Digital Image Processing

Dr. Mubashir Ahmad (Ph.D.)

Motion segmentation

- Introduction:
- Motion segmentation is a complex task in computer vision that involves separating moving objects from the background in a video sequence. Various mathematical techniques and algorithms are employed to achieve accurate and robust motion segmentation.
- Motion segmentation is the process of separating moving objects from the background in a video sequence.
- It plays a crucial role in various computer vision applications such as surveillance, autonomous driving, and human-computer interaction.

Background Subtraction

- Background subtraction methods assume that the background is static and subtract it from the current frame to extract moving objects. Let's denote the current frame as I(x, y, t), where (x, y) represents a pixel location, and t represents the time index.
- a. Background Modeling
- This is typically done by updating a background model over time. One common approach is to use a running average of pixel intensities:
- $B(x, y, t) = \alpha * B(x, y, t-1) + (1 - \alpha) * I(x, y, t)$

Background Modeling

- Here, B(x, y, t) represents the background model at time t, α is a decay factor controlling the rate of update, and I(x, y, t) is the current frame.
- The background model represents the long-term average of pixel values and adapts to gradual changes in the background.
- b) Foreground Detection:
- Once we have the background model, we can subtract it from the current frame to obtain the foreground mask. The foreground mask indicates the pixels that are likely to belong to moving objects:
- $M(x, y, t) = |I(x, y, t) - B(x, y, t)| > T$

Background Modeling

- Here, M(x, y, t) is the binary foreground mask, and T is a threshold value. If the absolute difference between the pixel value in the current frame and the corresponding background model exceeds the threshold, the pixel is considered part of the foreground.

Challenges in Motion Segmentation:

- Occlusion:
- Occlusion occurs when one object is partially or completely hidden by another object.
- It poses a significant challenge for motion segmentation algorithms as they need to distinguish between foreground and background objects accurately.
- Dynamic Backgrounds:
- If the background is not static, traditional background subtraction methods may struggle to accurately detect moving objects.
- Adapting the background model or incorporating temporal information can help address this challenge.

Background Subtraction

- Given an image (mostly likely to be a video frame), we want to identify the foreground objects in that image!

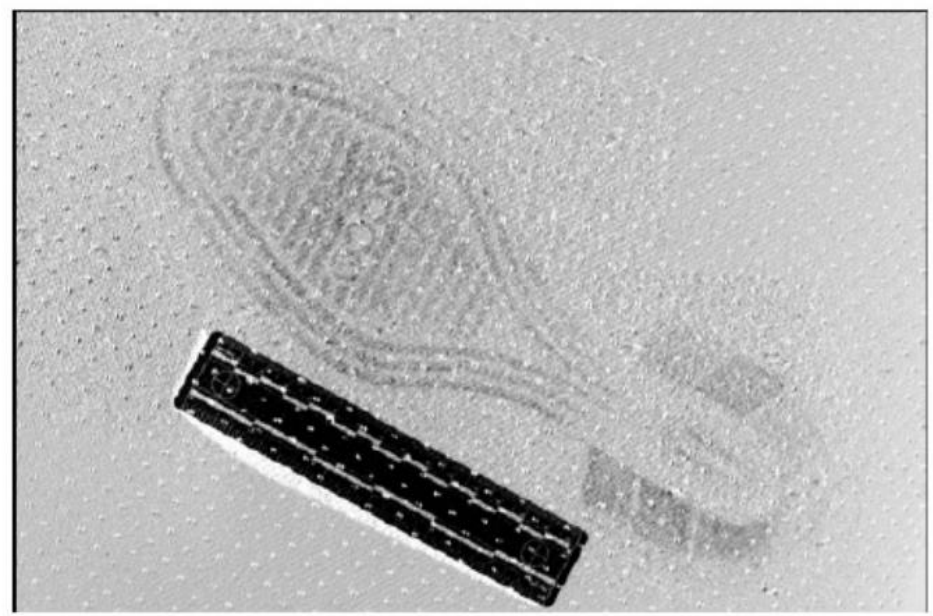
⇒

Motivation

- In most cases, objects are of interest, not the scene.
- Makes our life easier: less processing costs, and less room for error

