[10] Camera Projection

(a) [2] True or False: The orthographic projection of two parallel lines in the world must be parallel in the image.

```
True
```

(b) [3] Under what conditions will a line viewed with a pinhole camera have its vanishing point at infinity?

```
The line is in a plane parallel to the image plane.
```

(c) [5] A scene point at coordinates (400,600,1200) is perspectively projected into an image at coordinates (24,36), where both coordinates are given in millimeters in the camera coordinate frame and the camera's principal point is at coordinates (0,0,f) (i.e., $u_0 = 0$ and $v_0 = 0$). Assuming the aspect ratio of the pixels in the camera is 1, what is the focal length of the camera? (Note: the aspect ratio is defined as the ratio between the width and the height of a pixel; i.e., k_u/k_v.)

```
u = fx/z, so f = uz/x = 24 * 1200 / 400 = 72 mm.
```

[16] Hough Transform and RANSAC

After running your favorite stereo algorithm assume you have produced a dense depth map such that for each pixel in the input image you have its associated scene point's (X, Y, Z) coordinates in the camera coordinate frame. Assume the image is of a scene that contains a single dominant plane (e.g., the front wall of a building) at unknown orientation, plus smaller numbers of other scene points (e.g., from trees, poles and a street) that are not part of this plane. As you know, the plane equation is given by $ax + by + cz + d = 0$.

(a) [8] Define a *Hough transform* based algorithm for detecting the orientation of the plane in the scene. That is, define the dimensions of your Hough space, a procedure for mapping the scene points (i.e., the (X, Y, Z) coordinates for each pixel) into this space, and how the plane's orientation is determined.

Assuming the plane is not allowed to pass through the camera coordinate frame origin, we can divide by d, resulting in three parameters, $A = a/d$, $B = b/d$, and $C = c/d$ that define a plane. Therefore the Hough parameter space is three dimensional corresponding to possible values of A, B, and C. Assuming we can bound the range of possible values of these three parameters, we then take each pixel's (X, Y, Z) coordinates and increment all points $H(p,q,r)$ in Hough space that satisfy $pX + qY + rZ + 1 = 0$. The point (or small region) in H that has the maximum number of votes determines the desired scene plane.

(b) [8] Describe how the *RANSAC algorithm* could be used to detect the orientation of the plane in the scene from the scene points.

Step 1: Randomly pick 3 pixels in the image and, using their (X,Y,Z) coordinates, compute the plane that is defined by these points.
Step 2: For each of the remaining pixels in the image, compute the distance from its (X,Y,Z) position to the computed plane and, if it is within a threshold distance, increment a counter of the number of points (the "inliers") that agree with the hypothesized plane.
Step 3: Repeat Steps 1 and 2 many times, and then select the triple of points that has the largest count associated with it.
Step 4: Using the triple of points selected in Step 3 plus all of the other inlier points which contributed to the count, recompute the best planar fit to all of these points.

1. In Canny edge detection, we will get more discontinuous edges if we make the following change to the hysteresis thresholding:
   (a) increase the high threshold
   (b) decrease the high threshold
   (c) increase the low threshold
   (d) decrease the low threshold
   (e) decrease both thresholds

   Solution c

2. Mean-shift is a nonparametric clustering method. However, this is misleading because we still have to choose
   (a) the number of clusters
   (b) the size of each cluster
   (c) the shape of each cluster

   (d) the window size
   (e) the number of outliers to allow

   Solution d

3. If you are unsure of how many clusters you have in your data, the best method to use to cluster your data would be
   (a) mean-shift
   (b) k-means
   (c) expectation-maximization
   (d) markov random field
   (e) none of the above are good methods

   Solution a

4. Normalized cuts is an NP-hard problem. To get around this problem, we do the following:
   (a) apply k-means as an initialization
   (b) allow continuous eigenvector solutions and discretize them
   (c) converting from a generalized eigenvalue problem to a standard one
   (d) constraining the number of cuts we make
   (e) forcing the affinities to be positive

   Solution b or c

5. To decrease the size of an input image with minimal content loss, we should
   (a) High-pass filter and down-sample the image
   (b) Crop the image
   (c) Apply a hough transform
   (d) Down-sample the image
   (e) Low-pass filter and down-sample the image

   Solution e

6. When applying a Hough transform, noise can be countered by
   (a) a finer discretization of the accumulator
   (b) increasing the threshold on the number of votes a valid model has to obtain
   (c) decreasing the threshold on the number of votes a valid model has to obtain
   (d) considering only a random subset of the points since these might be inliers

   Solution b

7. In which of the following scenarios can you use a weak perspective camera model for the target object?
   (a) A squirrel passing quickly in front of you.
   (b) An airplane flying at a very high attitude.
   (c) The Hoover tower when you are taking a photo of it right in front of it.
   (d) A car beside you when you are driving.

