UNIT III

HOUGH TRANSFORM

Line Detection using Hough Transform

Introduction

Detecting straight lines in an image is one of the fundamental tasks in computer vision and image processing. Traditional edge detection operators like Sobel, Prewitt, or Canny can highlight edges, but they do not directly provide the equation of lines. The Hough Transform (HT) is a feature extraction technique used to detect simple shapes (lines, circles, ellipses) in images. It is especially powerful in detecting lines, even if they are:

Broken or discontinuous,

Hidden by noise,

Only partially visible.

Mathematical Representation of a Line

Cartesian Equation of a Line:

$y = mx + c$

m: slope of the line

c: intercept on the y-axis

This representation becomes unstable for vertical lines (slope $\rightarrow \infty$).

Polar (Hough) Representation of a Line:-To avoid issues with vertical lines, Hough Transform uses the polar form:

$\boldsymbol{\rho = x \cos \theta + y \sin \theta}$

rho ρ : perpendicular distance from origin to the line

θ :- angle between x-axis and perpendicular from origin

(x, y): coordinates of a point on the line

Concept of Hough Transform

Each edge point (x, y) in the image can correspond to many possible lines passing through it. For each possible angle **θ** we compute **ρ**. This creates a sinusoidal curve **ρ, θ** in parameter space called the Hough space. Where multiple curves intersect → corresponds to a line that passes through multiple edge points.

Algorithm for Line Detection using Hough Transform

Algorithm for Line Detection using Hough Transform:

Edge Detection:

- Apply an edge detection algorithm (e.g., Canny edge detector) to the input image to identify potential edge pixels. This step reduces the amount of data to process, as only edge points contribute to the Hough transform.

Parameterization of Lines:

- Instead of using the Cartesian equation $y = mx + c$, which has issues with vertical lines (infinite slope), the polar form $\rho = x \cos(\theta) + y \sin(\theta)$ is typically used. Here, ρ is the perpendicular distance from the origin to the line, and θ is the angle of the normal vector to the line.

Accumulator Array Initialization:

- Create a 2D accumulator array (Hough space) where the axes represent the parameters ρ and θ. The size of this array depends on the desired resolution for ρ and θ and the image dimensions. Initialize all cells in this array to zero.

Voting Process:

- Iterate through each detected edge pixel (x, y) in the image.
- For each edge pixel, consider a range of possible θ values (e.g., from 0 to 180 degrees).
- For each θ, calculate the corresponding ρ using the equation $\rho = x \cos(\theta) + y \sin(\theta)$.
- Increment the corresponding cell (ρ, θ) in the accumulator array. This "vote" indicates that a line with parameters (ρ, θ) could pass through the current edge pixel.

Peak Detection:

- After iterating through all edge pixels, search for local maxima (peaks) in the accumulator array. These peaks represent the most probable lines in the original image. A higher value in an accumulator cell indicates that more edge pixels lie on the line defined by that (ρ, θ) pair.

Line Extraction:

- Set a threshold for the accumulator values. Any (ρ, θ) pair with an accumulator value above this threshold is considered a detected line.
- Convert these (ρ, θ) parameters back to Cartesian coordinates to draw or represent the detected lines in the original image space.

Foot of Normal Method

In computer vision, the "foot of normal method" is a technique, similar to the standard polar coordinate parameterization used in the Hough Transform, that represents a line using the distance (ρ) and angle (θ) of the perpendicular from the origin to the line. While mathematically similar, it is sometimes used as an alternative to save computation by avoiding repeated arctangent calculations, though it can suffer from lower orientation accuracy for lines close to the origin due to its reliance on a discrete parameter space.

How it works

1. **Edge Detection:** First, the image is preprocessed to detect edges, typically using a method like the Canny edge detector.
2. **Parameterization:** For each edge point (x, y), the system calculates its corresponding (ρ, θ) pair.
3. **Hough Space:** Each point's (x, y) coordinates are mapped to a sinusoid in the Hough space, where the parameters (ρ, θ) are the coordinates.

4. **Accumulator Voting:** The Hough transform accumulates "votes" in the accumulator array.

5. **Line Detection:** Intersections of these sinusoids in the accumulator space represent collinear points, which are identified as lines.

