21AIE332T- Image and Video Processing

Dr. K. Lakshmi Narayanan
Assistant Professor
Department of Computational Intelligence
SRMIST - KTR Campus

Image Modulation Models

1. Image Modulation Models: A Detailed Discussion

2. Single-Component Demodulation

- 2.1 Resolving Ambiguities in the Model
- 2.2 Multidimensional Energy Separation
- 2.3 Demodulation by Complex Extension

3. Multicomponent Demodulation

- 3.1 Dominant Component Analysis
- 3.2 Channelized Component Analysis

Image Modulation Models

- Image modulation models are fundamental in image processing and computer vision, particularly in analyzing spatial variations in images.
- These models help describe how image signals can be decomposed and processed for various applications, including texture analysis, feature extraction, and image enhancement.
- One of the main goals of image modulation models is to represent image intensity as a combination of amplitude and phase modulated components.
- This is inspired by the classical signal processing techniques used in 1D signal analysis, particularly in demodulation techniques applied to speech and radio signals.
- In image processing, these methods allow for effective analysis of fine-scale structures and textures.

Image Modulation Models

2. Single-Component Demodulation

- Single-component demodulation refers to the process of extracting the amplitude and phase of a single modulated component from an image.
- This is useful in cases where the image contains a dominant texture or frequency component that needs to be analyzed independently.
- The primary goal of single-component demodulation is to estimate the amplitude envelope and phase of an image function, often represented as:

$$I(x, y) = A(x, y) \cos(\phi(x, y))$$

where:

- $A(x, y)$ is the amplitude (or envelope) function,
- $\phi(x, y)$ is the phase function,
- $I(x, y)$ represents the image intensity.

Since real-world images are often complex and contain multiple overlapping textures, estimating $A(x, y)$ and $\phi(x, y)$ is not always straightforward. Several methods exist to resolve the ambiguities and challenges in demodulation.

Image Modulation Models

2.1 Resolving Ambiguities in the Model

One of the main challenges in single-component demodulation is that the phase and amplitude estimation can be ambiguous due to:

- Interference from multiple frequency components,
- Noise in the image,
- Lack of a well-defined dominant orientation.

To resolve these ambiguities, different filtering and decomposition techniques are used. For instance, bandpass filtering via Gabor or wavelet transforms helps isolate the dominant frequency components, thus reducing interference.

Another approach is **phase unwrapping**, which ensures smooth transitions in phase estimation, preventing artificial discontinuities. Techniques such as local energy-based methods also help in achieving robust phase and amplitude extraction.

Image Modulation Models

2.2 Multidimensional Energy Separation

Multidimensional energy separation extends traditional energy separation methods (used in 1D signals) to images. It is based on the **Teager-Kaiser Energy Operator (TKEO)**, which is used to compute instantaneous amplitude and frequency without the need for complex signal transformations.

For a 2D image $I(x, y)$, the local energy can be computed using directional derivatives as follows:

$$E_x = I_x^2 - I_x I_{xx}, \quad E_y = I_y^2 - I_y I_{yy}$$

where I_x and I_y are the first-order derivatives in the horizontal and vertical directions, and I_{xx}, I_{yy} are the second-order derivatives.

By using these energy terms, the local frequency and amplitude can be extracted more efficiently, especially in textured images where traditional filtering may fail.

Image Modulation Models

2.3 Demodulation by Complex Extension

- Another way to perform single-component demodulation is by **complex extension**, where the real image signal is extended into a complex-valued representation using quadrature filters or the Hilbert transform. This allows for an analytic signal representation:

$$I_c(x, y) = I(x, y) + jH[I(x, y)]$$

where $H[I(x, y)]$ is the Hilbert transform of the image. From this complex representation, the amplitude $A(x, y)$ and phase $\phi(x, y)$ can be directly computed:

$$A(x, y) = \sqrt{I(x, y)^2 + H[I(x, y)]^2}$$

$$\phi(x, y) = \tan^{-1}\left(\frac{H[I(x, y)]}{I(x, y)}\right)$$

- This method is particularly useful when extracting phase information for applications such as motion estimation and phase-based image analysis.