   Solution b

8. What is the biggest benefit of image rectification for stereo matching?
   (a) Image contents are uniformly scaled to a desirable size.
   (b) All epipolar lines intersect at the vanishing point.
   (c) All epipolar lines are perfectly vertical.
   (d) All epipolar lines are perfectly horizontal.
   (e) Epipoles are moved to the center of the image.

   Solution d

9. Which of the following factor does not affect the intrinsic parameters of a camera model?
   (a) Focal length
   (b) Offset of optical center
   (c) Exposure
   (d) Image resolution

   Solution c

13. **(10 points) Linear Filter**
    In this problem, you will explore how to separate a 2D filter kernel into two 1D filter kernels. Matrix K is a discrete, separable 2D filter kernel of size $k \times k$. Assume k is an odd number. After applying filter K on an image I, we get a resulting image I_K.

    (a) (1 point) Given an image point (x, y), find its value in the resulting image, $I_K(x, y)$. Express your answer in terms of I, k, K, x and y. You don't need to consider the case when (x, y) is near the image boundary.

    Solution
    $I_K(x, y) = \sum_{i=1}^{k} \sum_{j=1}^{k} K_{ij} I(x - i + \frac{k}{2}, y - j + \frac{k}{2})$

    (b) (5 points) One property of this separable kernel matrix K is that it can be expressed as the product of two vectors $g \in \mathbb{R}^{k\times 1}$ and $h \in \mathbb{R}^{1\times k}$, which can also be regarded as two 1D filter kernels. In other words, $K = gh$. The resulting image we get by first applying g and then applying h to the image I is I_{gh}. Show that $I_K = I_{gh}$.

$$
\begin{aligned}
I_K(x,y) &= \sum_{i=1}^{k}\sum_{j=1}^{k} K_{ij} I(x - i + \frac{k}{2}, y - j + \frac{k}{2}) \\
&= \sum_{i=1}^{k}\sum_{j=1}^{k} g_i h_j I(x - i + \frac{k}{2}, y - j + \frac{k}{2}) \\
&= \sum_{j=1}^{k} h_j \sum_{i=1}^{k} g_i I(x - i + \frac{k}{2}, y - j + \frac{k}{2}) \\
&= \sum_{j=1}^{k} h_j I(x, y - j + \frac{k}{2})
\end{aligned}
$$

    (c) (4 points) Suppose the size of the image is $N \times N$, estimate the number of operations (an operation is an addition or multiplication of two numbers) saved if we apply the 1D filters g and h sequentially instead of applying the 2D filter K. Express your answer in terms of N and k. *Ignore the image boundary cases so you don't need to do special calculations for the pixels near the image boundary.*

    Solution
    For the 2D filter, there are k^2 multiplication operations and $k^2 - 1$ addition operations for each pixel. In total, $N^2(2k^2 - 1)$.

    For each of the 1D filters, there are k multiplication operations and $k - 1$ addition operations for each pixel. In total, $N^2(4k - 2)$.

    So the number of operations saved is $N^2(2k^2 - 4k + 1)$

16. **(5 points)** You are using k-means clustering in color space to segment an image. However, you notice that although pixels of similar color are indeed clustered together into the same clusters, there are many discontiguous regions because these pixels are often not directly next to each other. Describe a method to overcome this problem in the k-means framework.

    Solution
    Concatenate the coordinates (x, y) with the color features as input to the k-means algorithm.

20. **(3 points)** True / False : Given a set of 3 images as shown in Fig. 7, finding and labeling the image in the center as "containing cat" is considered a *detection* task in recognition. Why or why not?

    Solution
    False. We are not localizing the cat in the image, so this is considered classification.

23. **(3 points)** True / False : If we initialize the k-means clustering algorithm with the same number of clusters but different starting positions for the centers, the algorithm will always converge to the same solution. Why or why not?

    Solution
    False. Different initializations will result in different clusters because they are local minima.

(10 points) Fitting Circles with RANSAC

RANSAC is a powerful method for a wide range of model fitting problems as it is easy to implement. In class, we've seen how RANSAC can be applied to fitting lines. However, RANSAC can handle more complicated fitting problems as well, such as fitting circles. In this problem we will solve for the steps needed to fit circles with RANSAC.