Line Localization

Line localization in computer vision is the process of detecting and precisely identifying the position and orientation of lines within an image, often transforming from a 2D image to a 1D line representation. This is frequently achieved using techniques like the **Hough transform**, which converts lines into points in a parameter space, or through **deep learning models** that use convolutional neural networks to predict line properties. Line localization is crucial for tasks like **lane detection** in autonomous driving, **object recognition**, and 3D scene reconstruction.

How it works:

1. **Edge Detection:** The process often begins with detecting edges, which are transitions between colors or objects.

2. **Hough Transform:**

- This technique transforms the problem from finding lines in an image (spatial space) to finding points in a parameter space.

- A line in the image corresponds to a point in the Hough space.

- Points that are aligned in the image will intersect at a single point in the Hough space, indicating a detected line.

3. **Deep Learning Approaches:**

- Modern methods use Convolutional Neural Networks (CNNs) to predict straight lines and their characteristics, such as orientation and position.

- For example, a deep learning model might first identify the line in an image and then apply a Hough transform for more accurate detection, as seen in tree branch detection.

4. **Image Preprocessing:**

- Techniques like color filtering and gradient analysis are used to isolate potential line pixels from the background noise.

- A region of interest (ROI) or mask can be applied to focus on the specific area where lines are expected, such as the lane lines on a road.

Key Applications:

- **Autonomous Driving:** Detecting road lanes for self-driving vehicles.
- **Mobile Robotics:** Reconstructing and understanding 3D scenes made of straight lines.

- **Power Line Maintenance:** Identifying tree branches that need pruning to prevent power line interruptions.
- **Object Detection:** Identifying lines as features of objects to help in recognition and localization.

Line Fitting

Line fitting in computer vision is the process of identifying the parameters of a line that best explains a set of observed data points, often extracted from an image, by minimizing the errors between the points and the line. Common methods include **least-squares fitting**, which minimizes vertical distances but is sensitive to nearly vertical lines and outliers, and Hough Transform**-based methods**, which are more robust for detecting lines even with noise and gaps. Algorithms like RANSAC are used to robustly fit models to data containing outliers by randomly sampling subsets of data to find the best model.

How it works:

1. **Data Acquisition:** Edge points or other features are extracted from an image, resulting in a set of 2D coordinates (x_i, y_i).
2. **Model Selection:** A line model is chosen. Common representations are:

- **y = mx + c:** Minimizes vertical errors but is poor for near-vertical lines.
- **ax + by + c = 0 or Ax + By + C = 0:** A more general form that can handle vertical lines by minimizing perpendicular distances.

3. **Parameter Estimation:** Algorithms are used to find the line's parameters (like m and c, or A, B, C) that best fit the data points.

Common Techniques:

Least Squares:

- Principle: Minimizes the sum of squared errors, which can be the vertical distances (in the y = mx + c form) or perpendicular distances.
- **Matrix Form:** The system is often expressed in matrix form as Xb = y, where the parameters 'b' are estimated using the pseudo-inverse.
- **Limitations:** Sensitive to outliers, and minimizing vertical errors is problematic for near-vertical lines.

Hough Transform:

- **Principle:** A robust method that transforms the problem from fitting points to fitting lines in a parameter space. It is good at finding lines with gaps and noise by converting points into curves in the parameter space.

- **Process:** Points in the image are converted into sinusoidal curves in the Hough space; the peak in the Hough space corresponds to the line that best fits the points.

Random Sample Consensus (RANSAC):

- **Principle:** A robust method for fitting models to data with outliers.
- **Process:** It iteratively selects a random subset of points, fits a model to this subset, and then checks how many other points fit this model. The model with the most inliers is chosen.

Applications:

- **Lane Detection:** Identifying road lanes in autonomous driving.
- **Edge Detection:** Detecting boundaries of objects or structures in an image.
- **Vanishing Point Estimation:** Finding points where parallel lines appear to converge in an image.
- **Vectorization:** Converting edge-based representations into more structured formats.

