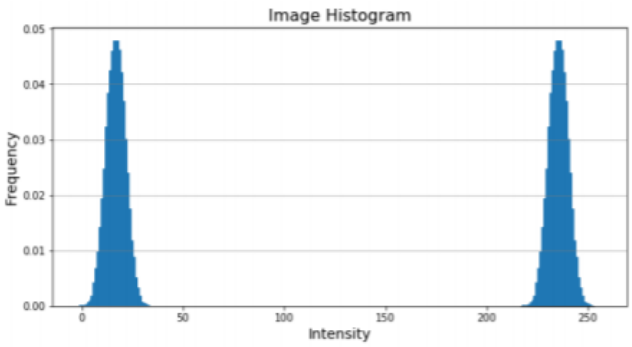
MID-TERM EXAMINATION - COMPUTER VISION (CS-GY 6643)

Pruthviraj R Patil, Computer Science, MS.
New York University, Tandon School of Engineering
Email: prp7650@nyu.edu

Question 1. Image Intensity

Assuming the following image intensity histogram:



- Sketch the corresponding cumulative distribution function.
- Sketch two possible images which could have generated this intensity histogram.

Solutions:

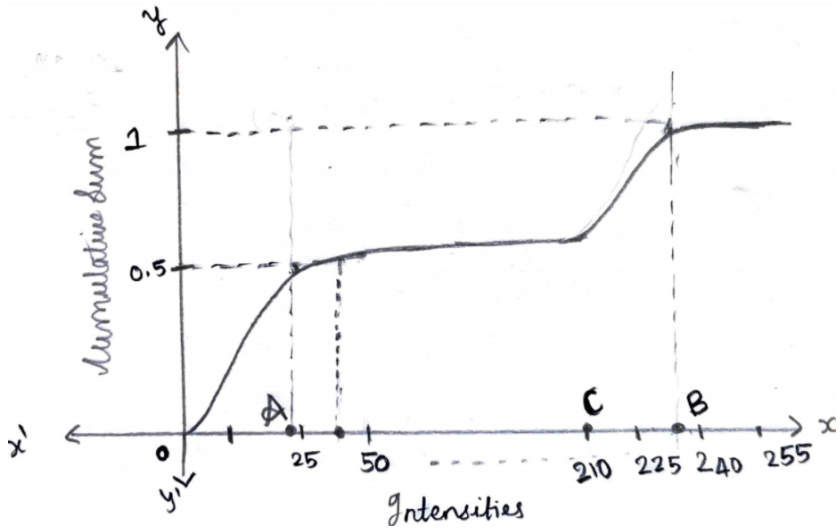


Figure 1.1 CDF of the given PDF

Explanation:

The cumulative sum of frequencies being on the y-axis can at max have the value 1. From the given PDF graph, it can be seen that there are two bell-shaped curves. Hence, the first curve starts from 0 and ends approximately at the 40th intensity bin, peaking around 20. That is why in the CDF image here, there is a sudden escalation of the cumulative sum from the 0th bin till

around 20th bin (A as marked in the picture) and after that, the curve sees a negligible increase in cumulative sum values. Similarly, at the position C, (i.e. around 200th bin), another bell curve starts at the corresponding location in the given PDF. That is why in the CDF too, a sudden escalation in cumulative sum occurs from C to B(around 230).

The two possible images to get this distribution are

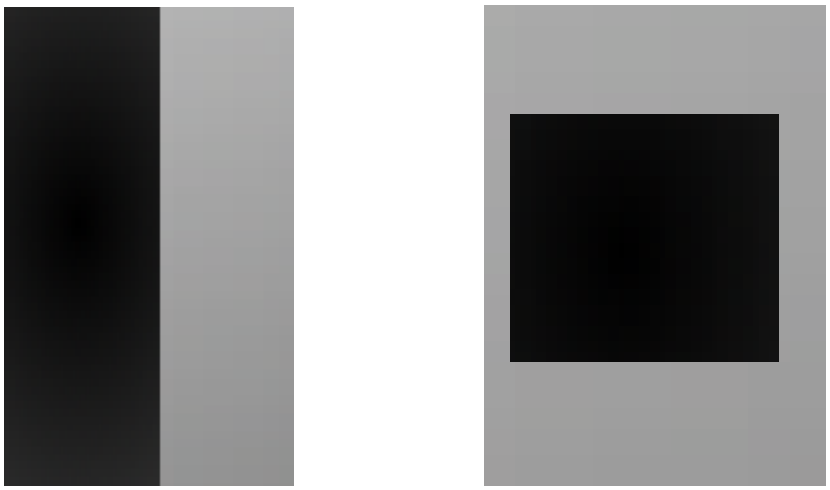


Figure 1.2. Two possible images to output the given pdf

Explanation:

The images must contain the pixel value in the range of (0 to 45) and (200 to 210 to 255). This can be found in either of the above images. In these images there are two major partitions in which there are pixels form a bell shaped curve when transformed to their corresponding pdfs.

Question 2. Filtering

Part A.

- Explain the difference between a box filter and a Gaussian filter.