(a) (1 point) What is the minimum number of points we must sample in a seed group to compute an estimate for a uniquely defined circle?

Solution
3 points, as there are 3 degrees of freedom in a circle, the (x,y) coordinates of the center and the radius.

(b) (5 points) If we obtain a good solution that has few outliers, we want to refit the circle using all of the inliers, and not just the seed group. For speed purposes, we would like to keep the same center of the estimated circle from the seed group, and simply refit the radius of the circle. The equation for our circle is given as:

$$(x - x_c)^2 + (y - y_c)^2 = r^2$$

where (x_c, y_c) are the coordinates of the center of the circle, and r is the radius. The error we would like to minimize is given by:

$$E(x_c, y_c, r) = \sum_{i=1}^{m} (\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - r)^2$$

where m is the number of inliers. Derive the new radius r that minimizes this least squares error function.

Solution

$$\frac{\partial E}{\partial r} = -2 \sum_{i=1}^{m} (\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - r)$$

$$r = \frac{1}{m} \sum_{i=1}^{m} (\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2})$$

(c) (4 points) One of the benefits to RANSAC is that we are able to calculate the failure rate for a given number of samples. Suppose we know that 30% of our data is outliers. How many times do we need to sample to assure with probability 20% that we have at least one sample being all inliers? You can leave your answer in terms of log functions.

Solution

$$1 - 0.2 = (1 - (0.7)^3)^k$$

$$k = \frac{\log(1-0.2)}{\log(1-(0.7)^3)}$$

(c) **4 points:** If 30% of the data are outliers, and we want a 20% probability of having at least one sample being all inliers, the number of samples N can be calculated using:
$P(\text{failure}) = (1 - (1 - \epsilon)^n)^k$
Where ϵ is the outlier ratio (0.3), n is the number of points (3), and k is the number of samples.
Solving for k:
$0.2 = (1 - (1 - 0.3)^3)^k$
$k = \frac{\log(0.2)}{\log(1-(0.7)^3)}$
This represents the number of samples required to ensure a 20% success rate.

13. [15 points] Assume that you have captured a point cloud with a single dominant plane (e.g. the front wall of a building) at unknown orientation, plus smaller numbers of other scene points (e.g. trees, poles, a street, etc.) that are not part of this plane. Each point in the point cloud is represented by (x_i, y_i, z_i) coordinates. The equation of a plane is given by $ax + by + cz + d = 0$.

    (a) [5 points] Now, we would like to fit a plane to these points using least squares. Write the linear system whose solution minimizes the objective function in a least squares sense.

    Solution:

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ \vdots & \vdots & \vdots & 1 \\ x_n & y_n & z_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \mathbf{0} \text{ subject to } \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \neq \mathbf{0}$$

    (b) [5 points] Describe how the RANSAC algorithm could be used to fit a plane to these points in the scene.

    Solution: We can choose a randomized subset of consisting of three points from our point cloud and use them to form a triangle. Then, we can take the cross product of the triangle edge vectors to obtain a normal and the barycenter of the triangle to obtain an origin that together define a plane. Using this definition, we find all points that are within some threshold distance along the normal of the plane and denote these as the set of inliers. We repeat the process until the model with the most inliers has been found.

    (c) [5 points] Define a Hough transform based algorithm to fit a plane to these points in the scene. That is, define the dimensions of your Hough space, a procedure for mapping the scene points (i.e. the (x, y, z) coordinates for each pixel) into this space, and how the plane is determined.

    Solution: The Hough space is defined by the parameters $\alpha = a/d$, $\beta = b/d$, and $\gamma = c/d$. Then, each point corresponds to a plane $\alpha x + \beta y + \gamma z + 1 = 0$ in Hough space. We can partition our Hough space using a uniform 3D grid, find all Hough cells through which the plane corresponding to each point passes through, and then find the Hough cell with the most intersections. The center of this cell will correspond to a set of parameters that determine the plane's orientation.

Bonus question: Consider an image that is empty except for two sets of points. One is a set of points distributed roughly uniformly on a circle of radius r centered at point C1, which is near the center of the image. The other set of points is distributed on a circle of radius 2r, which is centered at point C2, which is located inside the other circle. Assume the points in each set are distributed densely enough so that the distances between points on the same circle are smaller than the distances between points on different circles. Describe what segmentation the k-means clustering algorithm would produce for this example, and briefly explain why.