RANSAC for straight line detection

RANSAC (Random Sample Consensus) is a powerful, iterative algorithm in computer vision for estimating a mathematical model from a dataset containing many outliers. It works by repeatedly selecting random subsets of data to fit a model, then counting how many data points support that model. The model with the largest "consensus" of supporting points (inliers), found after many iterations, is selected as the best fit, effectively ignoring the erroneous outlier points.

How RANSAC Works

1. **Randomly Select a Subset**: The algorithm randomly picks the minimum number of data points required to define a specific mathematical model (e.g., four points to define a homography in 2D image geometry).
2. **Fit a Model**: A model (like a line, plane, or transformation) is fitted to this minimal subset of data.
3. **Count Inliers**: All other data points are then tested against this candidate model to see if they fit within a certain error threshold. Points that fit are called "inliers".
4. **Repeat and Evaluate**: Steps 1-3 are repeated for a predetermined number of iterations.
5. **Select the Best Model**: The model that generated the largest number of inliers across all iterations is chosen as the final estimated model.
6. **Refine the Model**: Optionally, a final refinement step can be performed using all the identified inliers to re-calculate the model with greater accuracy.

Why RANSAC is Useful in Computer Vision

- **Robustness to Outliers**: Real-world data, such as images, often contains errors or "outliers". RANSAC is designed to effectively handle these outliers, providing a robust solution that traditional methods like least squares (which are sensitive to outliers) cannot.
- **Versatility**: It can be applied to a wide range of computer vision problems requiring robust parameter estimation.

Common Applications in Computer Vision

- Image Alignment: Estimating geometric transformations (like homographies) between two images for tasks such as image stitching or video stabilization.
- Stereo Vision: Estimating the fundamental matrix to find correspondences and depth between stereo images.
- Feature Matching: Finding common features between two images or frames and filtering out incorrect matches.
- Lane Detection: Finding the correct lane lines on a road surface for self-driving cars, even with shadows and other obstructions.

HT based circular object detection

Circular Hough Transform Implementation: Unlike the linear HT, the CHT relies on equations for circles. The equation of the a circle is,

$$r^2 = (x - a)^2 + (y - b)^2 \quad (3)$$

Here a and b represent the coordinates for the center, and r is the radius of the circle. The parametric representation of this circle is

$$x = a + r*\cos(\theta) \quad (4)$$

$$y = b + r*\sin(\theta) \quad (4)$$

In contrast to a linear HT, a CHT relies on 3 parameters, which requires a larger computation time and memory for storage, increasing the complexity of extracting information from our image. For simplicity, most CHT programs set the radius to a constant value (hard coded) or provide the user with the option of setting a range (maximum and minimum) prior to running the application. For each edge point, a circle is drawn with that point as origin and radius r. The CHT also uses an array (3D) with the first two dimensions representing the coordinates of the circle and the last third specifying the radii. The values in the accumulator (array) are increased every time a circle is drawn with the desired radii over every edge point. The accumulator, which kept counts of how many circles pass through coordinates of each edge point, proceeds to a vote to find the highest count. The coordinates of the center of the circles in the images are the coordinates with the highest count.

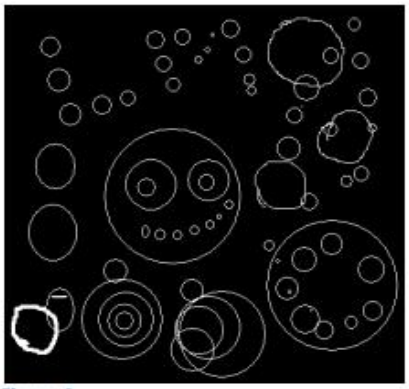

Figure 4

Input image – after edge detection

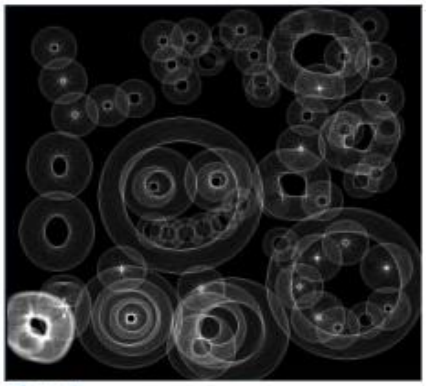

Figure 5

Image after voting

Ellipse detection

Ellipse detection in computer vision is the process of identifying and localizing elliptical shapes within an image. This task is crucial in various applications, including object recognition, medical imaging, robotics, and industrial inspection, as many real-world objects or their projections can be represented by ellipses.