Solution:

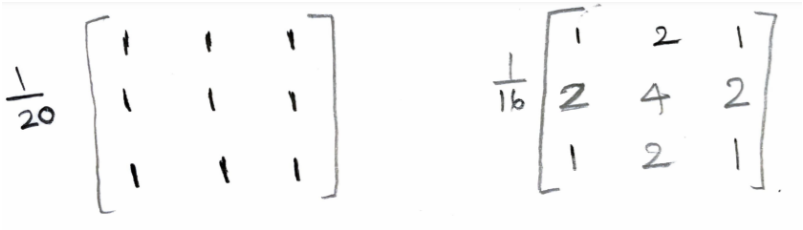


Figure 2.1 Box filter(Mean filter) and Normalized Gaussian filter kernels

According to the figure 2.1, in the Gaussian filter, the center pixel has the largest value and values on its neighboring points are reduced. Hence, the corresponding pixels on image patches that are near to the center in case of the Gaussian filter, influence more on smoothening rather than the ones in the distant space. This is because the Gaussian filter tends to measure the pixels in the bell curve around the most centered pixel so that the distant pixels get lower weightage. Whereas, in the case of a box filter, equal weightage is given to every individual component of the patch of the image over which the kernel is moved on. This is just an averaging over all the neighboring pixel values of the target pixel. This is done regardless what distance the neighboring pixels are placed at from the center pixel.

But, one of the advantages of box-filters smoothing is that they can be calculated easily and at a faster rate than that of the Gaussian smoothing process.

Part B.

- Write one property of median filtering. Don't describe how median filtering works, but rather give one property of the output of median filtering

Solution:

The filtered output image is obtained by placing the median of the kernel (image patch) values with the center of that image patch. One of the main advantages of the median filtering is that it avoids local biases especially in the case of salt and pepper noise. Hence, it is

unprecedentedly apt to use a Median filter in the case of such noise presence in the image to preprocess it.

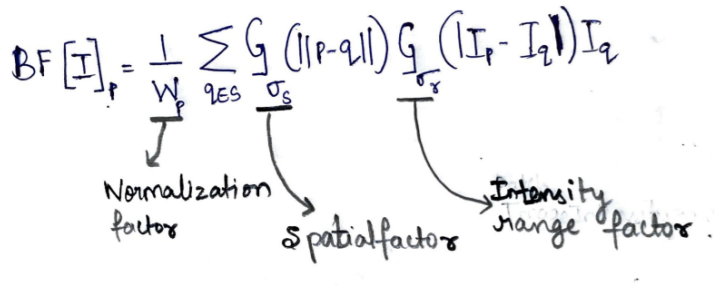
Another property of it's output is that: if the output depends on the size of the window chosen. That is why if we run the median filtering again using the same kernel size, unlike the rest of the filters' outputs, this output remains unchanged. However, in the presence of short-tailed noises, median filters sometimes tend to give out false noise edges.

Part C.

• In your own words, explain the concept of bilateral filtering.
• Explain the two terms *Gσs* and *Gσr* and describe intuitively what it means to multiply the two terms together.
• Describe and explain the output of bilateral filtering with a very large (infinite) *Gσr* .

Solution:

In the case of Gaussian filtering, only spatial factors of the images are considered. That is, the mean and the variance of the kernel points from its center is specified inorder to consider the centermost element of the image patch in the smoothing process. But, there is no usage of the wide intensity range in that patch that also contributes significantly to that image patch. To overcome this problem, Bilateral filtering uses both spatial and intensity range into consideration to apply filters over the image patch. The equation of the Bilateral filtering is given below:



Here, σ*s* specifies the spatial extent of the kernel/size of the neighbourhood considered whereas σ*r* specifies the minimum threshold (amplitude) to consider if the edge is present. The purpose of multiplying these both specifies that it considers the pixels that fall exclusively in the spatial range as well as the intensity range considered. But, the process is costly to compute as it is a non-linear process. Moreover, the process has to be run over each and every pixel. Hence, proves to be costly and this tradeoff has to be done in-order to get a better output than Gaussian filtering.

In the case where *Gσr* tends to be infinite, the bilateral filter acts as same as Gaussian filtering.. This is because, here, the significance of the *Gσr* is nullified as it determines the

minimum amplitude for the edges to occur and we have set it to zero hence giving the intensity range factor no role to perform in this process. That is why the output is the same as that of the Gaussian filtering.

Question 3. Edge detection

- Explain how image derivatives are related to edges in the image.
- Consider the 1D image below. Sketch the first and second derivative and explain how each can be used to determine the location of edges in the original image f(x). How many edges are there?

Solution:

The derivatives of the image is nothing but successive subtraction of the pixels. This helps in detecting where the image pixel intensities vary the most. But, edges are the positions in the image that determine where the intensity values are changed to the most, separating them from the other object setting a boundary(edge). This is how the image derivatives are helpful in detecting the edges in the image. The first and the second derivatives of the image with respect to the 1-D image is shown below.

$$f'(x_i) = f(x_i + h) - f(x_i)$$
$$f''(x_i) = f''(x_i + h) - f'(x_i)$$

Where there is an edge, there is more deflection in the pixel intensity values. In the image given below, we can see that the first derivative of the given one-dimensional image, it can be inferred that wherever the bell-shaped curve occurs, that means that there is a possibility of edge at that position. There are a total of three such bell curves (two positive and one negative) in the corresponding figure below (figure 3.1). Hence, there are three edges present in that image.

Similarly, in the case of the second derivative of f, the points where the curve crosses zero or the x axis (towards the negative side) are the positions of the edge. This is how the derivatives of the image are helpful in detecting the edges of the image.