Image Modulation Models

3. Multicomponent Demodulation

While single-component demodulation is useful for analyzing images with a dominant frequency, real-world images often contain multiple overlapping modulated components. Multicomponent demodulation extends the techniques above to separate and analyze different components individually.

3.1 Dominant Component Analysis

One way to handle multiple components is through **dominant component analysis (DCA)**, which involves:

1. Identifying the strongest modulated component in the image,
2. Extracting its amplitude and phase,
3. Subtracting this component from the original image,
4. Repeating the process iteratively for the next strongest component.

This is often done using bandpass filtering combined with adaptive filtering techniques.

Image Modulation Models

3.2 Channelized Component Analysis

Channelized analysis refers to using a **filter bank** approach, where the image is decomposed into multiple frequency channels, and each channel is demodulated separately. This method is common in Gabor wavelet decomposition and steerable pyramid transforms.

Each filter in the bank extracts components corresponding to different orientations and scales, which are then individually analyzed for amplitude and phase variations.

This approach is widely used in texture segmentation, pattern recognition, and biomedical image analysis, where multiple textural components exist in an image.

Image Modulation Models

Conclusion

Image modulation models provide powerful tools for analyzing image structure, particularly in textured images. While single-component demodulation methods such as multidimensional energy separation and complex extension are useful for isolating dominant structures, multicomponent demodulation techniques like dominant component analysis and channelized filtering allow for a more comprehensive decomposition.

The techniques discussed are widely used in computer vision applications, including:

•Texture classification,

•Image enhancement,

•Motion estimation,

•Medical imaging.

Image Segmentation

Segmentations :-

→ Image segmentations is the process of partioning is digital image into multiple system (set of pixel)

→ The goal of segmentation is to simplify or change the representation of an image into something i.e.; more meaningful and easier to analysis

→ Image segmentation is typically used to locate objects and boundaries in images.

Image Segmentation

REGION BASED SEGMENTATION :-

→ Region based segmentations is a technique for determining the region directly.

→ Region based technique relay on common patterns in intensity values within a cluster of neighbouring values/pixels.

→ The cluster is refer to as the region and the goal of the segmentations algorithm is to group regions according to their anatomical and functional role.

→ Segmentations is the process of Extracting and representating informations from an image ~~into~~ is to group pixels together into regions of similarity.

→ Region based segmentations method attempt to partician or group regions. according to common

Image Segmentation

→ This image properties consist of

* intensity values from original image or computed values based on image operator.
* Texture or patterns that are unique to Each regions
* Spartial profile that provides multi-dimensional image data.

Image Segmentation

→ The principle approach in region based segmentation are –

(i) Thresholding

(ii) Region growing

(iii) Region spilting or margin

(iv) clustering in feature space.

(v) Region growing is a simple region based segmentation methods it is also classified as a pixel based image segmentation method.

(vi) The basic formulation for region basedsegmentation is

(a) $\bigcup_{i=1} R_i = R$ (segmentation most be complete that means Every pixels most be in the region).

Image Segmentation

(b) R_i is a connected region $i = 1, 2 \dots n$
(requires that the points in a region most be connected in some pre-defined sense.)

(c) $R_i \cap R_j = \phi$, $i \neq j$ (~~req~~ indicates that the region most be disjoined.)

(d) $P(R_i)$ = true for $i = 1, 2 \dots n$
(deals with the properties that most be satisfied by the pixels in the segmented region. for
Example → $R(R_i)$ = true for if all pixel in R_i have the same gray ~~sale~~ scale level.