The k-means clustering algorithm would produce two clusters: one for the points on the inner circle (radius r centered at $C1$) and one for the points on the outer circle (radius $2r$ centered at $C2$).

Why:

- **Initial Centroids**: Start randomly but converge near $C1$ and $C2$.
- **Distance Minimization**: Points on the same circle are closer to each other than to points on the other circle.
- **Final Clusters**: One centroid moves to $C1$, clustering points on the inner circle. The other moves to $C2$, clustering points on the outer circle.

This results in two distinct clusters corresponding to the two circles.

(ii) Describe what segmentation the **normalized-cut** algorithm would produce for this example, and briefly explain why.

Using an affinity measure that is based on the distance between pairs of points, and because the distance between several points within a given circle is less than the distance between points in different circles, the normalized-cut algorithm will compute a bi-partition that *correctly* segments this image into the two sets of circular points.

RANSAC for Model Fitting

RANSAC loop:

1. Randomly select a minimum number of points for model fitting
2. Compute a model from these points
3. Find *inliers* to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of model on all of the inliers

- Keep the model with the largest number of inliers

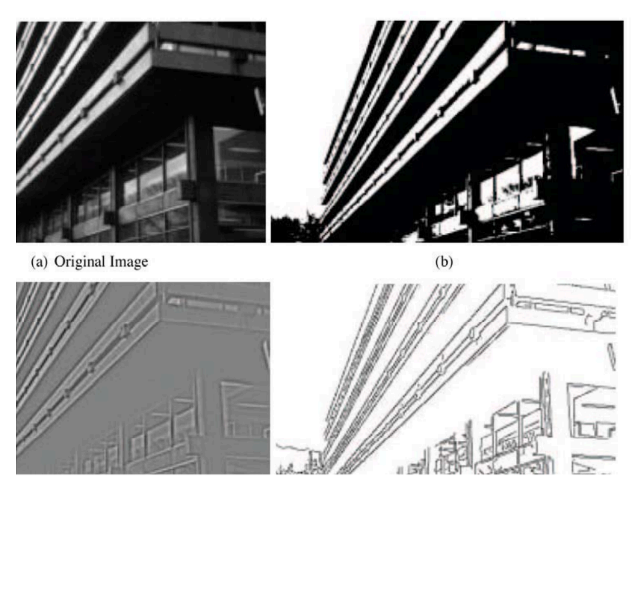
RANSAC Pros/Cons

Pros:
- General method suited for a wide range of model fitting problems
- Easy to implement and easy to calculate its failure rate

Cons:
- Only handles a moderate percentage of outliers without cost blowing up
- Many real problems have high rate of outliers (but sometimes selective choice of random subsets can help)

- Linear filtering:
  - Form a new image whose pixels are a weighted sum of original pixel values
  - Use the same set of weights at each point
- 1D linear filter and 2D linear filters
- Convolution
  - Properties: commutative, associative, distributive, shift, shift-invariance
- Cross-correlation

Filter and edge detection



Possible Operations:

- •Difference of Gaussians
- •Threshold
- •Box
- •Fourier
- •Inverse Fourier
- •Canny
- •Gaussian

K-Means

- Basic idea: randomly initialize the *k* cluster centers, and iterate between the two steps we just saw.
  1. Randomly initialize the cluster centers, $c_1, ..., c_K$
  2. Given cluster centers, determine points in each cluster
     - For each point p, find the closest c_i. Put p into cluster i
  3. Given points in each cluster, solve for c_i
     - Set c_i to be the mean of points in cluster i
  4. If c_i have changed, repeat Step 2
- Properties
  - Will always converge to *some* solution
  - Can be a "local minimum"
    - Does not always find the global minimum of objective function:

$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} \|p - c_i\|^2$$

K-Means Pros/Cons

- Pros
  - Simple, fast to compute
  - Converges to local minimum of within-cluster squared error
- Cons/issues
  - Setting k?
  - Sensitive to initial centers
  - Sensitive to outliers
  - Detects spherical clusters only
  - Assuming means can be computed

13. **(10 points) Fundamental Matrix**

When you show your friend around Stanford campus, he takes a picture I of the Hoover tower. Later he wants to take a bigger picture of the Hoover tower, but his camera has no zoom-in function, so he walks forward for a short distance and takes a new picture I'. Assume there is only a forward translation of the camera perpendicular to the image plane and the movement distance is d.

(a) (5 points) Find the essential matrix F between I and I' in terms of d.

