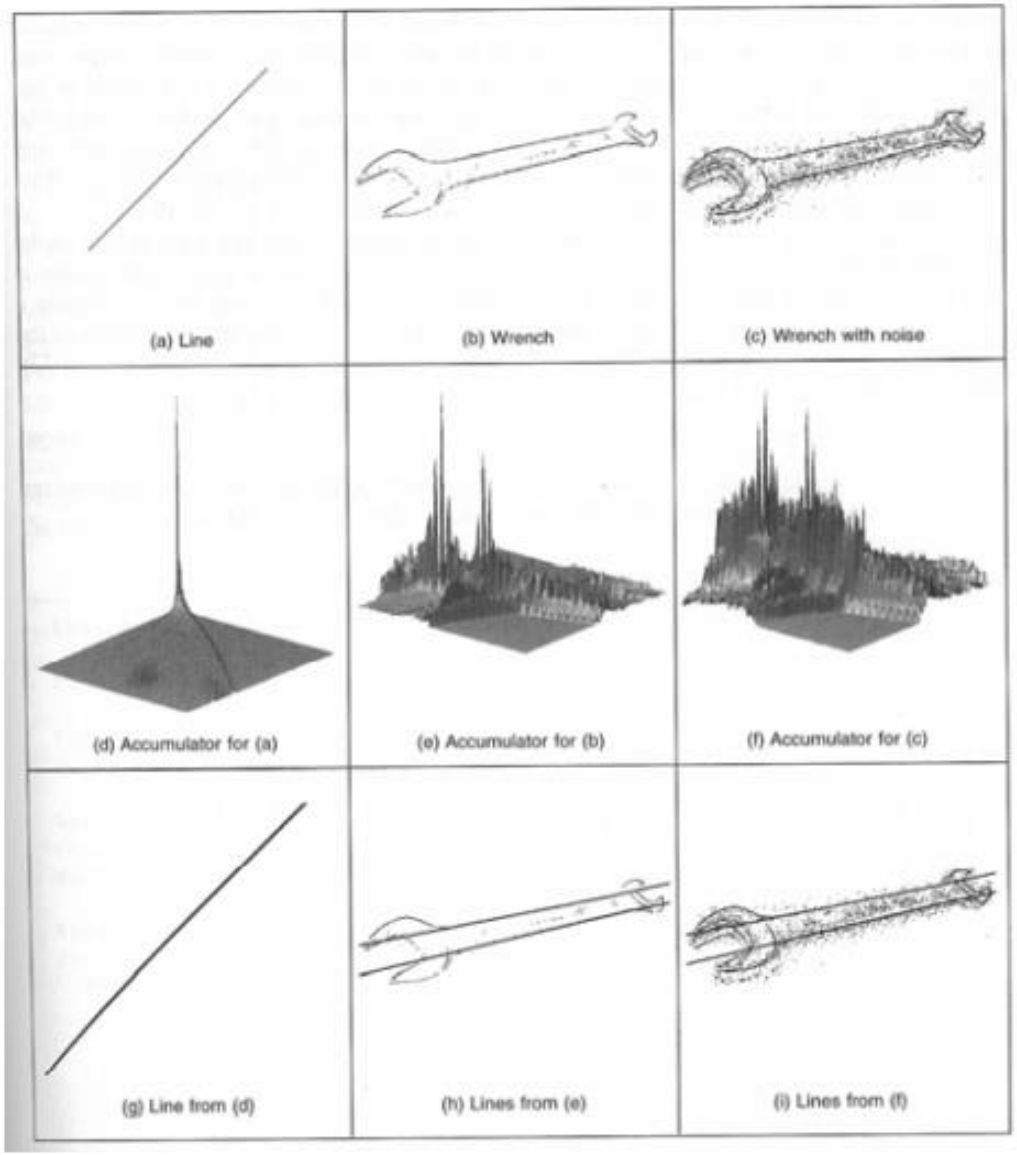
Hough Example

- Assume input features are thresholded edge features.
- Hough transform will help to “connect” features and identify edge contours, direction, ….

