CMSC 491/691: Computer Vision **Spring 2024**

Lecture 6: 2024-02-14 Image Features 2

Lecturer: Tejas Gokhale *Scribe: Olivia Amaral*

6.1 Recap of Last Lecture

This document contains detailed notes from the image features 2 lecture, refer to slides as well for even more context when following along with this scribing

- Idea of a feature: important points in the image, which are useful for recognition. Useful because we can use this for detection, description, matching, and searching
  - Important aspects: shape is conveyed, unique features, and unusual image regions
  - Features can be detected using the Harris Corner Detector, where corners = good features
  - Once you detect edges, you can detect corners. Edges are okay and flat regions are not good
  - This can be done by moving a window inside the image
  - If error is high, it is a corner. Low error is an edge... the initial summation function is expensive to compute, so instead we use Taylor Series to get two eigenvalues. All we have to do is define a thresholding function and a good features has both eigenvalues in that region
- If lambda 1 and lambda 2 are high, it's a corner
- If lambda 1 is high and lambda 2 is low, it's a vertical edge
- Covariance - features that look the same should translate and scaling is not covariant. So, a good corner for a small object no longer a good corner because scaled it would look like an edge and would give low error
- In detecting window size, each Laplacian is like a different window, as you keep increasing the variation of the Laplacian it is like you are increasing the window size
- Slide 15 shows 6 different images with various standard deviations applied to them. The number above the image is the standard deviation and anything red is peak response which is considered to be a good feature

6.2 Feature Descriptors

Describe represents the point we collected with useful information by storing pixel identities/values and 1 pixel can be stored from a 3x3 neighborhood. However, if the lighting conditions or intensity changes, then the same point intensities will be different (called image patch)

6.2.1 Image Gradients

- Strong x derivatives, storing how image intensities change instead of actual values
- Invariant to actual intensity values
- Losing some useful color information though doesn't maintain color data
- This is a better idea than the first

6.2.2 Color Histogram

- Counts the colors in the image. Not very useful because we are limited by color names
- X axis stores pixel intensities, y axis stores number of pixels. Think of this as a scan of the image and seeing how many pixels have each intensity value
- Nice idea but we can do something better. We don't want to count each of these 0 to 255. We can do ranges on x axis (0-16 or 0-32 for example) which is computationally better
- Could store three histograms with 16 bins, one histogram for red, green, and blue
- Benefit of this: maintain color data and easy to scale. If image is scaled up, histogram would be the same

6.2.3 Spatial Histograms

- Can compute multiple histograms for different corners/sections of the image
- If you had same image with jumbled pixels, the histogram would look the same
- Localizing regions of image and plotting histograms for each image
- Better than just doing color histogram
- Retains rough spatial layout
- Some invariance to deformations

6.2.4 Orientation Normalization

- Another idea. Stores color and intensities, but also orientation
- In addition to pixel location and scale, also store the angle/orientation
- Allows us to distinguish between different corners. This would solve the geometrical problem

All descriptors listed above are simple, state of the art don't use. There are many more feature detectors that exist.

Invariance vs Discriminability:

- Two important properties every image should have
- Should handle transformation, contrast, illumination, rotation

6.3 Scale Invariant Feature Transform (SIFT)

6.3.1 SIFT

- Very good invariant feature detector
- Has several components. Does detection, localization, orientation, description
- Both a feature detector and feature descriptor. Similar to color histogram and Harris Corner Detector combined, but better than both

6.3.2 History of SIFT

- Rejected multiple times because most people didn't know what was novel about the idea
- Highly impactful and eventually found a good venue to publish and became the most highly cited paper in all of engineering and science in less than a year

6.3.3 Scale Invariant Feature Transform

- Steps in doing this. 16x26 grid/patch then doing edge detection and computing angle it makes between x and y. Minus 90 degrees
- Throwing out weak edges means not using edges below a certain threshold. Edges have magnitudes
- Pixel intensities of 50 and 75, there's an intensity change of 25. 75 to 80 is only an intensity of 5 which means it's not a strong edge.
- Histogram of only the orientations, not the magnitudes
- For every edge we have magnitude and angle/orientation. We've thrown out low magnitude and store angles for all the others.
- Plot is from 0 to 2pi because those are angles. Y axis is how many of those edges there are
- Arrow picture thing on bottom is a visualization of the histogram. Its also a histogram, x=0 is longer as to say there are that many. Length of each line is same of the count of the histogram.
- Shift bins means you always get the angle where count is biggest first. We are just circularly shifting the list so that the max is first.. So, the order matters. Sorting this would make us lose angles. Maximum is like angle zero. Not a sort operation, but a shift operation
- 8 ranges (0-45, 45-90, etc)

6.3.4 SIFT Descriptor

- Dividing 16x16 patch into 4 patches of 4x4. Basically do last part (angle to histogram) but repeat for smaller patches
- 16 cells * 8 = 128 dimensional

6.3.5 Properties of SIFT

- Sift is super robust because it can handle changes in viewpoints up to 60 degrees
- Can handle changes in illumination because it stores edges
- Some edges might become weak according to our threshold
- Ex: image has a triangle, with high intensity inside and low intensity outside. Stores arrows pointing in that perpendicular direction to lines of triangle
  - In very extreme cases when angle drops below threshold wouldn't be computed. Very extreme though
  - Rotating a triangle would change the angles, but because histogram is created, etc.. the descriptor would still look the same
- This is a fast feature descriptor computationally
- Images on slide shows same sculptures where the boxes are the features that SIFT detected
  - Able to detect same features regardless of viewpoint/illumination
- Can still be detected if there's occlusion in the image

6.4 Summary

We have now learned how to detect and describe features, the next step is figuring out how to match them

6.5 Feature Matching

- This is an "approximate search" problem. We have a list of features, each represented by a list or vector, we just want to find a similar one
- Converted image to list of features, we need to define a distance function to compare. Then apply that function on all features to find the best one
- Histogram is nothing but a list
- Better Approach to euclidean match... calculating a ratio where a big number would mean ambiguous matches. If we compute ratio from the example on the slides, we would probably get 1. Which shows its ambiguous.
- Sometimes weird matches come up and we have outliers, but this topic will be revisited at a later date
- Figure out which are true matches by comparing distance between the feature descriptions. Now we can count number of true and false matches
- Maximize number true positives
- Human annotation. This is expensive though because you'd need to employ a human to do this
- The example on the slides shows that the true positive = 600/800. Meaning the system says 600 matches, human said 800 matches so that's why true positive

- Recall and specificity on graph: ideal location is top left, which would be more true positives and less false positives
- For the ROC curve, idea is to push the curve as high as possible