Solution
In general, $E = \hat{t}R$, where $\hat{t}$ is a skew-symmetric matrix related to translation vector. For a forward translating camera, we have

$$R = I, \hat{t} = \begin{bmatrix} 0 & -d & 0 \\ d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

therefore,

$$E = \begin{bmatrix} 0 & -d & 0 \\ d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

(b) (5 points) The tip of the Hoover tower on I is $p = (x\ y)$, what is its corresponding epipolar line l in I'? Express l in terms of x, y and d.

Solution
From $l = Ep$, the epipolar line for point $p = [x\ y\ 1]$ is

$$l = \begin{bmatrix} 0 & -d & 0 \\ d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} -yd \\ xd \\ 0 \end{bmatrix}$$

15. **(10 points) Perspective Projection**

In figure 1, there are two parallel lines l_1 and l_2 lying on the same plane Π. l_1' and l_2' are their projections through the optical center O on the image plane Π'. Let's define plane Π by $y = c$, line l_1 by equation $ax + bz = d_1$, and line l_2 by equation $ax + bz = d_2$.

(a) (3 points) For any point $P = (x, y)$ on l_1 or l_2, use the perspective projection equation below to find the projected point $P' = (x', y')$ on the image plane. f' is the focal length of the camera. Express your answer in terms of a, b, c, d, z and f'.

$$\begin{cases} x' = f'\frac{x}{z} \\ y' = f'\frac{y}{z} \end{cases} \tag{1}$$

Solution
According to the perspective projection equation, a point on l projects onto the image point defined by

$$\begin{cases} x' = f'\frac{x}{z} = f'\frac{d-bz}{az}, \\ y' = f'\frac{y}{z} = f'\frac{c}{z}. \end{cases}$$

(b) (7 points) It turns out l_1' and l_2' appear to converge on the intersection of the image plane Π' given by $z = f'$ and the plane $y = 0$. Explain why.

Solution
This is a parametric representation of the image δ of the line Δ with z as the parameter. This image is in fact only a half-line since when $z \to -\infty$, it stops at the point $(x', y') = (-f'\frac{b}{a}, 0)$ on the x' axis of the image plane. This is the vanishing point associated with all parallel lines with slope $-\frac{b}{a}$ in the plane Π. All vanishing points lie on the x' axis, which is the horizon line in this case.

19. **(5 points)** As shown in figure 3, a point Q is observed in a known (i.e. intrinsic parameters are calibrated) affine camera with image plane Π_1. Then you translate the camera parallel to the image plane with a known translation to a new image plane Π_2 and observe it again.

(a) (2 points) Draw the image points Q_1' and Q_2' on Π_1 and Π_2 on the left figure of Fig. 3. Is it possible to find the depth of the 3D point Q in this scenario? Briefly explain why.

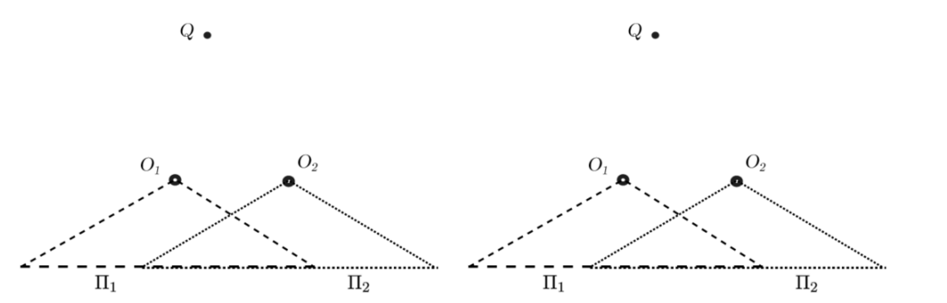


Figure 3: 3D point reconstruction

Solution
The solution of both parts is in figure 4.

No. We cannot determine point Q because it can be any 3D point on the line.

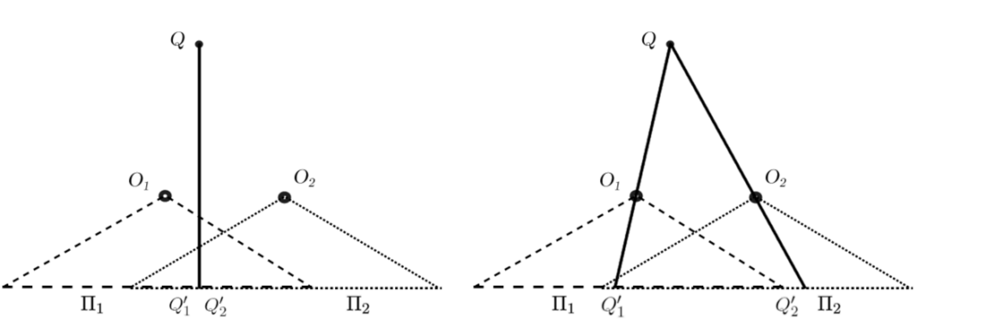


Figure 4: 3D point reconstruction

(b) (3 points) What if this is a perspective camera? Draw Q_1' and Q_2' on the right figure of Fig. 3. Is it possible to find the depth of the 3D point Q in this scenario? Briefly explain why.