Challenges in Ellipse Detection:

- **Noise and Quantization:**

  Image noise and the discrete nature of pixels can distort the ideal elliptical shape, making accurate edge detection and parameter estimation difficult.

- **Occlusion and Incompleteness:**

  Partial occlusion can result in incomplete ellipses, which are harder to detect and fit accurately compared to full ellipses.

- **Complex Backgrounds and Illumination Variations:**

  Cluttered backgrounds and uneven lighting conditions can introduce distractions or obscure elliptical features.

- **Computational Cost:**
  Some traditional methods, particularly those involving extensive parameter space searches, can be computationally intensive.
  Methods for Ellipse Detection:

  Ellipse detection methods can be broadly categorized into traditional and deep-learning-based approaches:

  Traditional Methods:

- **Hough Transform:** This method transforms image points into a parameter space where ellipses are represented by peaks. While robust to noise, it can be computationally expensive due to the high dimensionality of the ellipse parameter space.
- **Least Squares Fitting:** This approach fits a general conic equation to detected edge pixels, then identifies elliptical parameters. It is sensitive to outliers and noise.

- **Edge Following Methods:** These methods utilize the connectivity of edge pixels to identify arc segments and then attempt to fit ellipses to these segments. They can be less reliable for incomplete ellipses.
- **Geometric Feature-Based Methods:** These methods leverage geometric properties like tangents, centroids, and symmetry to identify and fit ellipses.

Deep Learning Methods:

- **Convolutional Neural Networks (CNNs):** With the rise of deep learning, CNN-based object detection models have been adapted for ellipse detection. These models learn to identify elliptical features directly from image data, offering improved accuracy and robustness to noise and complex backgrounds compared to traditional methods.
- **Specialized Ellipse Detectors:** Recent research focuses on designing dedicated neural network architectures, like EDNet, that incorporate geometric properties of ellipses into their design and loss functions for enhanced accuracy, especially for small or occluded ellipses.
  Key Aspects of Modern Ellipse Detection:
- **Robustness:**

  Methods aim to be robust against noise, occlusion, and varying lighting conditions.
- **Accuracy:**

  Precise localization and parameter estimation of ellipses are crucial for many applications.
- **Efficiency:**

  Real-time or near real-time performance is often a requirement, particularly in robotics and industrial automation.
- **Handling Incompleteness:**
  Techniques are developed to effectively detect and fit partially visible ellipses.

<u>Generalized Hough Transform</u>

The Generalized Hough Transform (GHT), introduced by Dana Ballard in 1981, is a powerful image processing technique that extends the basic Hough Transform to recognize objects of arbitrary, complex shapes, not just analytically defined ones like lines or circles. It works by creating a precomputed R-table (reference table) that stores the spatial relationship between a shape's boundary points and an arbitrary reference point (or "anchor point"). This allows the GHT to perform template matching, detect objects even with partial occlusion or deformation, and find an object's location, orientation, and scale by accumulating "votes" in a multi-dimensional parameter space.

How it Works:

1. **Create the R-table:**

- A prototype of the target shape is defined.
- An anchor point (e.g., the centroid) is chosen for the shape.
- For every boundary point on the prototype shape, the algorithm computes its gradient direction (angle of the tangent) and the vector (distance and angle) from that point to the anchor point.
- This information is stored in the R-table, which maps gradient angles to these offset vectors.

2. **Process the Input Image:**

- The image is typically edge-detected to obtain boundary points.
- For each edge point found in the image:
- Its gradient direction is determined.
- This angle is used to look up corresponding offset vectors (r, α) in the R-table.
- For each offset vector, a "vote" is cast in the multi-dimensional parameter space (e.g., for location (x, y), rotation θ, and scale s).

3. **Find the Object:**

- Locations in the accumulator array with a high number of accumulated votes are identified as potential instances of the target shape. These peaks correspond to the location, orientation, and scale of the object in the image.