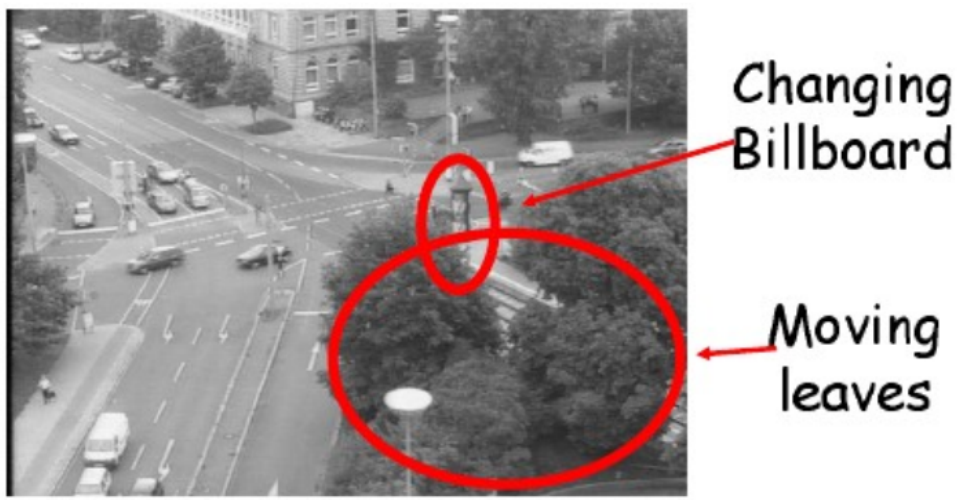
Widely Used!

- Traffic monitoring (counting vehicles, detecting & tracking vehicles),
- Human action recognition (run, walk, jump, squat, . . .),
- Human-computer interaction (“human interface”),
- Object tracking (watched tennis lately?!?),
- And in many other cool applications of computer vision such as digital forensics.



Requirements

- A reliable and robust background subtraction algorithm should handle:
  - Sudden or gradual illumination changes,
  - Long-term scene changes (a car is parked for a month).
  - high frequency, repetitive motion in the background (such as tree leaves, flags, waves, . . .)



Requirements

- ...continues
  - Secondary illumination effects (e.g. shadows cast by foreground objects)

Simple Approach

1. Estimate the background for time t.
2. Subtract the estimated background from the input frame.
3. Apply a threshold T to the absolute difference to get the foreground mask.

Image at time t

-

Background at time t

But, how can we estimate the background?

Frame Differencing

- Background is estimated to be the previous frame.
- Background subtraction equation then becomes:

$$B(x, y, t) = I(x, y, t - 1)$$

$$|I(x, y, t) - I(x, y, t - 1)| > T$$

- Depending on the object structure, speed, frame rate and global threshold, this approach may or may not be useful (usually not).

-

Frame Differencing

T=25

T=50

T=100

T=200

Mean Filter

- In this case the background is the mean of the previous n frames:

$$B(x,y,t) = \frac{1}{n}\sum_{i=0}^{n-1} I(x,y,t-i)$$

$$\left| I(x,y,t) - \frac{1}{n}\sum_{i=0}^{n-1} I(x,y,t-i) \right| > T$$

n=10

Estimated background

Estimated foreground

Mean Filter

n=20

Estimated background

Estimated foreground

n=50

Estimated background

Estimated foreground

Median Filter

- Assuming that the background is more likely to appear in a scene, we can use the median of the previous n frames as the background model:

$$B(x, y, t) = \underset{i=0\ldots n-1}{\text{median}} \left(I(x, y, t-i)\right)$$

$$\left|I(x, y, t) - \underset{i=0\ldots n-1}{\text{median}} \left(I(x, y, t-i)\right)\right| > T$$

n=10

Estimated background

Estimated foreground

Median Filter

n=20

Estimated background

Estimated foreground

n=50

Estimated background

Estimated foreground

Advantages vs. Shortcomings

- Advantages:
  - Extremely easy to implement and use!
  - All pretty fast.
  - Corresponding background models are not constant, they change over time.

- Disadvantages:
  - Accuracy of frame differencing depends on object speed and frame rate!
  - Mean and median background models have relatively high memory requirements.
    - In case of the mean background model, this can be handled by a running average

Advantages vs. Shortcomings

- There is another major problem with these simple approaches:

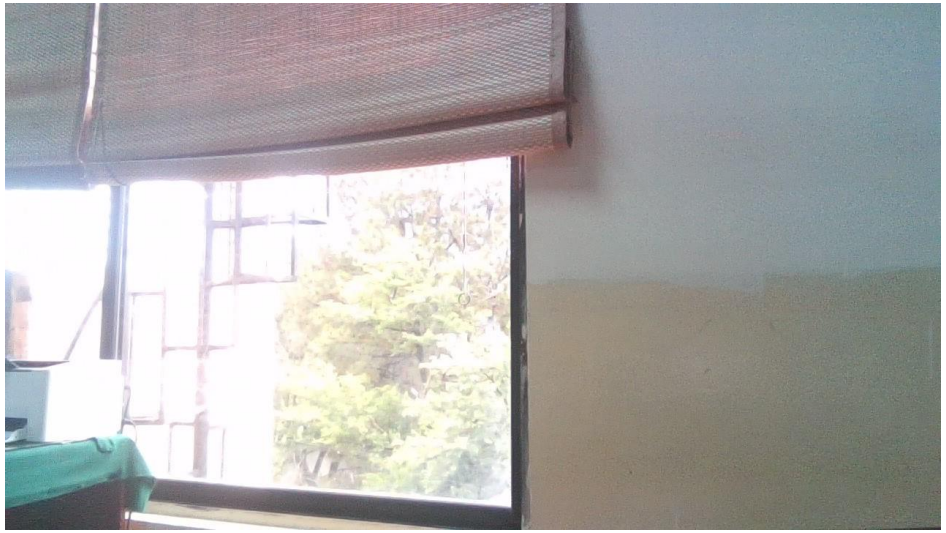
  1. There is one global threshold, Th, for all pixels in the image.

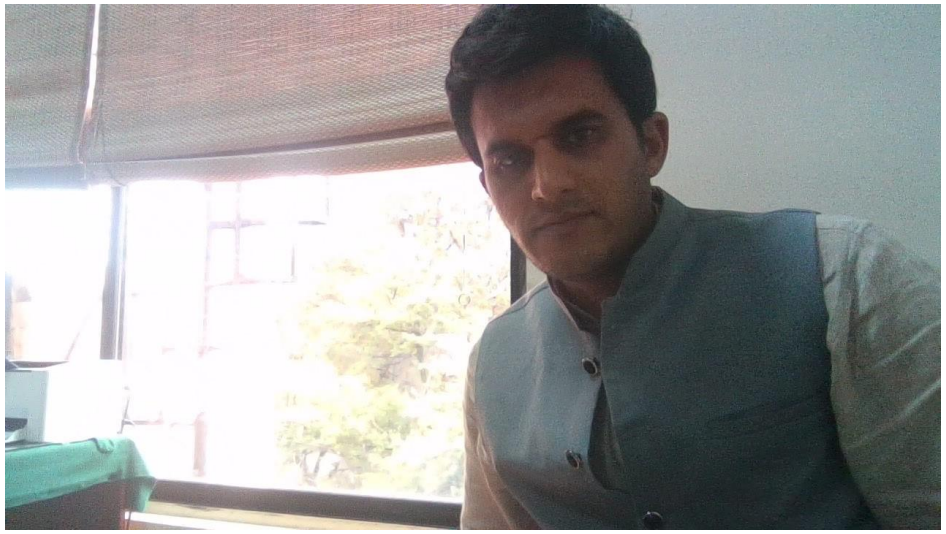
  2. And even a bigger problem:
  this threshold is not a function of t.

- So, these approaches will not give good results in the following conditions:
  - if the background is bimodal,
  - if the scene contains many, slowly moving objects (mean & median),
  - if the objects are fast and frame rate is slow (frame differencing),
  - and if general lighting conditions in the scene change with time!

Applications of Motion Segmentation:

- Surveillance Systems:
- Motion segmentation is crucial in video surveillance systems to detect and track objects of interest, such as intruders or suspicious activities.
- Autonomous Driving:
- Motion segmentation plays a vital role in autonomous vehicles by helping to identify and track pedestrians, vehicles, and other objects in the scene.
- Human-Computer Interaction:
- Motion segmentation enables gesture recognition and tracking for natural and intuitive human-computer interaction in applications like gaming and virtual reality.





Motion detection

- Motion segmentation is a fundamental task in computer vision that involves separating moving objects from the background in a video sequence.
- Various methods, such as background subtraction, optical flow, graph cut, and deep learning, have been used to tackle this problem.
- Overcoming challenges like occlusion and dynamic backgrounds is crucial for achieving accurate and robust motion segmentation.
- The applications of motion segmentation span a wide range of domains, including surveillance, autonomous driving, and human-computer interaction.

Shot Boundary detection

- Shot boundary detection is a crucial task in video processing and analysis that aims to identify transitions between shots or scenes in a video sequence. Shot boundaries can be classified into different types, such as cuts, fades, dissolves, wipes, and others. In this explanation, we will focus on the detection of cut shot boundaries, which are the most common and straightforward type.
- Pixel Intensity Difference:
- One of the simplest and widely used methods for cut shot boundary detection is based on the pixel intensity difference between consecutive frames.