Solution
Yes, because we can do triangulation in this case.

(6 points) Fundamental matrix estimation

(a) (2 points) What is the rank of the fundamental matrix?

Solution
The rank is 2.

(b) (4 points) In the 8-point algorithm, what math technique is used to enforce the estimated fundamental matrix to have the proper rank? Explain how this math technique is used to enforce the proper matrix rank.

Solution
In the 8-point algorithm, SVD can be used to enforce the estimated F has rank 2. Specifically, we compute the SVD decomposition $F = U\Sigma V$, and we then zero out diagnals of Σ except for the two largest singular values to obtain $\tilde{\Sigma}$. We can reconstruct $F = U\tilde{\Sigma}V$.

Note: All of the following questions require you to specify whether the given statement is true or false and provide an explanation. No credit will be awarded without a valid

[10] **Corner Detection**

(a) [2] When would detecting corners be more appropriate than detecting edges as an initial step in an application using computer vision?

Detecting corners would be more appropriate when only a sparse set of points are needed, especially facilitating the detection of corresponding points in multiple images. This is used, for example, in stereopsis and motion tracking.

(b) [4] The Harris corner detection algorithm computes a 2 × 2 matrix at each pixel based on the first derivatives at that point and then computes the two eigenvalues of the matrix, λ_1 and λ_2, where $\lambda_1 \le \lambda_2$. How can these two values be used to label each pixel as either a locally smooth region (S), an edge point (E), or a corner point (C)? Give your answer by specifying "Label pixel S if ...", "Label pixel E if ..." and "Label pixel C if ..."

The Harris corner detector first computes $R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$ and then
labels a pixel S if $|R| \approx 0$ (or, alternatively, $\lambda_1 \approx \lambda_2 \approx 0$);
labels a pixel E if $R < T_1 < 0$ (or, alternatively, $\lambda_1 \approx 0$ (corresponding to the direction of the edge) and λ_2 is large (corresponding to the normal direction at the edge)); or
labels a pixel C if $R > T_2 > 0$ (or, alternatively, λ_1 and λ_1 are both large).

(c) [4] Given the two eigenvalues specified in (b), explain in English the rationale and difference(s) between detecting corner points using the criterion $\lambda_1 > T_1$ (which is used in the Tomasi and Kanade algorithm) versus the criterion $\lambda_1\lambda_2 > T_2$ (which is used in the Harris algorithm), where T_1 and T_2 are appropriate thresholds.

Since $\lambda_1 \le \lambda_2$, λ_2 corresponds to the direction of maximum change in intensity and λ_1 corresponds to the direction of minimum change in intensity. Therefore, the criterion $\lambda_1 > T_1$ is used to test if there is more than one direction with a strong edge present. The criterion $\lambda_1\lambda_2 > T_2$ could hold if $\lambda_2 >> \lambda_1$ and yet λ_1 is relatively small compared to λ_2. The criterion $\lambda_1 > T_1$ ensures that both eigenvalues are large as defined by T_1. Ideally, we'd like to ensure that both eigenvalues are large and they are of similar size.

5. [15] **Active Contours**

The energy functional that is used with active contours (snakes) usually contains the three terms: $E_{continuity}$, $E_{smoothness}$, and E_{image} (the first two terms are often combined to define a term called E_{int}).

(a) [9] For each of these three terms explain briefly what it measures, how it is defined, and what happens if the term is omitted.

$E_{continuity} = ||dv/ds||^2$ measures the degree of rigidity or elasticity of the contour. If it is omitted, there can be discontinuities along the contour. Removing it can also cause the snaxels to "bunch up."

$E_{smoothness} = ||d^2v/ds^2||^2$ measures the degree of bending or stiffness of the contour. Omitting it allows tangent discontinuities along the contour. This may cause the snake to become quite jagged and may overfit noisy data.