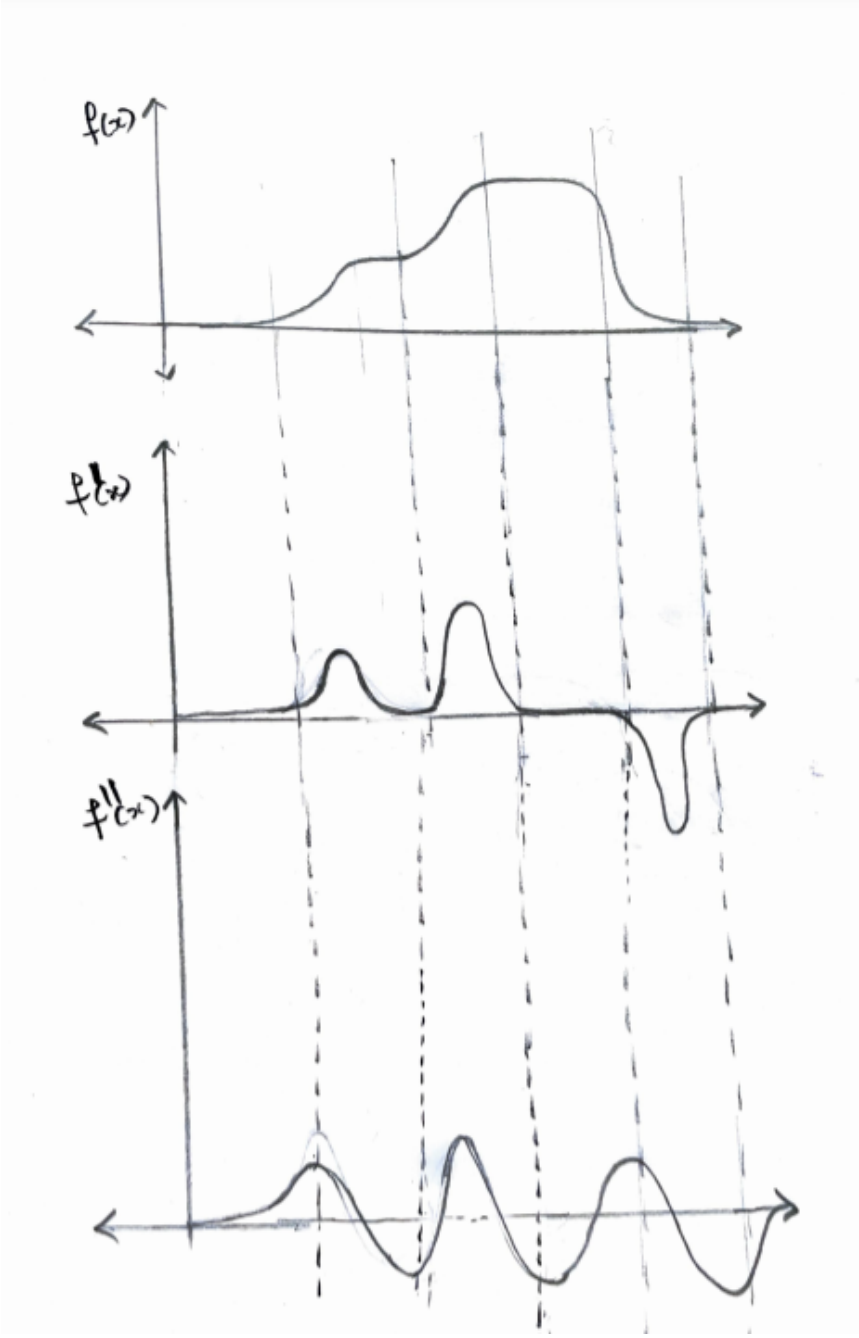
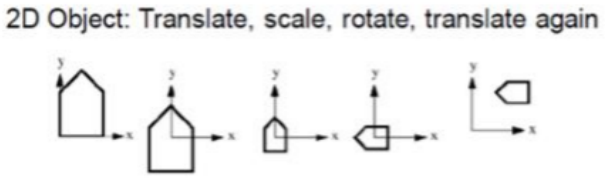


Figure 3.1. First and Second derivatives of the given 1-D line

Question 4. Homogeneous Coordinates:

Recall the following observation about a series of 2D transformations:



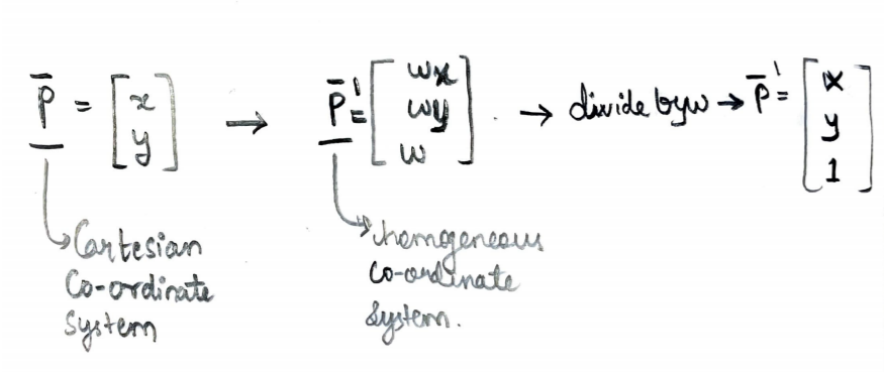
$$\vec{P'} = T2 + (R \cdot S \cdot (T1 + \vec{P}))$$

With this problem in mind, introduce the concept of homogeneous coordinates. What benefit does this representation provide? How does this change the way translation is represented?