Thresholded edge images

Visualizing the accumulator space
The height of the peak will be defined by the number of pixels in the line.

Thresholding the accumulator space and superimposing this onto the edge image



GEORGETOWN UNIVERSITY

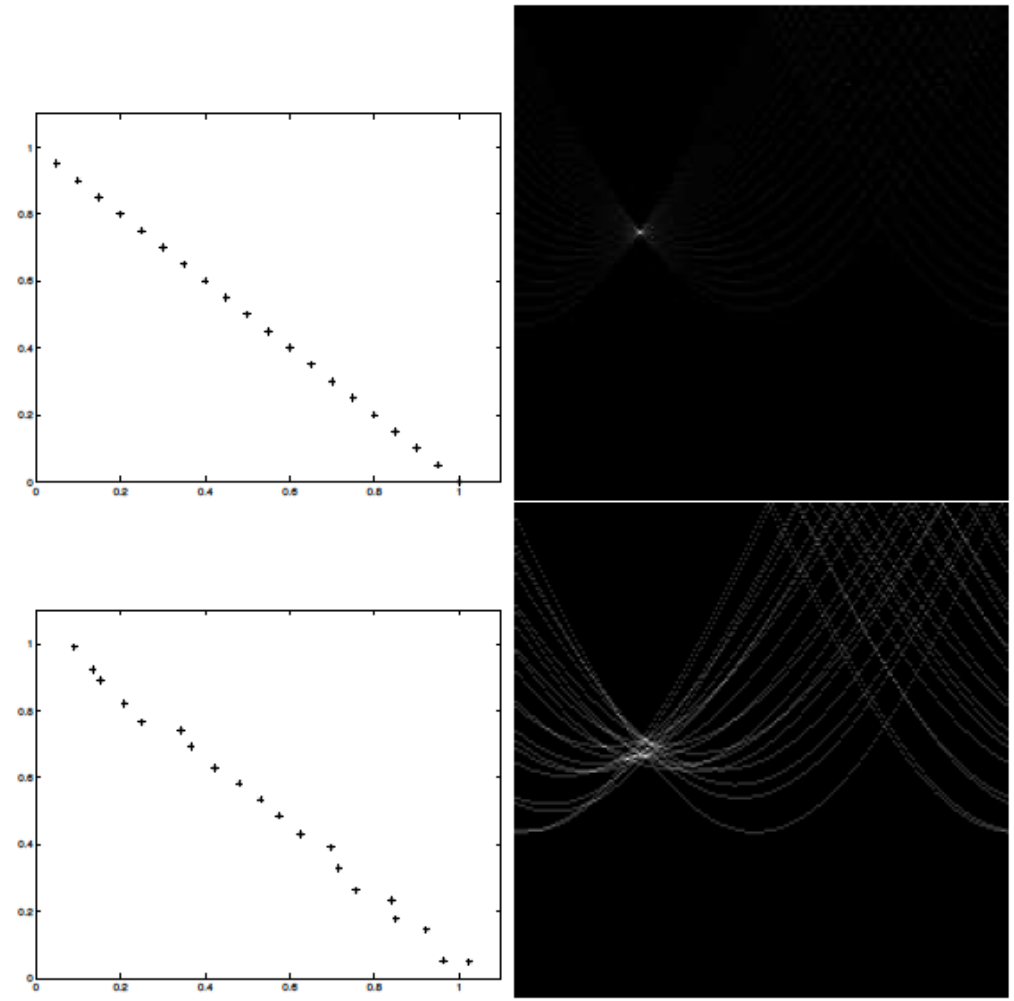


FIGURE 10.1: The Hough transform maps each point like token to a curve of possible lines (or other parametric curves) through that point. These figures illustrate the Hough transform for lines. The **left-hand column** shows points, and the **right-hand column** shows the corresponding accumulator arrays (the number of votes is indicated by the gray level, with a large number of votes being indicated by bright points). The **top** row shows what happens using a set of 20 points drawn from a line. On the **top right**, the accumulator array for the Hough transform of these points. Corresponding to each point is a curve of votes in the accumulator array; the largest set of votes is 20 (which corresponds to the brightest point). The horizontal variable in the accumulator array is θ, and the vertical variable is r; there are 200 steps in each direction, and r lies in the range $[0, 1.55]$. On the **bottom**, these points have been offset by a random vector, each element of which is uniform in the range $[0, 0.05]$. Note that this offsets the curves in the accumulator array shown next to the points and the maximum vote is now 6 (which corresponds to the brightest value in this image; this value would be difficult to see on the same scale as the top image).