(e) $P(R_i \cup R_j)$ = false for $P(R_i)$ is a logical predicate defined over the point in set R_i and ϕ is the all set. (indicates that the region Region R_i and R_j are different in the sense of predicate (P))

Image Segmentation

Concept of Seed points :-

→ The first step in region growing is to select a set of seed points.

→ It's selections is based on some user criteria

→ The initial region begins at the Exact at the location.

→ ~~The initial region begins~~ of the seeds.

→ The region are then grown from these seeds points to adjacent points depending on a region membership criteria.

→ Region growing as a procedure that groups pixels or subregion into larger region.

Image Segmentation

→ Region growing method can provides origin image which have clear Edges with good segmentation result

→ Region based segmentation technique is computationally expensive. It is a local method with no global view view of the problem sensitive to noise

→ Region based segmentation approach are based on pixel properties such as homogenity and spartial

Image Segmentation

Fundamentals of Image Segmentation

Image segmentation is the process of dividing an image into meaningful regions or objects. It is a crucial step in image analysis, as it helps in object detection, recognition, and scene understanding. Segmentation techniques are broadly classified into:

1. **Discontinuity-Based Methods** – Detects edges, lines, or points where there is a significant change in intensity.
2. **Similarity-Based Methods** – Groups pixels based on similarity in intensity, texture, or color (e.g., region growing, clustering).

Image Segmentation

1. Point, Line, and Edge Detection

1.1 Point Detection

Point detection identifies individual pixels with intensity changes using the **Laplacian filter** or other difference operators. If f(x,y) is the intensity function of an image, a simple detection mask is:

$$W = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

A pixel is classified as a point if the response R(x,y) exceeds a predefined threshold.

1.2 Line Detection

Line detection finds straight or curved lines using convolution masks like:

$$W_{\text{horizontal}} = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

$$W_{\text{vertical}} = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

Applying these masks highlights lines in the corresponding direction.

Image Segmentation

1.3 Edge Detection

Edges represent boundaries of objects and are detected using **gradient-based operators** such as:

•**Sobel Operator** (first derivative approximation):

- **Sobel Operator** (first derivative approximation):

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

The edge magnitude is computed as:

$$G = \sqrt{G_x^2 + G_y^2}$$

- **Canny Edge Detector** – Uses Gaussian smoothing and non-maximum suppression to extract strong edges while reducing noise.

Image Segmentation

2. Thresholding

Thresholding is a simple and effective segmentation technique where pixels are classified based on intensity levels.

2.1 Global Thresholding

A fixed th

$$g(x,y) = \begin{cases} 1, & \text{if } f(x,y) \geq T \\ 0, & \text{otherwise} \end{cases}$$

Otsu's method finds the optimal T by maximizing the variance between object and background intensities.

2.2 Adaptive Thresholding

For images with varying illumination, different thresholds are applied to sub-regions.

2.3 Multilevel Thresholding

When an image contains multiple objects, multiple thresholds T1,T2,…,Tn separate regions based on intensity.

Image Segmentation

3. Segmentation by Region Growing and Region Splitting & Merging

3.1 Region Growing

Starts with **seed points** and expands based on similarity criteria (e.g., pixel intensity, texture).

Steps:

1. Select initial seed pixels.
2. Grow the region by adding neighboring pixels if they meet a similarity criterion.
3. Stop when no more pixels satisfy the condition.

This method works well for homogenous regions but is sensitive to noise and seed selection.

3.2 Region Splitting and Merging

Instead of growing regions, this method:

- **Splits** an image into smaller regions until each is homogenous.
- **Merges** adjacent regions that meet a similarity condition.

This technique is implemented using **quadtree decomposition**, where an image is recursively divided into four quadrants until homogeneity is achieved.

Image Segmentation

4. Region Segmentation Using Clustering and Superpixels

4.1 Clustering-Based Segmentation

Clustering partitions pixels into groups based on intensity, color, or texture.