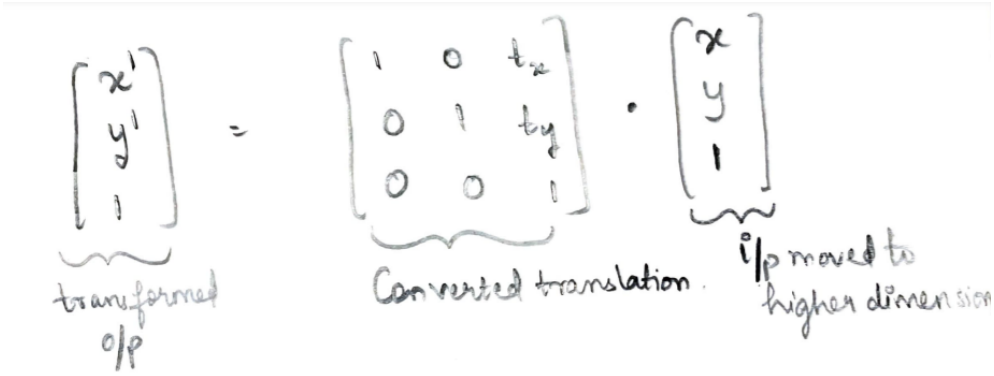
Solution:

In this problem, there are 4 consecutive transformations that have to be performed over the image P. But, the only issue is that the translation transformation is additive in nature and the rest are multiplicative in manner. This results in increased time complexity to perform the required set of transformations if they are known beforehand.

That is why, to overcome this limitation, we can harness the homogeneous coordinates system. In this, we add an additional dimension to the 'w' to the image vector. This is shown below:



After finding the transformation, we can divide the outcome by 'w' to get rid of the scaling factor and get the required pure cartesian points pitched to the higher dimension space.



Benefits by this process are as follows:

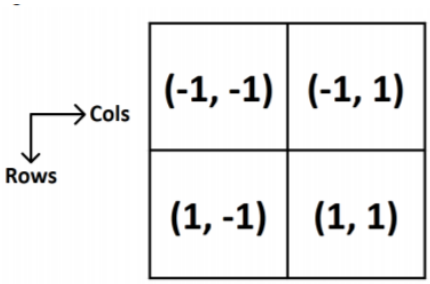
1. The process is scalable easily. For any given k dimensioned input, we have to transform the input image vector and the transformation matrix to k+1th dimension and continue the process.
2. The whole process takes place in a single step. This can be done by multiplying all the transformation matrices after moving them to the higher dimension and storing it beforehand. Later, the input image can directly be multiplied with this single matrix to undergo all the transformations at once. (we must know the order of the transformations we want to apply, else the results might vary).

Hence, homogeneous coordinates allow for an easy combination of multiple transformations by concatenating several matrix-vector multiplications. This change the way translation is represented to be: P' = T2.R.S.T1.P

Question 5. Image Transformations

Part A

- Consider the following known image f below, which is a 2x2 image with world coordinates associated with each pixel. These world coordinates represent a (rows, cols) coordinate system with the origin at the center of the image. Also consider the forward transformation to be scaling: Srows = 2.0 and Scols = 3.0.



- Compute the new world coordinates for all pixels in f under the forward transformation.
- Draw the output canvas g and denote clearly where the pixels of f land in g.
- Describe what you see and use that description to explain image transformation under the forward transformation.

Solution:

given co-ordinate points = (-1,-1), (-1,1), (1,-1), (1,1).

given $S_{rows} = 2.0$ $S_{Cols} = 3.0$

∴ Forward transformation: the new world co-ordinates are:

$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -2 \\ -3 \end{bmatrix}$, $\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}\cdot\begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$, $\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}\cdot\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \end{bmatrix}$

$= (-2,-3)$ $(-2,3)$ $(2,-3)$

$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}\cdot\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix} = (2,3)$

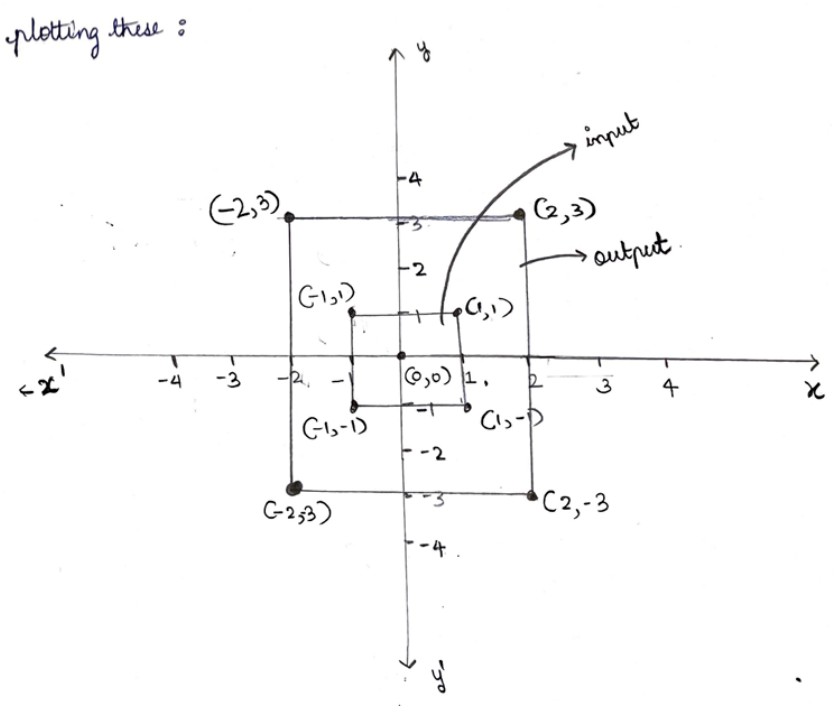


Figure 5.1: Canvas to illustrate the forward transformation

(Note: Here, Homogeneous coordinates aren't considered because there is only one type transformation to be carried on)

The image is scaled. The output image now has the pixels with centers (-2, -3), (2, 3), (2, -3), (-2, 3). This transformation was normal and didn't need any interpolations because this was an unit image patch and also the values in the transformation matrix were integers. But, we may face the problem of splatting in the case of inverse transformation.