Shot Boundary detection

- a) Frame Difference:
- The frame difference method calculates the absolute difference between corresponding pixels in two consecutive frames:
- $D(x, y, t) = |I(x, y, t) - I(x, y, t-1)|$
- Here, $D(x, y, t)$ represents the pixel intensity difference at location (x, y) and time index t, $I(x, y, t)$ represents the pixel value at (x, y) in the current frame, and $I(x, y, t-1)$ represents the pixel value at (x, y) in the previous frame.
- b) Thresholding: After calculating the frame difference, a thresholding step is applied to determine whether a shot boundary is present. If the intensity difference exceeds a predefined threshold, a shot boundary is detected:

Shot Boundary detection

- $M(x, y, t) = D(x, y, t) > T$
- Here, $M(x, y, t)$ represents the binary mask indicating the presence of a shot boundary, and T is the threshold value.
- c) Color Histogram Difference:
- In this approach, color histograms are computed for consecutive frames, and the difference between histograms is calculated:
- $H(t) = \text{ComputeHistogram}(I(t))$
- $H(t-1) = \text{ComputeHistogram}(I(t-1))$
- $D(t) = \text{CompareHistograms}(H(t), H(t-1))$

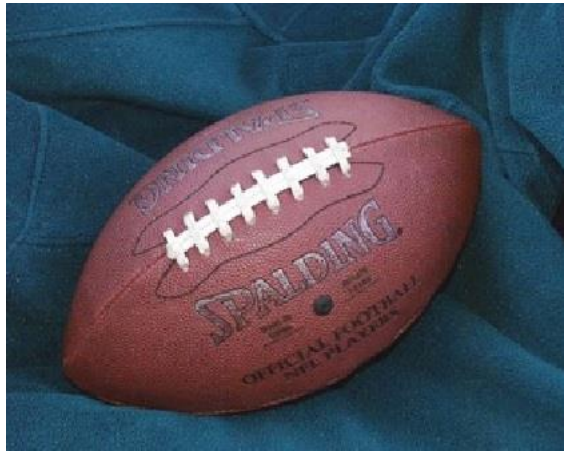
Region Oriented Segmentation

- Region Oriented Segmentation is a computer vision technique used to partition an image into meaningful regions or segments based on the similarity of their properties.
- The goal of region-oriented segmentation is to group pixels or regions with similar characteristics together and differentiate them from the surrounding areas.
- Region Growing: The most fundamental region-oriented segmentation algorithm.
- Start with a seed pixel or region and iteratively grow the region by adding neighboring pixels that meet certain similarity criteria.

Region Oriented Segmentation

1. Select a seed pixel or region as the starting point.
2. Define a similarity criterion based on pixel intensities, color, texture, or other image properties.
3. Add neighboring pixels that satisfy the similarity criterion to the region.
4. Repeat steps 2 and 3 until no more pixels can be added or a stopping condition is met.
5. Output the final segmented regions.

Region Growing setting theshold



Region Merging Algorithm (Combining Similar Regions)

- Another common approach in region-oriented segmentation.
- Start with an initial over-segmentation of the image and iteratively merge adjacent regions based on similarity criteria.

1. Generate an initial over-segmentation of the image, typically using techniques like superpixels or grid-based segmentation.
2. Define a similarity measure between neighboring regions.
3. Merge adjacent regions that satisfy the similarity measure.
4. Repeat steps 2 and 3 until no more regions can be merged or a stopping condition is met.
5. Output the final segmented regions.

Applications of Region-Oriented Segmentation

- Image and Video Segmentation:
  - Object recognition and tracking
  - Image editing and manipulation
  - Video analysis and understanding
- Medical Imaging:
  - Tumor detection and analysis
  - Brain segmentation and analysis
- Robotics and Autonomous Systems:
  - Environment perception and mapping
  - Object detection and grasping

Hough Transform

- The Hough Transform is a popular computer vision algorithm used for detecting simple shapes, such as lines and circles, in an image. It was invented by Paul Hough in 1962 and has since become an essential tool in image processing and computer vision applications.
- The Hough Transform works by converting image space (x, y) coordinates into parameter space (usually represented as Hough space) to detect shapes that can be represented by mathematical equations. For example, in the case of lines, each point (x, y) in the image space corresponds to a line in the parameter space.