Key Features and Advantages:

- **Arbitrary Shape Detection:** Unlike the standard Hough Transform, GHT can detect any shape, not just those described by simple equations.
- **Robustness to Occlusion:** It can recognize objects even if they are partially obscured.
- **Tolerance to Noise and Deformation:** The GHT is robust to small deformations and noise in the shape.
- **Template Matching:** It functions as a powerful template matching method, effectively identifying a model shape within a larger image.

Spatial Matched Filter

In Computer Vision, spatial matched filtering is a template matching technique that uses cross-correlation between a known pattern (template) and a larger image to find occurrences of that pattern. By sliding a filter mask (the template itself or its inverse) across the image and calculating the sum of products at each location, it generates a new image highlighting areas where the pattern matches the image. This allows for the detection and localization of a specific object or feature within an image.

How it Works

1. **Template Definition:**

   You have a small, known image or pattern (the template) you want to search for in a larger image.

2. **Cross-Correlation:**

   The core operation is cross-correlation. The filter mask (the template) is moved across the larger image, pixel by pixel.

3. **Pixel-wise Multiplication:**

   At each position, the values of the template are multiplied by the corresponding pixel values of the image under the mask.

4. **Summation:**

   The results of these multiplications are summed up to produce a single value for that position in the output image.

5. **Output Image:**

This process is repeated for every possible position of the template in the larger image, resulting in a correlation surface.

6. **Detection:**
   High values in the output image indicate strong matches between the template and the image, revealing the location of the pattern.
   Key Characteristics and Applications

- **Feature Detection:**

  It's effective for finding known, well-defined features or objects, such as a specific type of circuit or text in an image.

- **Template Matching:**

  The process is essentially a form of template matching, where the filter acts as a "template" for the object being sought.

- **Linear Filter:**

  It is a type of linear spatial filter, as it involves a linear operation (sum of products) across the neighborhood of pixels.

- **Noise Sensitivity:**
  While effective for known patterns, the performance of matched filters can be affected by noise and variations in the image.
  In essence, a spatial matched filter "looks" for a specific shape or pattern by seeing how well it aligns with the image at every possible location, outputting a map of similarity.

GHT for ellipse detection

The Generalized Hough Transform (GHT), also known as the Generalised Hough Transform (GHT), is a modification of the traditional Hough Transform that allows for the detection of any arbitrary shape, including ellipses, in an image by using a pre-defined reference shape or model. To detect ellipses using GHT, a reference table is first built for the desired elliptical shape, and then this table is used to match against the input image to find peaks in an accumulator array, which correspond to the detected ellipses.

How it works:

1. **Reference Table Creation:**

   A reference image or model of the ellipse is used to create a reference table, often called the R-Table. This table stores information about the shape's structure, such as the relationship between edge points and the shape's centroid or another reference point.

2. **Edge Detection:**

   The input image is pre-processed to detect the edges of potential shapes, typically using edge detection algorithms.

3. **Voting in Accumulator Array:**

   For each edge point in the image, the GHT uses the information from the reference table to "vote" for potential parameters of the elliptical shape.

4. **Peak Detection:**
   The votes are accumulated in a high-dimensional space that represents the possible parameters of the ellipse (e.g., center, major axis, minor axis, and orientation). Peaks in this accumulator array indicate

the presence of an ellipse, with the location of the peak corresponding to the specific parameters of the detected ellipse.
Key advantages:

- **Arbitrary Shapes:**

  GHT can detect arbitrary shapes, not just simple analytic shapes like lines and circles, making it useful for detecting irregular shapes or objects with varying forms.

- **Robustness:**
  It is generally robust to missing data points in the image, as it works by matching a model to edge information.
  Limitations:

- **Computational Cost:**

  GHT can be computationally intensive and require a large amount of memory, especially for complex shapes with many parameters, as it involves high-dimensional accumulator arrays.

- **Parameter Space:**
  Ellipse detection, in particular, can involve a large parameter space (e.g., five parameters for an ellipse: center coordinates x and y, semi-major axis a, semi-minor axis b, and orientation θ), which increases storage and computational demands.