Part B

- Write the backward/inverse transformation matrix which maps g to f.
- Verify it is the correct transformation by applying it to the transformed points you computed in part a), to make sure you get back the original world coordinates of f.
- Using arrows, sketch the location in f of three additional pixels in g under the backward transformation (does not have to be exact).
- Describe what you see and use that description to explain image transformation under the inverse transformation.

Solution:

The inverse transformation :

output $x' = S \cdot x$ ← input ⟶ Eqn ①

(S: scaling matrix)

$\therefore \quad S^{-1} x' = x.$ ⟶ Eqn ②

but, given $S = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$.

we know that, inverse of the diagonal matrix is obtained by taking reciprocals of its diagonal elements.

$\therefore \quad S^{-1} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix}$ is the inverse transformation matrix.

computed points in part a : $(-2,-3), (-2,3), (2,-3), (2,3)$

$\therefore$ Applying Eqn ② on these, we get :

$$\begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} -2 \\ -3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$\therefore \quad (x_1, x_2) = (-1, -1)$

$$\begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} -2 \\ 3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$(x_1, x_2) = (-1, 1)$

$$\begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} 2 \\ -3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$(x_1, x_2) = (1, -1)$

$$\begin{bmatrix} 1/2 & 0 \\ 0 & 1/3 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$(x_1, x_2) = (1, 1)$

$\therefore$ It is correct transformation as we are getting back our world co-ordinates back.

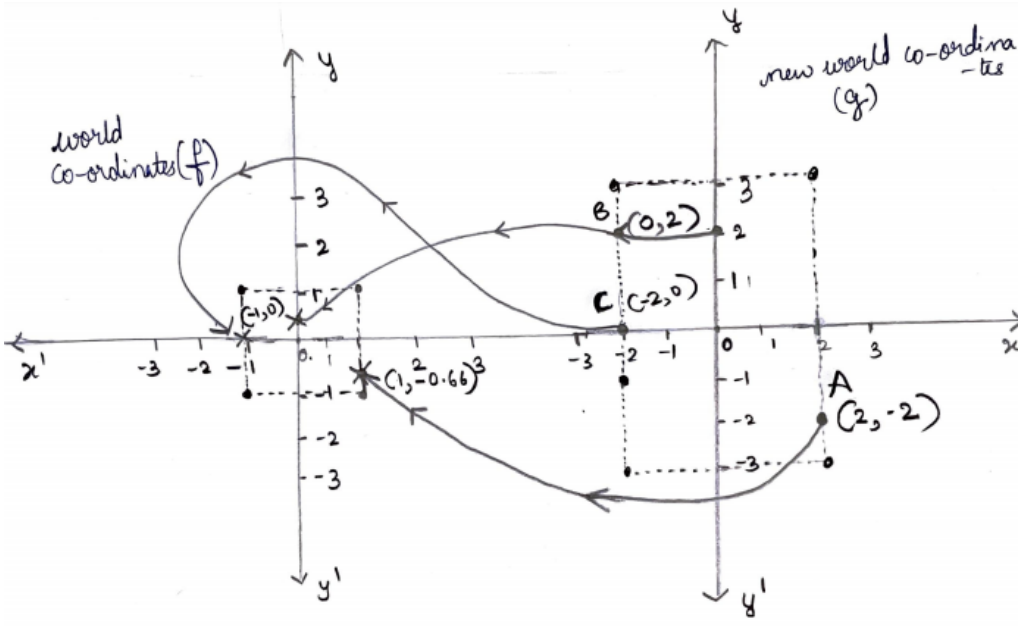
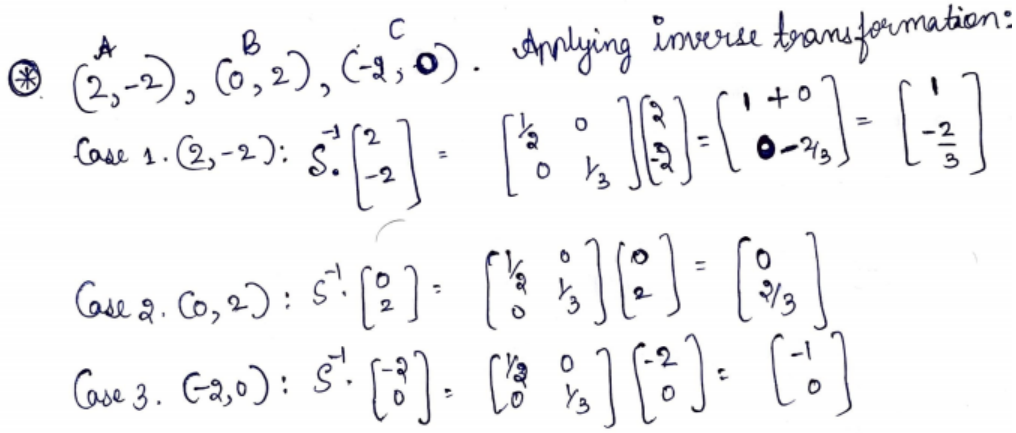


Figure 5.2. Figure to illustrate the location in f of three additional pixels in g under the backward transformation



Here, the chosen additional points are (2, -2), (0, 2), (-2, 0). After applying the inverse transformation, we get the points referred back to: (1, -0.66), (0, 0.66), (-1, 0). This leads us to use either the nearest neighbor interpolation or the Bilinear interpolation due to the splatting of the output transformed points after applying transformation.