Hough Transform

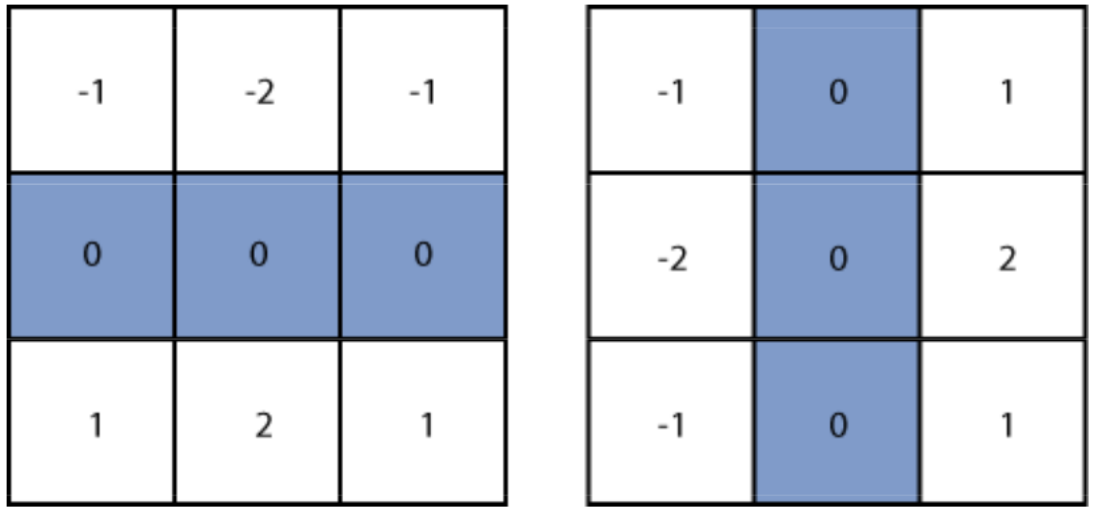
- The Hough transform is a technique used in image processing to detect shapes within an image, such as lines or circles. It works by creating a parameter space, where each point in the space represents a possible line or curve in the image. The image is then transformed into this parameter space, and the points that have the most votes, or "accumulation", are considered to be the most likely locations of the shapes. This technique is particularly useful for detecting shapes that are not easily represented in Cartesian coordinates, such as circles or ellipses.

Hough Transform

- Classical Hough Transform was concerned with the identification of Lines in images, but later it was extended to finding positions of arbitrary shapes, most commonly circles or ellipses.
  The simplest case of Hough Transform is detecting straight lines. In general straight lines are represented as $y = mx + c$.
- In most cases it is used for edge linking. Because noise in the image cause gapes

Hough-transform – the input

- The input image must be a thresholded edge image.
- The magnitude results computed by the Sobel operator can be thresholded and used as input operator can be thresholded and used as input.



Hough-transform – the input

- A thresholded edge image is the starting point for Hough transform.
- What does a Sobel filter produce?

- Approximation to the image gradient:

$$\nabla f(x)$$

- ...which is a vector quantity given by:

$$\nabla \mathbf{f}(x,y) = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

Hough-transform – the input

- The gradient is a measure of how the function f(x,y) changes as a function of changes in the arguments x and y.
- The gradient vector points in the direction of maximum change.
- The length of this vector indicates the size of the gradient:

$$\nabla f = |\nabla \mathbf{f}| = \sqrt{G_x^2 + G_y^2}$$

Hough-transform – the input

- Horisontal edges:
  - Compute $g_x(x,y)=H_x*f(x,y)$
  - Convolve with the horisontal filter kernel H_x
- Vertical edges:
  - Compute $g_t(x,y)=H_y*f(x,y)$
- Compute the gradient operator as:

$$g(x,y)=\sqrt{g_x^2(x,y)+g_y^2(x,y)}$$ Gradient-magnitude (kant-styrke)

