Homework 4 Questions

Instructions

- 4 questions.
- Write code where appropriate.
- Feel free to include images or equations.
- Please make this document anonymous.
- **Please use only the space provided and keep the page breaks.** Please do not make new pages, nor remove pages. The document is a template to help grading.
- If you really need extra space, please use new pages at the end of the document and refer us to it in your answers.

Questions

Q1: Imagine we were tasked with designing a feature point which could match all of the following three pairs of images. Which real world phenomena and camera effects might cause us problems? Use the MATLAB function *corner* to investigate. $corner(I, 1000)$.

RISHLibrary — Chase — LaddObservatory

A1: From RISHLibrary, we can see that feature points are invariant to illumination. However, shift in camera angle introduces new objects to the image and thus causes feature points to change. As shown in Chase images, blur also causes feature points to change, but not much. Most of the corners are correctly detected in Chase1.jpg and Chase2.jpg with exception of a few different ones. LaddObservatory images show that feature points are not invariant to zooming in of camera. Detected feature points are different in differently zoomed versions of LaddObservatory images.

```
image = imread('Chase1.jpg');
g_image = rgb2gray(image);
C = corner(g_image, 1000);
```






Q2: In designing our feature point, what characteristics might we wish it to have? Describe the fundamental trade-off between feature point invariance and discriminative power. How should we design for this trade-off?

A2:
In designing our feature point, we wish it to be invariant to be accidental image transformations, including both apperance variance (difference in brightness, illumination, etc) and geometric variation (camera rotation, camera translation, etc). However, at the same time, we also wish it to sustain discriminative power, or ability to actually determine the difference between two distinct features.

A good image descriptor should be largely invariant to such transformations. However, a feature that is perfectly invariant has little or no discriminative power. Similarly, a point with high discriminative power has low invariance. This is so because as a feature point invariance increases, the ability for it to distinguish two different points diminishes. For example, descriptors that are rotationally invariant have poor discriminative power - they will map patches that look different to the same descriptor. The problem thus lies on finding the trade-off between feature point invariance and discriminative power.

To find a point at which a point can satisfy both invariance and discriminative power, we can first start off with a perfectly invariant point. Then, we can decrease its invariance little by little until discriminative power that is satisfory is achieved.

Q3: In the Harris corner detector, what do the eigenvalues of the 'M' second moment matrix represent? Discuss both how they relate to image intensity and how we can interpret them geometrically.

A3:
Second moment matrix 'M' as given as:
$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$.

M can also be given as a diagonalized form as:
$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R.$

Consider an ellipse that is formed when $E(u,v) = \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$ is cut by a horizontal slice. Eigenvalues λ_1 and λ_2 of M determined the axis lengths of formed ellipse, which are given as $\frac{1}{\sqrt{\lambda_{max}}}$ and $\frac{1}{\sqrt{\lambda_{min}}}$.
Thus, we can also see that eigenvalues determine whether the region is corner or not. If either one of the eigenvalues is approximately 0, the region is not a corner.

Since eigenvalues show if the region is a corner, edge, or a flat region, it also tells us if the region contains large intensity changes or not. If both eigenvalues are not close to zero, then the region contains a corner, and shifting a window will give large intensity changes. If one eigenvalue is close to zero, then the region contains an edge, and shifting a window in only a certain direction will give intensity change. If both eigenvalues are near zero, then the region is flat, meaning there are little intensity changes in that region.

Q4: Explain the difference between the Euclidean distance and the cosine similarity metrics between descriptors. What might their geometric interpretations reveal about when each should be used? Given a distance metric, what is a good method for feature descriptor matching and why?

A4:
For two vectors $P = (p_1, p_2, ..., p_n)$ and $Q = (q_1, q_2, ..., q_n)$,

Euclidean distance looks at the distance between the two vectors. It is given as:
$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$.

Cosine similarity looks at the angle between the two vectors. It is used when the magnitudes of vectors do not matter. It is given as:
similarity $= \dfrac{P \cdot Q}{\|P\|_2 \|Q\|_2}$, where $a \cdot b = \|a\|_2 \|b\|_2 \cos\theta$.

Their geometric interpretations imply that cosine similarity is better than Euclidean distance when dealing with angles between the two vectors, since it computes the inner product of two vectors. Cosine similiarity is not affected by the magnitude of vectors.

In constrast, Euclidean distance is better than cosine similarity when the magnitudes of vectors need to be taken into consideration. It is used when the angle between the two vectors is not as important as the actual distance between them.

For feature descriptor matching, the feature descriptors are often not normalized. Therefore, it would be better to use cosine similarity method. Two similar features could be close together in terms of angle, but could be far away from each other in terms of magnitude. In that case, Euclidean distance method will determine those two vectors to be as far away from each other, whereas cosine similarity will determine them to be similar. In cases of feature descriptor matching, where the magnitudes of vectors do not matter, cosine similarity is the better method.