Question 6: Interpolation

Part. A

- Explain the difference between nearest neighbor interpolation and bilinear interpolation.
- Explain why nearest neighbor interpolation would be preferable when transforming a binary image.

Solution:

In the case of Nearest Neighbor interpolation the pixel value, which is nearest to the transformed coordinate index when mapped back to the world coordinates is assigned to the corresponding position in the output space. But, in the case of Bilinear interpolation, the process allows the nearby pixel values to contribute to assigning the pixel value in the output space. This somewhat takes the weighted average of neighbor components.

However, in the case applying these transformations on a Binary image, the input image pixels have the value of either 0 or 1. Hence, no matter how good the weighted averaging is done in the case of Bilinear interpolation, the output when further binarized contains the values of either 0 or 1. Hence, it is preferable to use nearest neighbor interpolation over the Binary images.

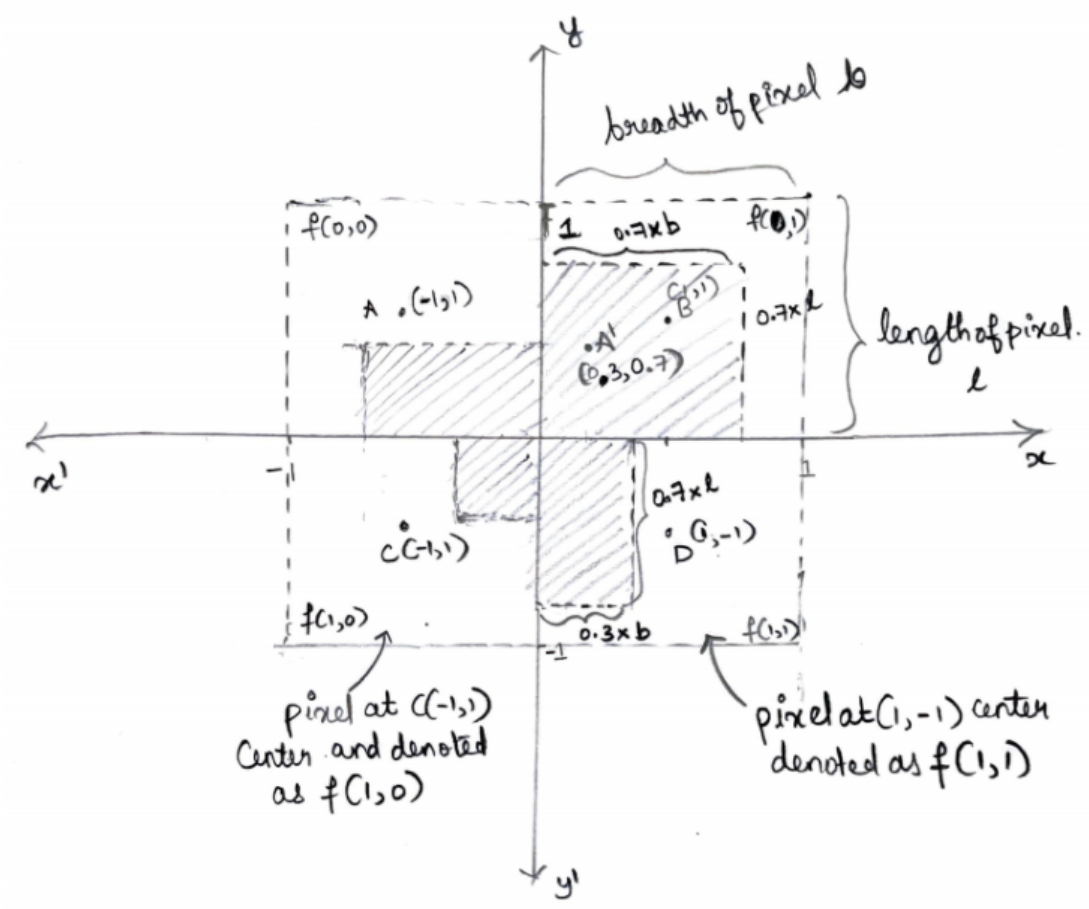
Part B

- Recall the formula for bilinear interpolation on the unit square: Consider the point (x,y) = (0.3, 0.7).

$$f(x,y) \approx f(0,0)\,(1-x)(1-y) + f(1,0)\,x(1-y) + f(0,1)\,(1-x)y + f(1,1)xy.$$

- Show graphically how the equation above could be interpreted as dividing the unit square into rectangular areas to be used as weights for interpolation.
- Describe how the area of the rectangles connect back to the equation. Do you notice an interesting pattern?

Solution:



Here, according to the equation given, if we put x and y values in it, we get:

$$f(x, y) = f(0,0) * 0.7 * 0.3 + f(0, 1) * 0.7 * 0.7 + f(1, 0) * 0.3 * 0.3 + f(1, 1) * 0.3 * 0.7$$

In this equation, 0.7, 0.3 determines how much of the **breadth and length** the rectangular patch from the pixel f(0, 0) must have . Similarly, the formula has weightage which determines what amount of breadth and length must be acquired from its **neighboring pixels**.