E_{image} measures features in the image that act as either attraction or repulsion forces on the contour. For example, using edge magnitude as a term makes the contour attracted to edges in the image. Omitting this term means that the evolution of the snake will not be influenced by the image data at all; with usual definitions of the total energy functional this will mean that the contour will shrink to a point or a line.

(b) [3] What is the effect of giving a negative weight to $E_{continuity}$?

This will cause snaxels to bunch up and tend to cause the contour to expand.

(c) [3] How could the energy functional definition be specified so as to cause the contour to expand?

One approach would be to modify E_{image} so that it also includes a "balloon" force in the normal direction of the contour at each snaxel point.

A second approach, given in the online reading, is to run the snake algorithm, then segment the resulting contour to eliminate high energy segments. Grow each end of each segment in the direction of its tangents; then run the snake algorithm on each segment. Finally, merge all of the final segments.

[10] **Segmentation using Normalized Graph Cut**

(a) [3] Define cut(A,B) and explain intuitively what it measures.

$cut(A,B) = \sum_{i \in A, j \in B} aff(i,j)$ measures the similarity between two groups of pixels defined by the disjoint sets A and B.

(b) [5] Say we want to find "regions" corresponding to object boundaries by grouping connected chains of "strong" edge points that are adjacent and their gradients imply a smooth contour. Define an affinity measure for this purpose that uses the magnitude, mag($\nabla I(p)$), and direction, direc($\nabla I(p)$), of the gradient, ∇I, at a pixel p in image I to compute $aff_{edge}(x,y)$ for a pair of adjacent pixels x and y. Include any additional computational steps before computing the affinity that make sense for obtaining good results. Briefly explain your definition.

Intuitively, if there is an edge between pixels x and y, then x and y are on opposite sides of the edge and the dissimilarity should be high, meaning the affinity should be low. On the other hand, if the edge directions at both x and y are nearly the same, then x and y are likely on the same contour and the affinity should be high. Thus affinity can be defined by measuring the angle between the gradient directions at the two pixels, weighted by the gradient magnitude. If the angle is near 180 degrees the affinity is low and if it is near 0 degrees, then the affinity is high. One way to do this is to compute the dot product of the two gradient vectors, $\nabla I(x) \cdot \nabla I(y)$, which computes the cosine of the angle between these vectors. Non-maxima suppression should be done first so that "thick" contours are not created.

(c) [2] Say we want to segment images into regions of uniform texture using a measure of the texture at each pixel by computing some function over a $(2n+1) \times (2n+1)$ neighborhood centered on the pixel. Briefly explain why using this function to define the texture affinity between two pixels, $aff_{texture}(x,y)$, will be biased depending on how close the pixels are to a region boundary.

Using a fixed window to compute the texture at a pixel will be "polluted" when the pixel is near a texture boundary because some of the pixels in the window will be in one region and other pixels will be in a different region, each with their own texture characteristics.

[20] **Edge Detection**

(a) [5] Show how an approximation to the first derivative of an image can be obtained by convolving the image with the kernel [1 −1] where the image is defined as

[56 64 79 98 115 126 132 133]

Ignore computing a value for the first and last image pixels (in other words, your result will be 6 values). In addition to showing the result of the convolution, indicate where edges would be detected and why.

8 15 19 17 11 6 1

Intensity discontinuities occur at the maxima of the first derivative, which this approximates. Here the maximum of 19 corresponds to the position where an edge is detected, corresponding to a position between the pixels with intensities 79 and 98 (or associated with one of these two pixels).

(b) [6] What property of the coefficients of a kernel ensures that an appropriate output is obtained for regions of constant intensity in an image when

(i) The kernel is approximating a first derivative.

The kernel values sum to 0.

(ii) The kernel is approximating a second derivative.

The kernel values sum to 0.

(iii) The kernel is approximating a Gaussian.

The kernel values sum to 1 (after normalization).

(c) [4] Describe a major advantage or disadvantage of using an isotropic operator instead of a non-isotropic operator with respect to the following issues:

(i) Computational efficiency.

Isotropic operators such as the Laplacian of a Gaussian are generally more computationally efficient than non-isotropic operators because they can be implemented using a single convolution (or multiplication in the frequency domain) followed by zero-crossing detection. Directional operators such as the Canny operator require a search for a local maxima.

(ii) Noise in the image.

Directional operators are generally more robust to noise by smoothing pixels which are on only one side of an edge. Also, an operator which uses higher order derivatives is more sensitive to noise.