4.1.1 K-Means Clustering

- Select k initial cluster centers.
- Assign each pixel to the nearest center.
- Update the cluster centers iteratively.

4.1.2 Mean Shift Clustering

- Moves data points toward the densest regions.
- Does not require a predefined number of clusters.

4.2 Superpixels

Superpixels group pixels into perceptually meaningful regions, preserving object boundaries.

- **SLIC (Simple Linear Iterative Clustering)** generates compact, uniform superpixels using K-means clustering. Superpixels improve segmentation efficiency by reducing the number of elements to process.

Image Segmentation

5. Region Segmentation Using Graph Cuts

Graph-based segmentation treats an image as a graph where:

•**Nodes** are pixels.

•**Edges** represent similarities (e.g., intensity difference).

5.1 Minimum Cut Algorithm

Partition the graph into foreground and background by finding the **minimum cut**, which minimizes the cost of separation.

5.2 Normalized Cut (N-Cut)

Improves segmentation by considering both:

•**Within-region similarity**,

•**Between-region dissimilarity**.

Graph cuts are widely used in interactive segmentation and medical image analysis.

Image Segmentation

6. Segmentation Using Morphological Watersheds

Watershed segmentation treats an image as a topographic surface where:

- Bright regions are **peaks**,
- Dark regions are **valleys**.

6.1 Marker-Based Watershed Algorithm

1. Compute the gradient magnitude of the image.
2. Place **markers** inside objects and background.
3. Flood-fill the regions from markers.

This method effectively separates touching objects, such as cells in biomedical images.

Image Segmentation

7. The Use of Motion in Segmentation

Motion-based segmentation identifies objects in videos based on their movement.

7.1 Optical Flow-Based Segmentation

•Computes pixel motion between consecutive frames.

•**Lucas-Kanade** and **Horn-Schunck** methods estimate motion vectors.

7.2 Background Subtraction

•Compares current and reference frames.

•Detects foreground objects by removing static background.

7.3 Motion Segmentation with Clustering

•Groups pixels with similar motion characteristics using K-means or Mean Shift.

Motion segmentation is crucial in video surveillance, object tracking, and autonomous driving.

Image Classification and Segmentation

IMAGE SEGMENTATIONS

→ Image segmentations is the process of particianing a digital image into multiple segments.

→ The goal of segmentation is to simplify or change the representations of an image into the form i.e.; more meaningful and Easier to analysis.

→ Image segmentation is used to locate object and boundaries in ~~object~~ images

→ It is the process of assigning a level to every pixel in an image such that pixels with the same level share the certain characteristics.

Image Classification and Segmentation

→ Application of image segmentations are:

(i) content based image retriever

(ii) machine vission

(iii) medical imaging including volume rentered images from computer tomography and magnetic resonanke imaging.
(Locate tumors and surgery planning)

(iv) pedestrian delection and face ~~detations~~ detections.

(v) traffic control system.

(vi) védeo survellance

Image Classification and Segmentation

(vii)
* segmentations attempts to partician the pixel of an image to groups that coordinations with the objects in an images.

→ segmentations algorithm generally based on one of the two basic properties of intensity values.

(a) Discontinuity

To partician an image based on abrupt changes in intensity (such as edges).

(b)
* Similarity :-

To partician an image into region that are according to a set of pre defined criteria.

Image Classification and Segmentation

Detections of ~~Discountities~~: Discontinuities

There are 3 basic type of gray level discontinuities -

I. points
II. lines
III. Edges

→ The common ~~one~~ m: is used to run a mask through the image.

POINT DETECTIONS:

→ The mask is the same as the mask of laplacian operation

→ The only differentes that has considered of interest are those large enough (as determine by) to be considered isolated points

$$|R| > T$$

Image Classification and Segmentation

LINE DETECTIONS :-

Horizontal mask will result with maximum response when a line passed through the middle row of a mask with a constant ~~gra~~ back ground.