Question 7: Segmentation

Part A

- Explain the main difference between segmentation based on the Hough transform and based on deformable contours (snakes).
- When would you use Hough transform segmentation and when would you use deformable contour segmentation?

Solution

In the case of Hough transform, the objects are determined by using the parametric form. We need pick very convenient parameterization to foresee the family of curves that we are interested in and proceed to plot them. Hence, the given image maps on to the parametric space. This is done by discretizing the parameter space into bins and voting every point on the parameter space to have generated those particular points on the image plane. This is clearly shown in the figure 7.1. The bright point formed at the center of the image in the parametric space indicates the presence of the circle.

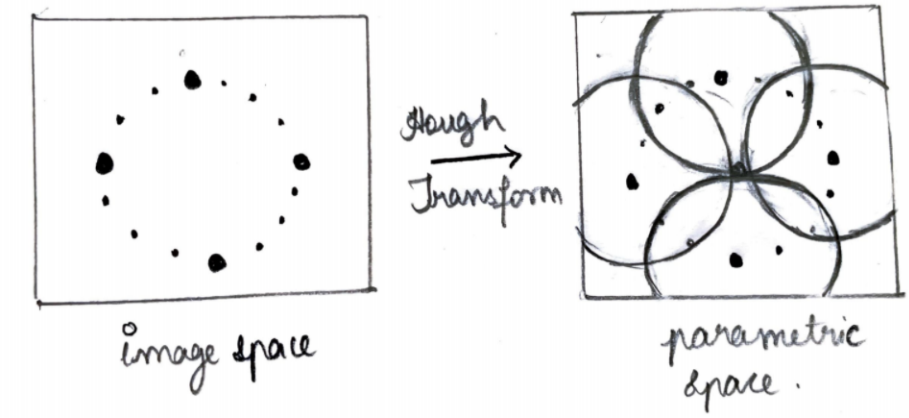


Figure: 7.1 Hough Transformation using parametric form of circles

But, in the case of Deformation contours, instead of the number of parameters for the required parametric curve to work over, we tend to **minimize the Energy function** (Cost function) that consists of two main components as shown below. I.e. E-Internal, E-External. E-internal determines the **continuity and smoothness** whereas the E-external determines gradient magnitude at that position. The weightages α, β, γ given to these components determine the amount of **spatial and intensity range** factors to be considered to deform the contour for the next epoch.

$$E_{Total} = \alpha E_{internal(continuity)} + \beta E_{internal(smoothness)} + \gamma E_{External}$$

equation 7.1

When to use them:

We can use Hough transformation if we are certain about the number of **parameters**, and if we can handle large amounts of **computation**. But, it is mainly used if there are **obvious types of edges** to be determined in the image (lines, circles etc).

Whereas, in the case of contour segmentation, even though there is a large possibility to **neglect the minute details** during the edge detection, the snake adjusts dynamically with respect to the given object boundaries according to its **energy components** initialized. Hence, if there is no need to give attention to detail, then we can use the latter methodology.

Moreover, if we use **dynamic image frames**, then it is apt to use deformable contour segmentation. This is because the snake can get adjusted by getting transformed to have the minimal energy.

Sometimes, in the case of Hough transformation, we can find the edges in the **obstructed image patches** (using threshold values) but cannot use contour segmentation (if the external energy component is less) in this case.

Part B

• In your own words, explain K-means clustering.
• Not including how to choose k, explain the main limitations of K-means clustering.
• How do these limitations naturally lead to the concept of Gaussian mixture models, and how do Gaussian mixture models address the limitation of K-means?
• Provide sketches and equations.

Solution:

K-Means clustering:

K-Means clustering, is one of the **unsupervised** learning methodologies that helps us to obtain information from the given set of points that aren't labelled. In this process, given n observations, the information is obtained by them by **partitioning them into k clusters** where every observation belongs to the cluster with the **nearest mean**. This is clearly visualized in figure 7.2.

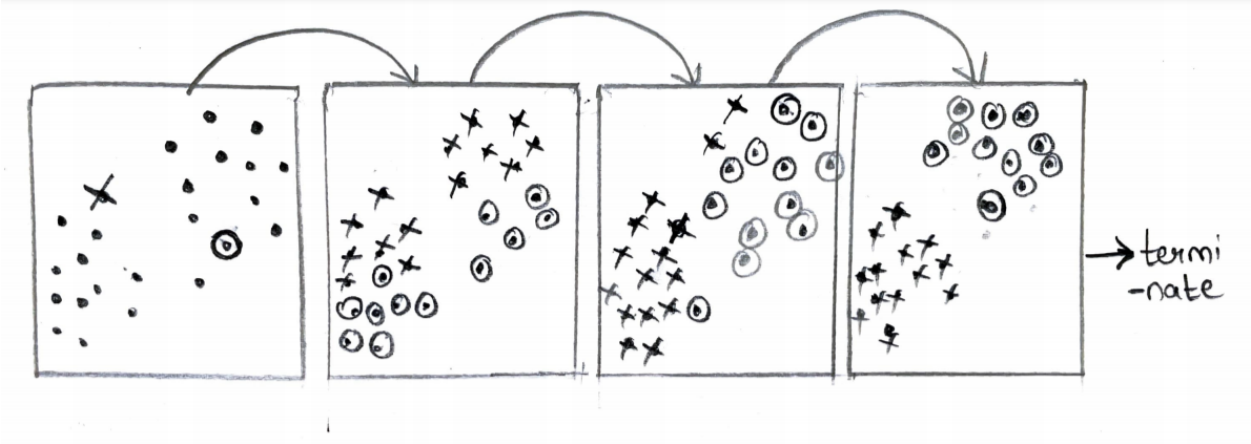
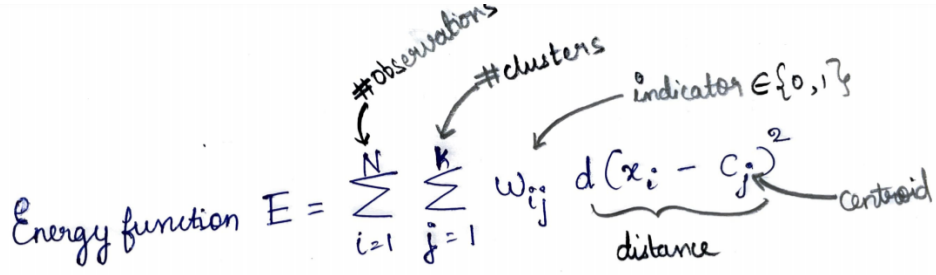


