Readings

- Grauman 3.1, 3.2 Intro, Detection of Interest Points & Regions
- Klette 9.1 Invariance, Features and Sets of Features
- Szeliski 4.1, 4.1.1 Points & Patches, Feature Detectors

Summary

Motivation

- How can we stitch two images together?
  - matching the parts which are the same on both image
  - aligning the images based on the matches

Image Matching

- three step process: finding key points, computing descriptors, matching descriptors
- "features": catch-all term for key points + descriptors
- Good local features are repeatable yet distinct, efficient to compute and local in extent

Harris Corner Detection

- A local area is distinct if it has a high SSD error wrt surrounding; this occurs at"corners"
- Computing SSD is inefficient; instead, estimate indirectly via H, the second moment matrix
- Corners are found where eigenvalues of H matrix are both sufficiently large
- For efficiency, approximate with a response function $R = \det(H) - k\, \mathrm{trace}^2(H)$, which does not require explicit eigenvalue decomposition
- Areas with high R are likely corners – localize to single pixels via non-maximum suppression

Invariances

- Invariance vs. equivariance
- Harris corners are equivarient to geometric transformations such as translation, rotation, semi-invariant to photometric transformations and not invariant to scaling
- Can use automatic scale selection to find keypoints by sweeping over a range of scales and searching for the scale with a local maximum
- Laplacian of Guassian (LoG) is a scale-sensitive key point detector which detects "blobs"

FAQs

Q: Is the H-matrix positive semi-definite by definition?

A: Yes. H by definition is a positive semi-definite matrix because $[u\ v]\ H \begin{bmatrix} u \\ v \end{bmatrix} = E$ where E is the SSD error and the SSD error is >= 0. This means that we will always have non-negative eigenvalues. Actually, the only way for E = 0 for a local patch is a perfectly constant / flat region but this does not occur in real images.

Q: In slide 18, to find the corner(s) in an image, we will have one E(u,v) value per window W, which we will know if a corner exists in a W with the H matrix. The u and v used for all the W are the same right?

A: We evaluate each window to see if it contains a corner or not. The window is located at x,y. To find out if that window has a corner, we use image content within that window, denoted by w(x,y) and shifting it around (in theory). The shifting is indexed by (u,v). In that sense we use the same (amount of) shifts for each window. In practice, we don't do any actual shifting, we simply compute the H matrix associated with the pixel at x,y, from which we will evaluate if a corner exists there or not.

Q. Is the visualization in slide 19 illustrating the computation of E(u,v) for the various W in an image? Is the u,v used in this slide showing the shift in W?

A. No the animations were illustrating the E(u,v) for only 1 w.

Q: Why is DoG approximation of LoG cheaper?

A: The LoG is not a separable kernel so it needs to be applied in 2D whereas Gaussians (and therefore DoG) is a separable kernel that can be applied twice in 1D. Therefore, we can apply a DoG as 4 1D kernels rather than apply the LoG as a single 2D kernel.

Where do we gain then? Suppose the kernel dimensions are mxm. Any time mxm > 4 x m, so any time our kernel dimension m > 4.

More details here: https://dsp.stackexchange.com/questions/51971/how-is-laplacian-of-gaussian-log-implemented-as-four-1d-convolutions/54279#54279

Q: Is computing SSD equivalent to doing auto-correlation?

A: The two are not equivalent. If you expand the original definition,

$$\begin{aligned} E(u,v) &= \sum_{x,y} [I(x+u,y+v) - I(x,y)]^2 \\ &= \sum_{x,y} [I^2(x+u,y+v) - I(x+u,y+v)I(x,y) + I^2(x,y)] \\ &= K_1 - \sum_{x,y} [I(x+u,y+v)I(x,y)] + K_2 \end{aligned}$$

Only the second term would be the autocorrelation.