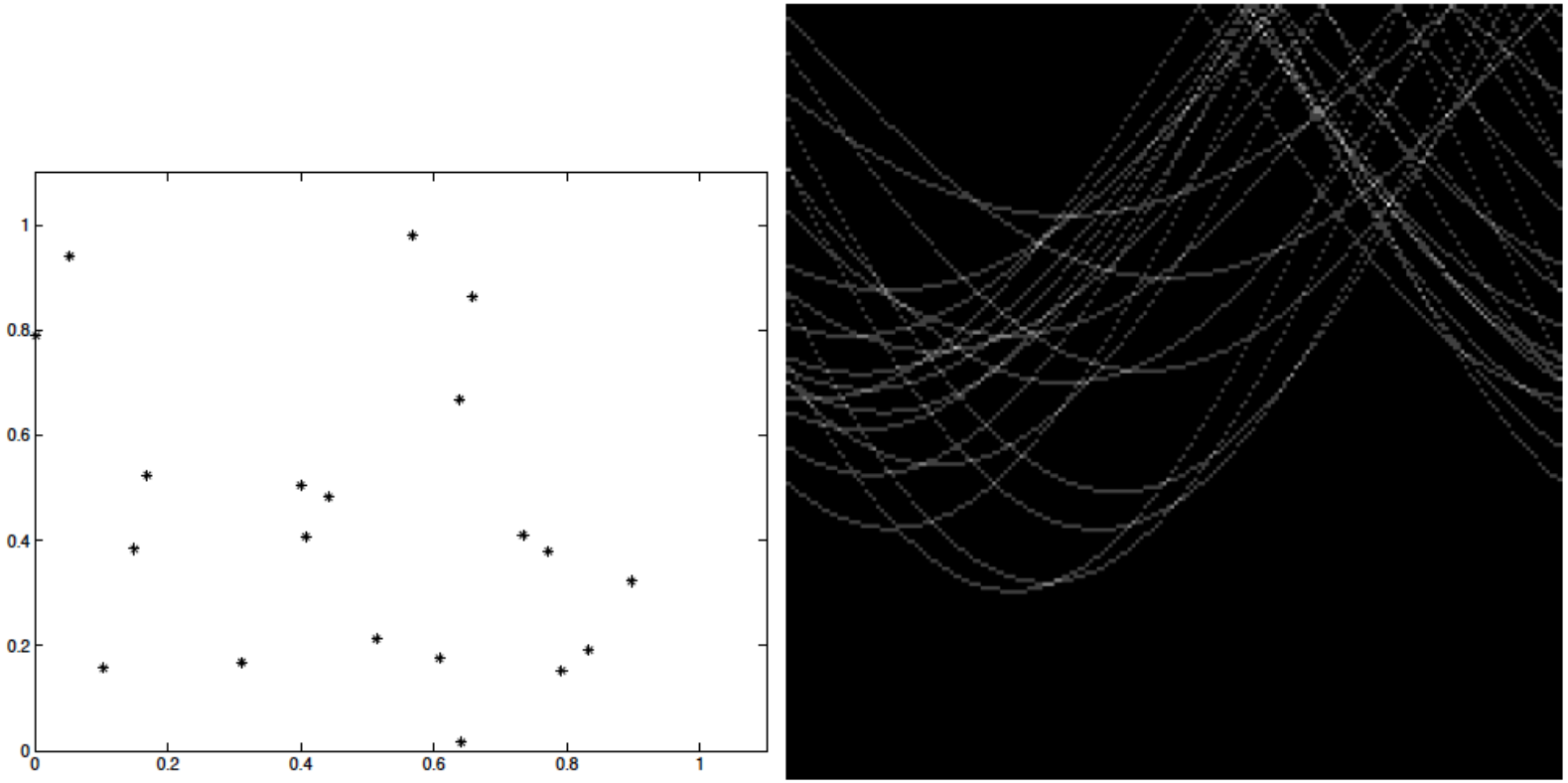


FIGURE 10.2: The Hough transform for a set of random points can lead to quite large sets of votes in the accumulator array. As in Figure 10.1, the **left-hand column** shows points, and the **right-hand column** shows the corresponding accumulator arrays (the number of votes is indicated by the gray level, with a large number of votes being indicated by bright points). In this case, the data points are noise points (both coordinates are uniform random numbers in the range $[0, 1]$); the accumulator array in this case contains many points of overlap, and the maximum vote is now 4 (compared with 6 in Figure 10.1).