Figure 7.2. Pictorial representation of the K-means process.

Explanation of the process:

1. Firstly, there is a need for determining the number of **clusters (k)** we want the data to be divided into. Even though there are many procedures to choose the value k, the simplest one remains by the process of observing the data by plotting it.
2. Next, the **initial cluster centroids** (k in number) for every cluster have to be selected, that is: $C = \{c1, c2, ... ck\}$.
3. For every data point in $\{x1, x2, x3, ... Xn\}$, the **Euclidian distance** from each and every centroid has to be computed. Let these distances be $D = \{[d1, d2, d3, d4, ... dk], [d1, d2, d3, d4, ... dk], [d1, d2, d3, d4, ... dk], [d1, d2, d3, d4, ... dk], ... \}$(for every data point).
4. Find the **minimum distance** $d_{j\,min}$ from each of the individual distance array elements from the distance set D and assign the observations to their corresponding clusters whose centroids are at the nearest distance. This process of step 3 and 4 are done using the **energy function** shown below.



5. **Group the elements** according to their clusters and find their individual cluster's mean values. $\{m1, m2, m3, \ldots mk\}$
6. Consider these means as the **new centroids** and repeat step 2 to step 5 **until the groups remain unchanged** or for the desired number of epochs.
7. Output the cluster elements with labels as cluster numbers.

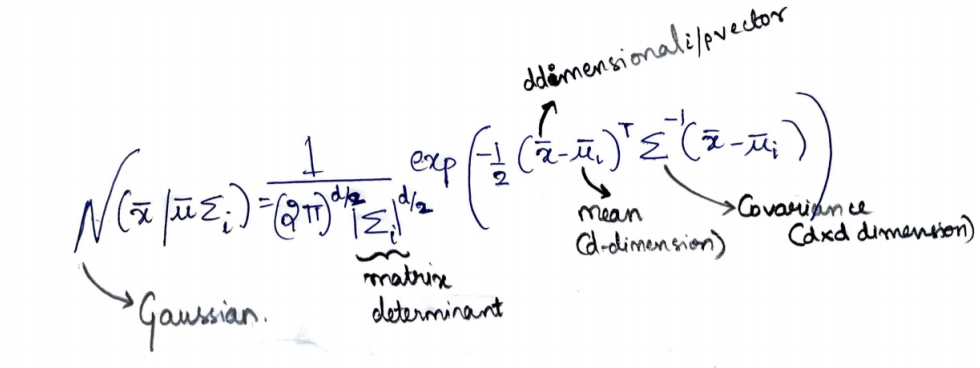
Limitations of K-Means Clustering:

1. Centroids' values can be **affected by the outliers**: This is because outliers, which do not fall into any of the groups are considered to be part of a group while calculating the mean distances. Hence, the distance from the centroid to its corresponding cluster element which is a centroid would be large enough to vary the centroid value (mean) in the next epoch anomalously.
2. The k-means clustering is **hard-clustering**: This means that there is no **notion for the probability** for the class assignments. The elements are assigned to be either present in the corresponding clusters/or not (0 or 1) (determined by the indicator variable 'w' in the energy function shown above).
3. K means clustering assumes that each and every cluster is of the **equal size and elements are placed with equal variance from centroid**. But, the clusters can differ in size in many real time situations.
4. If the data is dynamic, we might have to check the optimal k **periodically**.

All these limitations lead us to the process of Gaussian Mixture modelling. This is because:

In the Gaussian model, there are **no indicator variables** to say if the observation belongs to the cluster or not but the probabilities values array (**responsibility**) for every cluster k to determine the probability of the element to belong to that particular cluster. This is why we can classify the process of the Gaussian mixture modelling to be a **Soft-clustering model.**

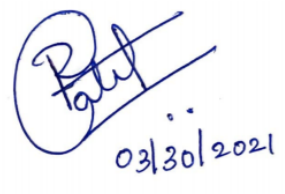
Every gaussian is given by the formula:



Hence, the mean and variance of every cluster **isn't considered to be the same** by the Gaussian mixture model. This proves to be more **robust** than K-Means in this case.

STUDENT HONOR PLEDGE

I pledge on my honor that I have not given or received any unauthorized assistance on this exam.



(Pruthviraj R Patil, N16324281)