→ The similar idea is used with other mask the prefer directions of each mask is weighted with a larger coefficient than other possible directions.

Image Classification and Segmentation

Edge detections :-

→ segmentations by finding pixels on a vision boundary!

→ The edges found by looking at neighbouring pixel.

→ Region boundary found by measuring gray value differences.

★ A Edge is a set of connected pixels that lay on the boundary between two regions.

An edge is a local concept whereas a region boundary owing to the way it is ~~differed~~ defined is a more global idea.

and boundary detections :-

Image Classification and Segmentation

global area.

Edge Linking and boundary detections :-

→ Edge linking and boundary detections operations are the fundamental steps in any image understanding

→ Edge linking process takes an un-ordered set of Edge pixels produced by an Edge detector as an input to form an ordered list of Edges.

→ Local Edge informations are utilised by Edge linking operations. Thus, Edge detections algorithm typically as followed by linking procedure to assemble Edge pixels into ~~Thus~~, meaningful Edges.

Introduction to Multiband Techniques for Texture Classification and Segmentation

Texture analysis is pivotal in image processing, enabling machines to interpret visual content through spatial patterns. This introduction explores multiband techniques, emphasizing Gabor filters, which excel in texture tasks due to their biological plausibility and joint spatial-frequency localization.

1.1 Image Texture

Image texture refers to the spatial arrangement of pixel intensities, characterized by properties like regularity, contrast, and directionality. Critical in applications such as medical imaging and remote sensing, texture analysis involves classification (labeling textures), segmentation (partitioning regions), and retrieval (matching textures in databases). Bovik highlights texture's role in mimicking human visual perception, necessitating methods that capture both local and global patterns.

1.2 Gabor Features for Texture Classification and Image Segmentation

Gabor filters, modeled after mammalian visual cortex cells, decompose images into frequency and orientation subbands. Their optimal resolution in both spatial and frequency domains makes them ideal for texture analysis. Applications include feature extraction for SVM-based classification and edge-aware segmentation, leveraging their ability to isolate texture boundaries.

Introduction to Multiband Techniques for Texture Classification and Segmentation

2. Gabor Functions: Theory and Implementation

2.1 One-Dimensional Gabor Function

The 1D Gabor function combines a Gaussian envelope with a complex sinusoid:

$$g(t) = e^{-\pi(t/\sigma)^2} \cdot e^{j(2\pi f t+\phi)}$$

Parameters:

- σ: Gaussian bandwidth (scale).
- f: Central frequency.
- ϕ: Phase offset.

2.2 Analytic Gabor Function

The analytic form excludes negative frequencies via the Hilbert transform, yielding a complex signal $g_a(t) = g(t) + j\mathcal{H}\{g(t)\}$, enhancing frequency localization.

2.3 Two-Dimensional Gabor Function: Cartesian Form

A separable 2D extension in Cartesian coordinates:

$$g(x, y) = e^{-\pi[(x/\sigma_x)^2+(y/\sigma_y)^2]} \cdot e^{j(2\pi(f_x x+f_y y)+\phi)}$$

Variables σ_x, σ_y control spatial scaling, while f_x, f_y define orientation.

2.4 Two-Dimensional Gabor Function: Polar Form

In polar coordinates, orientation θ and radial frequency f_r parameterize the filter:

$$g(r, \theta) = e^{-\pi[(r/\sigma_r)^2+(\theta/\sigma_\theta)^2]} \cdot e^{j(2\pi f_r r+\phi)}$$

This form aligns with rotational symmetries in natural textures.

Introduction to Multiband Techniques for Texture Classification and Segmentation

2.5 Multiresolution Representation with Gabor Wavelets

A filter bank of Gabor wavelets at multiple scales and orientations (e.g., 5 scales × 8 orientations) captures multiscale texture features. Bovik notes this mimics the human visual system's multichannel processing.