$$\theta(x,y)=\tan^{-1}\left(\frac{g_y(x,y)}{g_x(x,y)}\right)$$ Gradient-retning

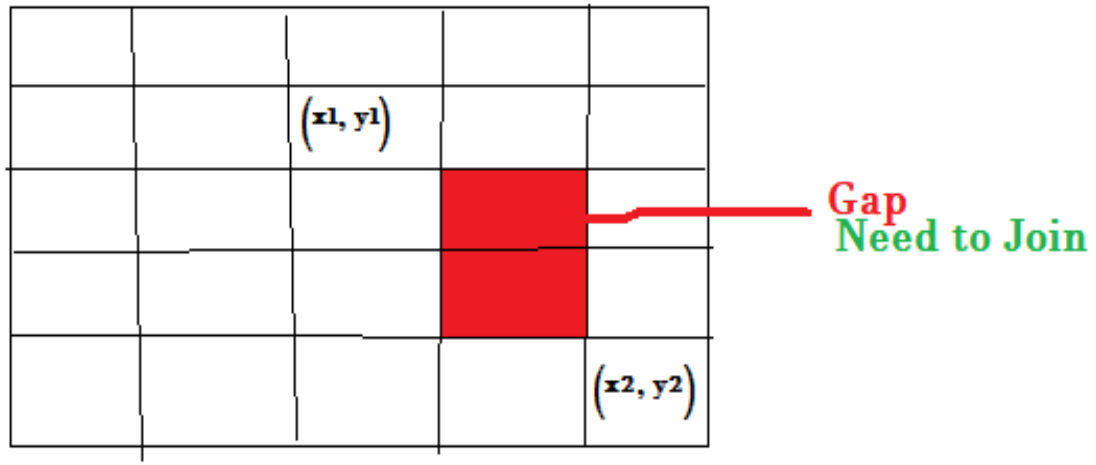
Hough-transform – the input

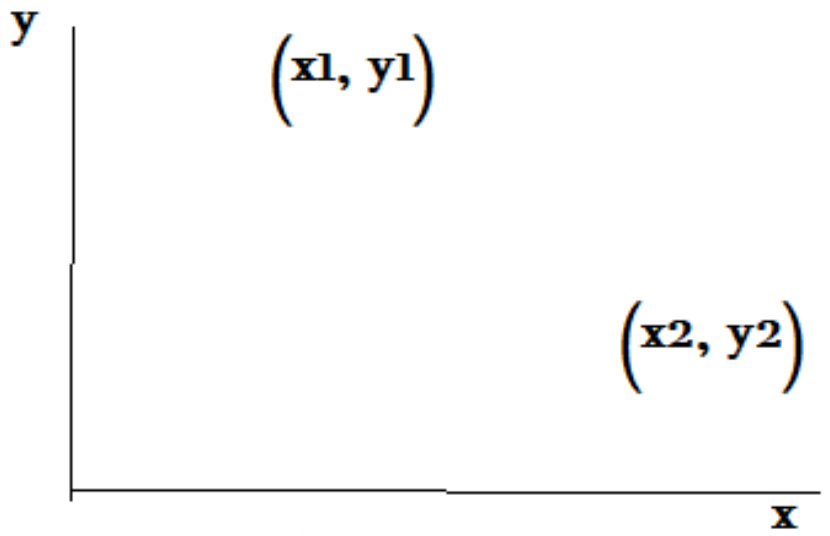
- The direction of this vector is also an important quantity.
- If α(x,y) is the direction of f in the point (x,y) then:

$$\alpha(x,y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

- Remember that α(x,y) will be the angle with respect to the x-axis
- Remember also that the direction of an edge will be perpendicular to the gradient in any given point