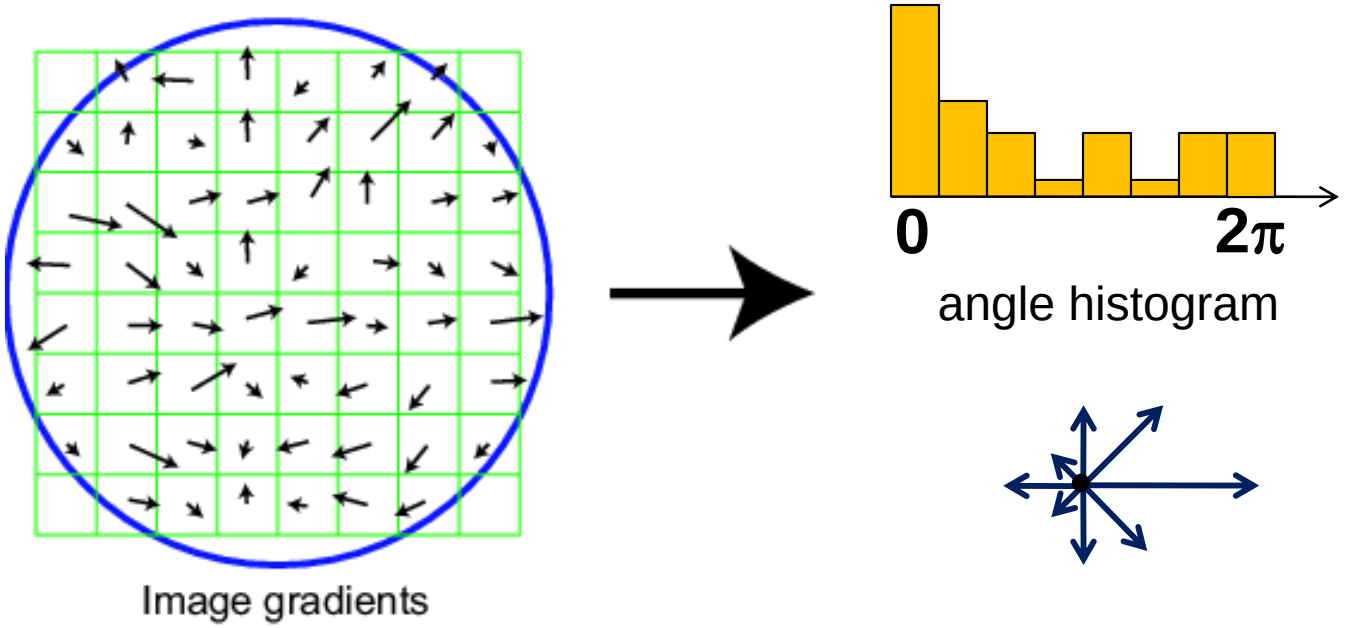
Mechanics of the Hough transform

- Construct an array representing θ, d
- For each point, render the curve (θ, d) into this array, adding one at each cell
- Difficulties
  - how big should the cells be?
    - too big - can't distinguish between quite different lines
    - too small - noise causes lines to be missed
  - How many lines?
    - count the peaks in the Hough array
  - Who belongs to which line?
    - tag the votes
- ***Hardly ever satisfactory in practice***
  - problems with noise and cell size defeat it

Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations

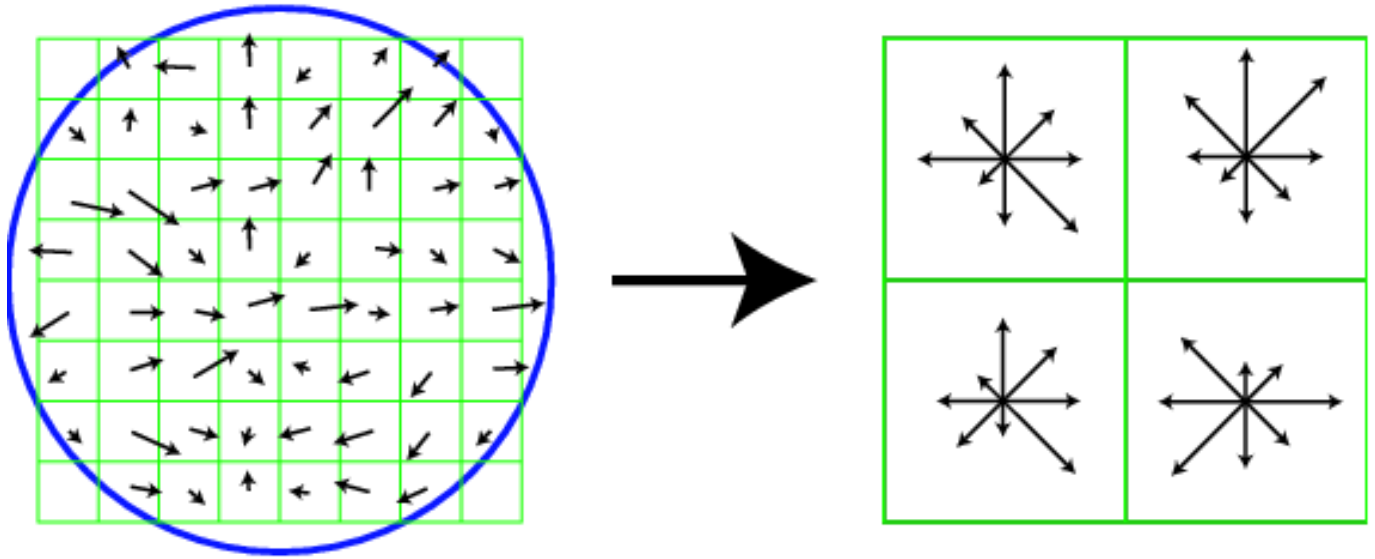


Adapted from slide by David Lowe

SIFT descriptor

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor

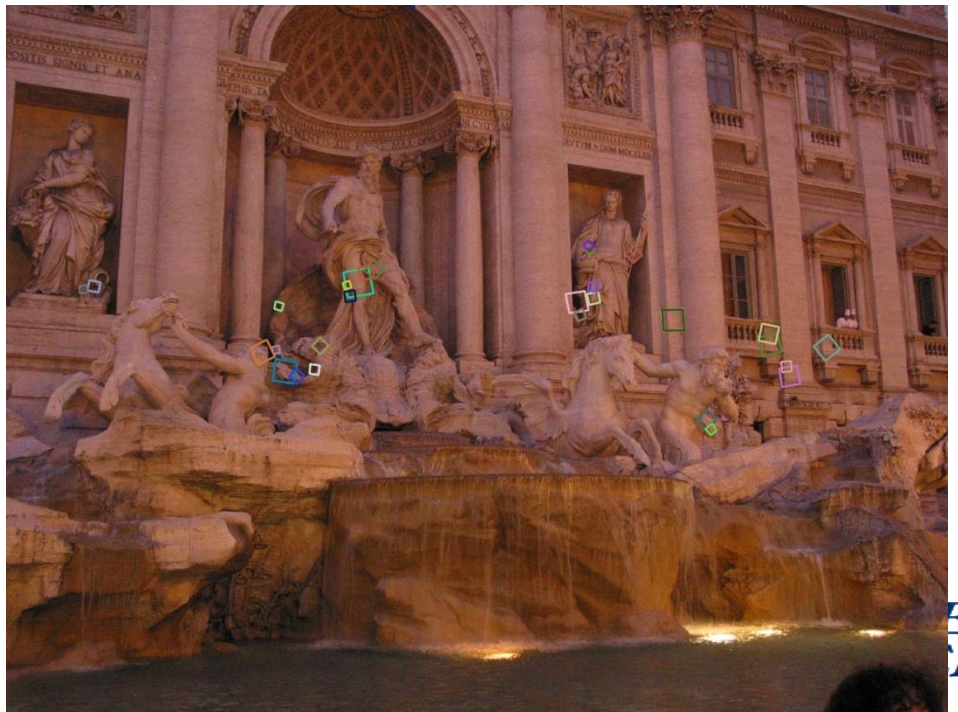


Adapted from slide by David Lowe

Properties of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT



MOPS: Feature Matching Approach

- We need a way to describe regions around each feature, so that matching is robust