3. Microfeature Representation

3.1 Transformation into Gabor Space

Convolving an image I(x,y) with Gabor filters yields complex responses Rk(x,y)where k*k* indexes scale/orientation. Magnitude and phase encode local texture properties.

3.2 Local Frequency Estimation

Dominant frequencies at each pixel are identified by maximizing |Rk(x,y)| across scales/orientations, yielding a frequency map for segmentation.

3.3 Transformation into Microfeatures

Microfeatures are formed by downsampling and concatenating |Rk(x,y)|, creating compact descriptors. Dimensionality reduction (PCA) may follow, per Bovik's recommendations.

Introduction to Multiband Techniques for Texture Classification and Segmentation

4. The Texture Model

4.1 The Texture Micromodel

Micromodels represent localized texture elements (e.g., edges, spots) via Gabor responses. Each microfeature vector $\mathbf{m}_i$ corresponds to a pixel neighborhood.

4.2 Microfeature

A microfeature $\mathbf{m}_i$ aggregates filter magnitudes in a window, encoding texture primitives. Statistical measures (mean, variance) enhance robustness.

4.3 The Texture Macro model

The macro model synthesizes microfeatures into global texture properties. Methods include:

- **Statistical Models**: Gaussian mixtures or Markov random fields.
- **Structural Models**: Graph-based representations of microfeature relationships.

Introduction to Multiband Techniques for Texture Classification and Segmentation

5. Experimental Results and Classification

Using the Brodatz database, Gabor features achieve >95% accuracy with SVM classifiers. Confusion matrices reveal challenges distinguishing isotropic textures. Bovik's benchmarks show superiority over co-occurrence matrices in heterogeneous datasets.

6. Image Segmentation Using Texture

Segmentation employs clustering (k-means, mean-shift) on Gabor microfeatures. Edge detection via phase discontinuities improves boundary precision. Results on satellite imagery demonstrate effective land-cover partitioning.

7. Image Retrieval Using Texture

Retrieval systems use L2-distance or cosine similarity on Gabor feature vectors. Bovik's experiments highlight robustness to illumination changes, with precision-recall curves outperforming wavelet transforms.

Video Segmentation and Change Detection

Video segmentation partitions video sequences into spatiotemporally coherent regions, while change detection identifies significant variations between frames. These tasks are critical for surveillance, video compression, and event recognition.

2. Change Detection

2.1 Detection Using Two Frames

Simple frame differencing computes pixel-wise intensity differences between consecutive frames. Thresholding yields a binary change mask. However, this method fails under global illumination changes or noise. Bovik emphasizes using **normalized correlation** or **Mahalanobis distance** to improve robustness.

2.2 Temporal Integration

Temporal smoothing (e.g., exponential averaging) reduces noise by accumulating evidence over multiple frames. Bayesian frameworks model pixel states (static/dynamic) using prior probabilities and likelihoods, updating beliefs over time.

Video Segmentation and Change Detection

2.3 Combination with Spatial Segmentation

Integrating motion cues with spatial features (e.g., edges, texture) enhances accuracy. Markov Random Fields (MRFs) unify spatial and temporal constraints, penalizing label inconsistencies between neighboring pixels.

2.4 Dominant Motion Segmentation

Dominant motion refers to the largest coherent movement in a scene (e.g., a walking person). A parametric model (e.g., affine or projective motion) is fitted to the dominant flow field using least squares. Outliers (non-dominant motions) are discarded, segmenting the foreground.

3. Dominant Motion Segmentation

3.1 Segmentation Using Two Frames

Optical flow between two frames estimates pixel displacements. Dominant motion is estimated via RANSAC, which iteratively fits models to inliers while rejecting outliers.

3.2 Temporal Integration

Kalman filters or particle filters track dominant motion parameters over time, refining estimates using historical data. This mitigates transient noise and occlusions.