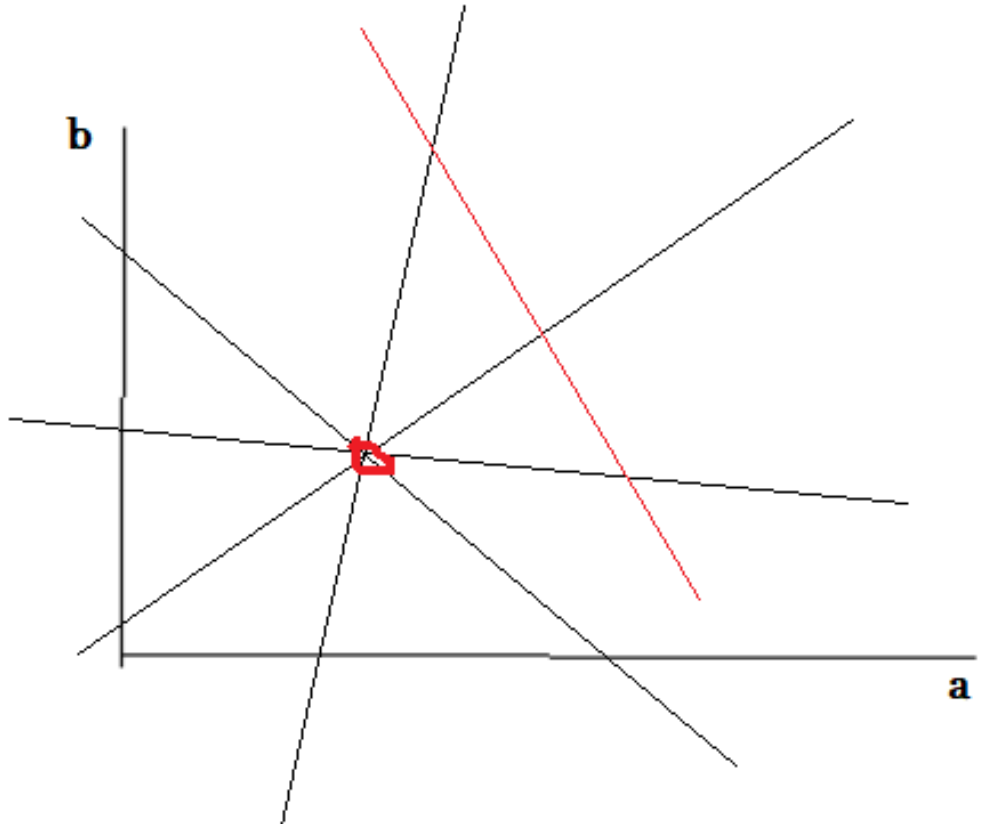
Hough-transform





Hough-transform

- y = mx + c
- Or y = ax + b
- b = -ax + y
- Now converted into a, b plan.
- The points in the x, y plan becomes a line in the a, b plan.



Hough-transform

- Given a set of points using Hough transform to join these points.
- A(1, 4), B(2, 3), C(3, 1), D(4, 1), E(5, 0). Find the required equation.
- These are values on x1 and y1 that pass through the line in the x, y plane.
- y = ax + b
- b = -ax + y