- How do you account for changes in viewpoint or scale?
- Tradeoff between discriminative power and invariance to appearance changes, may be domain specific

Rotation invariance for feature descriptors

Find dominant gradient orientation of the image patch

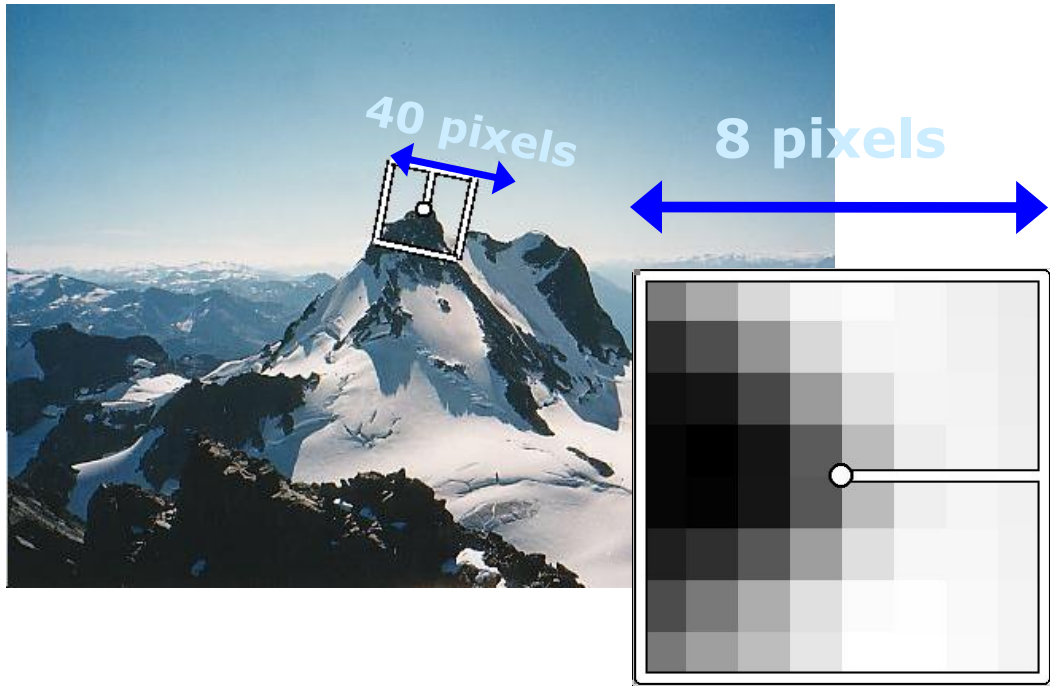
- Rotate the patch according to this angle
  - If this gradient is dominant in all instances of the feature in all images, this pre-alignment step assures rotation invariance