(d) [5] What is the purpose of (i) non-maxima suppression and of (ii) hysteresis that are done in the Canny edge detector?

(i) Non-maxima suppression thins responses so that only a single pixel in the gradient direction will be detected, thus produces a one-pixel wide sequence of edge points.
(ii) Hysteresis thresholding simultaneously attempts to eliminate weak edge points that are caused by noise while filling in gaps between strong edge points where only weak edge response is detected.

2. [14] **Camera Calibration**

(a) [5] Show how the projection of a point in a planar scene at world coordinates (X,Y) to pixel coordinates (u,v) in an image plane can be represented using a *planar affine camera model*.

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ 0 & 0 & p_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

or, equivalently since the overall scale of the above matrix does not matter,

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

(b) [3] Under what conditions is the use of an affine transformation appropriate when viewing a planar scene?

If the field of view of the scene plane is such that all points visible on the world plane are at approximately the same depth from the camera compared to the distance of the camera from the plane.

(c) [3] How many degrees of freedom are there to solve for in (a), and what is the minimum number of calibration points needed to estimate the calibration parameters?

There are 6 degrees of freedom since the overall scale does not matter. Therefore, we need at least 3 points.

(d) [3] What effects can a planar affine transformation have on parallel lines?

Planar affine transformations preserve parallelism.

3. [12] **Edge Detection**

Compare the Canny edge detector and the Laplacian-of-Gaussian (LoG) edge detector for each of the following questions.

(a) [3] Which of these operators is/are isotropic and which is/are non-isotropic?

LoG is isotropic and Canny is non-isotropic.

(b) [3] Describe each operator in terms of the order of the derivatives that it computes.

LoG is 2nd derivative and Canny is 1st derivative.

(c) [3] What parameters must be defined by the user for each operator?

LoG requires σ defining the scale of the Gaussian blurring. Canny requires σ and two thresholds for hysteresis.

(e) [3] Which detector is more likely to produce long, thin contours? Briefly explain.

Canny because of non-maximum suppression which thins and hyteresis thresholding which can fill in weak edge gaps.

[13] **Feature Detection and Description**

(a) [5] We want a method for *corner detection* for use with 3D images, i.e., there is an intensity value for each (x,y,z) voxel. Describe a generalization of either the Harris corner detector or the Tomasi-Kanade corner detector by giving the main steps of an algorithm, including a test to decide when a voxel is a corner point.

Similarly to the 2D case, we can estimate the average change in intensity for a shift (u,v,w) around a given voxel using a bilinear approximation given by

$$E(u,v,w) = [u\ v\ w]\ M \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

where M is a 3 x 3 matrix computed from partial derivatives in the 3 directions. Compute the 3 eigenvalues, λ_1, λ_2 and λ_3, of M, specifying an ellipsoid at the voxel that measures the variation in intensity in 3 orthogonal directions. Using the test in the Tomasi-Kanade detector, mark the voxel as a corner point if the smallest eigenvalue, λ_3, is greater than a threshold. Using the test in the Harris operator, mark the voxel as a corner point if $\lambda_1\lambda_2\lambda_3 - k(\lambda_1 + \lambda_2 + \lambda_2)$ is greater than a threshold.

(b) [8] The *SIFT descriptor* is a popular method for describing selected feature points based on local neighborhood properties so that they can be matched reliably across images. Assuming feature points have been previously detected using the SIFT feature detector, (i) briefly describe the main steps of creating the SIFT feature *descriptor* at a given feature point, and (ii) name three (3) scene or image changes that the SIFT descriptor is invariant to (i.e., relatively insensitive to).

(i) At each point where a SIFT "keypoint" is detected, the descriptor is constructed by computing a set of 16 orientation histograms based on 4 x 4 windows within a 16 x 16 pixel neighborhood centered around the keypoint. At each pixel in the neighborhood, the gradient direction (quantized to 8 directions) is computed using a Gaussian with σ equal to 0.5 times the scale of the keypoint. The orientation histograms are computed relative to the orientation at the keypoint, with values weighted by the gradient magnitude of each pixel in the window. This results in a vector of 128 (= 16 x 8) feature values in the SIFT descriptor. (The values in the vector are also normalized to enhance invariance to illumination changes.)

(ii) Because the SIFT descriptor is based on edge orientation histograms, which are robust to contrast and brightness changes and are detected at different scales, the descriptor is translation, rotation, scale, and illumination (both intensity change by adding a constant and intensity change by contrast stretching) invariant. It is not invariant to significant viewpoint changes.