b = -a + 4

b = -2a +3

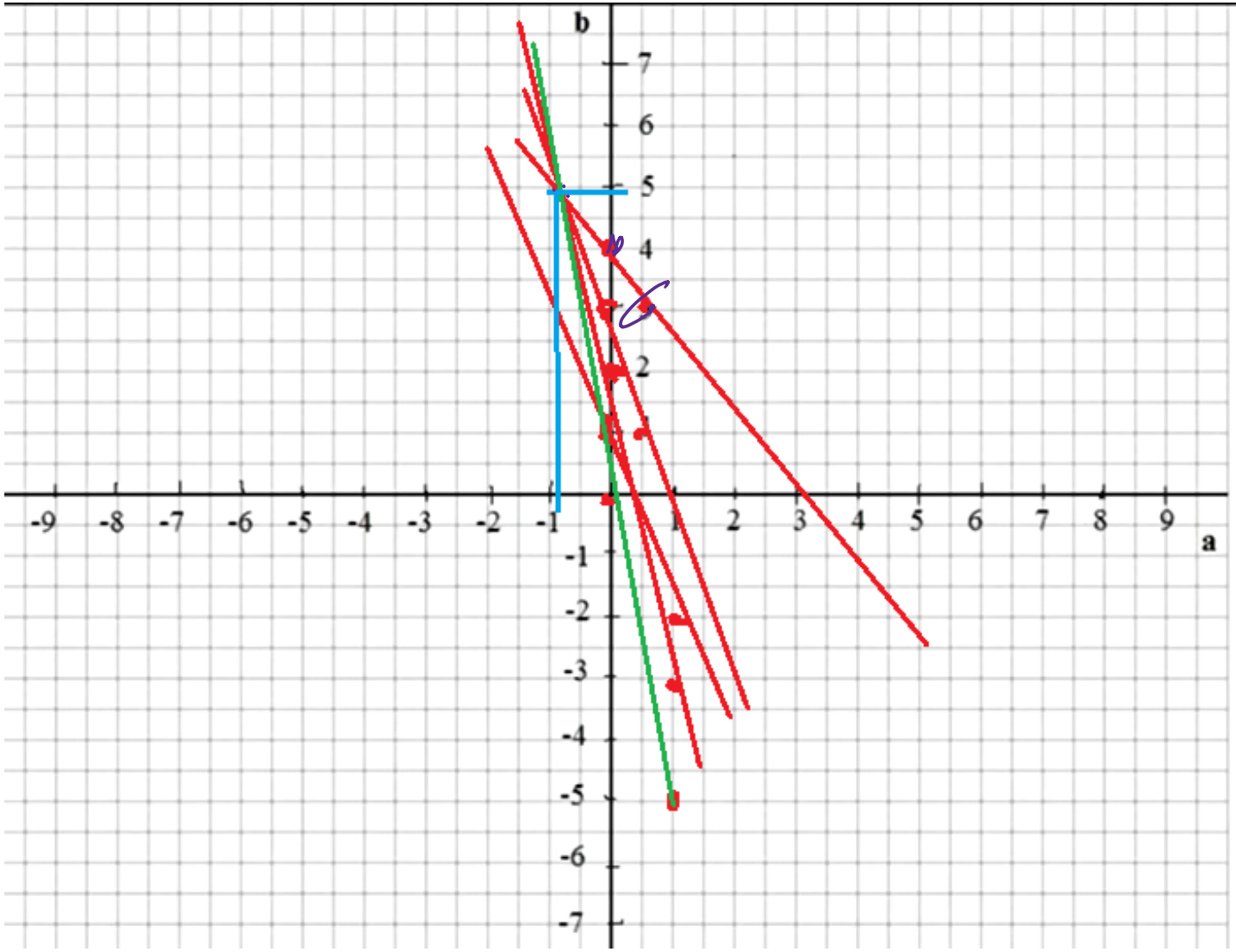
b = -3a +1

b = -4a + 1

b = -5a +0

Hough-transform

A(1, 4), B(2, 3), C(3, 1), D(4, 1), E(5, 0).



(0, 4), (1, 3)

$b = -a + 4$

(0, 3) (1, 1)

$b = -2a + 3$

(0, 1) (1, -2)

$b = -3a + 1$

(0, 1) (1, -3)

$b = -4a + 1$

(0,0) (1, -5)

$b = -5a + 0$

Put a = 0 and 1

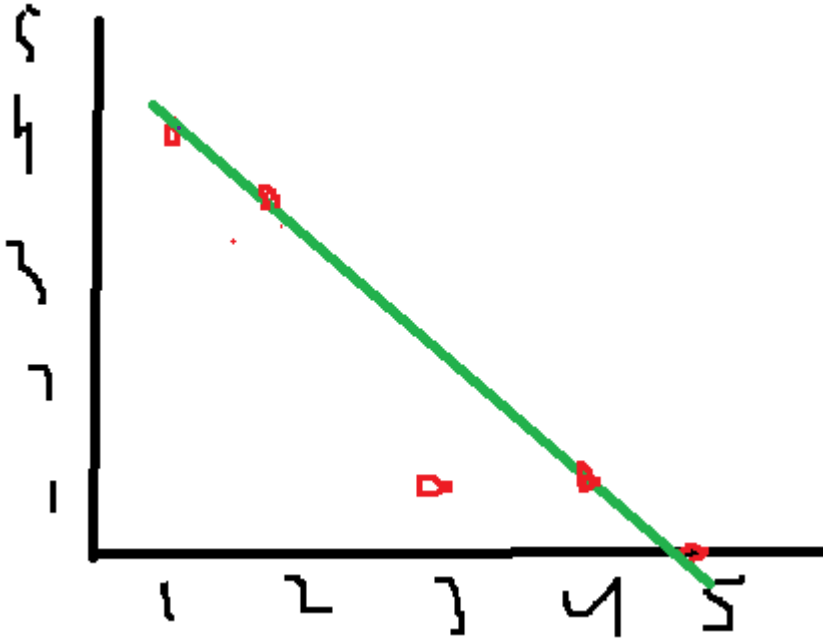
then draw a line

(-1, 5)

A(1, 4), B(2, 3), C(3, 1), D(4, 1), E(5, 0).

Hough-transform

- We got the values (-1, 5) which are a and b values.
- Put this in the equation y = ax + b
- Y = -1(x) + 5 => y = -x + 5 is the required equation that links all the edges.



Hough Transform results

Segmentation by clustering

- Segmentation by clustering is a technique used in data analysis and machine learning to divide a dataset into distinct groups or clusters based on the similarity of data points. It is an unsupervised learning approach, meaning that it does not rely on predefined labels or categories.
- The goal of clustering is to group data points together in such a way that points within the same cluster are similar to each other, while points in different clusters are dissimilar. Clustering can be applied to a wide range of domains, such as customer segmentation, image analysis, text mining, and anomaly detection.

Segmentation by clustering

- One popular clustering algorithm is K-means, which aims to partition the data into K clusters. The algorithm starts by randomly initializing K cluster centroids and then iteratively assigns data points to the nearest centroid, updating the centroid position based on the assigned points. This process continues until convergence, where the centroids no longer change significantly or a maximum number of iterations is reached.
- When applying clustering for segmentation, the resulting clusters can represent distinct groups or segments within the data. These segments can then be further analyzed or used for various purposes, such as targeted marketing, personalized recommendations, or anomaly detection.
- It's important to note that clustering is an exploratory technique, and the choice of algorithm and the number of clusters (K) often depends on the specific problem and domain knowledge. Evaluation metrics such as silhouette score or elbow method can be used to assess the quality of clustering results and determine the appropriate number of clusters.

K-mean Clustering (problem solution in next slide)