Figure by Matthew Brown

Multiscale Oriented PatcheS descriptor

Take 40x40 square window around detected feature

- **Scale** to 1/5 size (smooth and downsample to remove noise or localization errors)
- **Rotate** to horizontal
- Sample 8x8 square window centered at feature
- **Intensity** normalize the window by subtracting the mean, dividing by the standard deviation in the window



Adapted from slide by Matthew Brown

GEORGETOWN UNIVERSITY

Detections at multiple scales (MOPS with Scale Space)

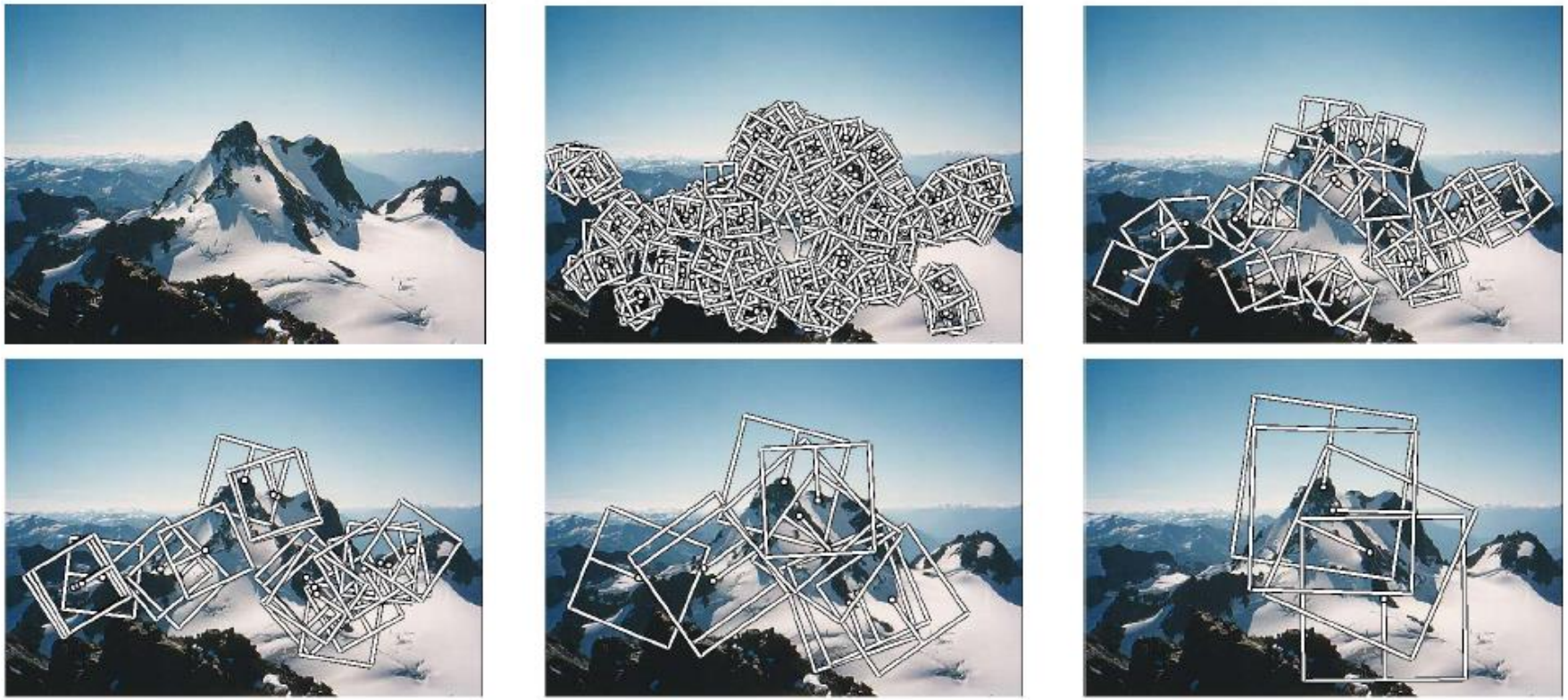


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

GEORGETOWN UNIVERSITY