GHT for feature collation

In computer vision, a Generalized Hough Transform (GHT) is a pattern recognition technique used to detect arbitrary shapes within an image, extending the traditional Hough transform's capability beyond simple geometric forms like lines and circles. The GHT works by creating a reference template of the object's shape, and then uses the detected edge points in an image to "vote" for candidate locations, orientations, and scales of the object. This approach converts the complex problem of global pattern matching into a more manageable local peak searching problem, effectively collating image features to find a match for the known model.

How GHT Works for Feature Collation

1. Reference Shape and Localization Point:

   A model shape is defined, including a chosen reference point (or "localization point").

2. Edge Detection:

   Edge pixels are detected in the input image.

3. R-Table (Reference Table):

   An R-table is pre-computed based on the reference shape. For each edge point in the template, this table stores the relative coordinates (r, α) of the reference point.

4. Voting Process:

   For each edge point in the input image, its gradient information (magnitude and direction) is used to look up corresponding (r, α) pairs in the R-table.

5. Accumulator Array:

   Each (r, α) pair is used to calculate a candidate position (xc, yc) for the reference point in the image. This calculation considers potential transformations such as rotation and scaling, which are also parameterized in the accumulator space.

6. Peak Detection:
   Accumulator cells that receive many votes indicate potential locations of the reference point of the shape. A local maximum in the accumulator array corresponds to a highly probable location and orientation of the object.
   Advantages of GHT

- Detects Arbitrary Shapes:

  It can identify objects of any shape, not just predefined geometric figures.

- Robustness:

  The GHT is robust to image noise, partial occlusions, and the presence of other structures in the image.

- Handles Transformations:
  It can detect objects that are translated, rotated, and scaled relative to the model.
  Limitations and Improvements

- Computational Complexity:

  Traditional GHTs can have high computational complexity, especially for complex 3D objects or when considering multiple transformations.

- Storage Requirements:

  The four-dimensional accumulator space can lead to significant storage needs.

- Occlusion Handling:

  While robust to moderate occlusion, severe occlusion can still generate many false solutions.

- Adaptive Algorithms:

  Researchers have developed adaptive algorithms, like the dual-point GHT, to optimize the R-table and reduce spurious votes.

- Linearization Techniques:
  Methods like the Linear GHT (LIGHT) and coordinate transformations have been proposed to improve parallelization and detection of curved features.

Object Location in Computer Vision

In computer vision, object localization is the task of identifying the specific position of an object in an image by drawing a bounding box around it, defining its spatial extent. It is a fundamental component of the broader task of object detection, which also includes classifying the object (e.g., "car," "person"). Deep learning models, especially Convolutional Neural Networks (CNNs), are commonly used to perform object localization by learning to recognize visual patterns and features from large datasets.

How Object Localization Works

1. **Object Detection:**

   The process often starts with an object detection algorithm that identifies potential objects within an image.

2. **Bounding Box Creation:**

   For each detected object, a bounding box (a rectangle) is drawn to precisely outline its boundaries.

3. **Feature Learning:**

Deep learning models, such as CNNs, are trained on massive datasets to learn the visual features that define different objects. This allows them to correctly identify and locate these objects in new images.
Key Aspects of Object Localization

- **Bounding Box:**

  This is the rectangular boundary that precisely defines an object's size and position within the image.

- **Accuracy:**

  Refers to how well the generated bounding box aligns with the actual object in the scene.

- **Real-time Performance:**

  The ability of the algorithms to perform localization in a very short amount of time, making it suitable for live applications.

- **Multi-class Support:**
  The capability to localize and identify various distinct object types in a single image.
  Object Localization vs. Object Detection

  While often used interchangeably, object localization is distinct from object detection.

- **Object Localization:** Identifies the position of an object.
- **Object Detection:** A combination of object localization (finding the object's location) and object classification (identifying the object's type or category).
  Applications

  Object localization is crucial for various applications, including:

- **Object Tracking:** Helps track the precise movement of objects in real-time.
- **Augmented Reality:** Essential for overlaying digital information onto real-world objects.
- **Self-Driving Cars:** Enables autonomous vehicles to understand their surroundings by precisely locating other vehicles, pedestrians, and obstacles.