3.3 Multiple Motions

When multiple motions coexist (e.g., cars in traffic), layered motion models decompose the scene into independently moving planes. Bovik notes the use of **mixture models** or **hierarchical clustering** for this purpose.

4. Multiple Motion Segmentation

4.1 Clustering in Motion Parameter Space

Pixels are grouped based on similarity in motion parameters (e.g., velocity, affine coefficients). K-means or mean-shift clustering assigns labels, leveraging Bovik's emphasis on unsupervised techniques.

4.2 Maximum Likelihood (ML) Segmentation

ML estimates labels by maximizing the likelihood of observed motions under Gaussian noise assumptions. The Expectation-Maximization (EM) algorithm iteratively refines motion parameters and labels.

4.3 Maximum Posteriori (MAP) Segmentation

MAP incorporates spatial priors (e.g., MRFs) into ML frameworks, balancing data fidelity with label smoothness. Energy minimization via graph cuts yields optimal segmentations.

4.4 Region-Based Label Assignment

Region-merging algorithms combine small regions with consistent motion, using criteria like the **Fisher discriminant** to preserve boundaries.

5. Simultaneous Estimation and Segmentation

5.1 Modeling

Jointly estimates motion parameters and segmentation labels. A variational framework minimizes an energy functional:

5.2 An Algorithm

Alternating optimization iterates between motion estimation (via Lucas-Kanade) and segmentation (via MRF inference), converging to a stable solution.

6. Semantic Video Object Segmentation

6.1 Chroma Keying

A classic technique for foreground extraction, chroma keying isolates objects based on color (e.g., green screens). Bovik highlights its limitations in natural scenes but praises its efficiency in controlled environments.

Video Segmentation and Change Detection

6.2 Semi-Automatic Segmentation

User-guided tools (e.g., active contours, graph cuts) refine object boundaries. Interactive frameworks like **Intelligent Scissors** or **GrabCut** combine human input with automated edge detection.

Applications and Challenges

•**Surveillance**: Detecting intruders via change detection.

•**Video Coding**: MPEG-4 uses object-based segmentation for compression.

•**Challenges**: Occlusions, shadows, and dynamic backgrounds remain open problems

Adaptive and Neural Methods for Image Segmentation

1. **Introduction**
2. **Artificial Neural Networks**
3. **Perceptual Grouping and Edge-Based Segmentation**
4. **Adaptive Multichannel Modelling for Texture-Based Segmentation**
5. **An Optimization Framework**
6. **Image Segmentation by Means of Adaptive Clustering**
7. **Oscillation-Based Segmentation**
8. **Integrated Segmentation and Recognition**
    - Feature Extraction
    - Classification
    - Pattern Recognition
    - Techniques Specific to Segmentation

Adaptive and Neural Methods for Image Segmentation

Image segmentation, the process of partitioning an image into meaningful regions, is a cornerstone of computer vision. Adaptive and neural methods leverage dynamic parameter adjustment and biologically inspired models to address challenges like noise, variability, and complexity.

1. Artificial Neural Networks (ANNs) for Segmentation

ANNs mimic the human brain's parallel processing to learn hierarchical features from data. In segmentation, they adaptively refine boundaries based on pixel intensity, texture, or motion cues.

•**Architectures**:

- **Multilayer Perceptrons (MLPs)**: Map pixel neighborhoods to segmentation labels (foreground/background).
- **Convolutional Neural Networks (CNNs)**: Extract spatial hierarchies, enabling end-to-end segmentation (e.g., U-Net).

•**Training**: Supervised learning using labeled datasets, minimizing cross-entropy loss.

•**Advantages**: Robustness to noise and illumination changes. Bovik highlights early ANN applications for edge detection and region growing.

Adaptive and Neural Methods for Image Segmentation

2. Perceptual Grouping and Edge-Based Segmentation

Perceptual grouping organizes low-level features (edges, corners) into coherent structures using Gestalt principles (proximity, similarity).

•**Edge Linking**: Connects fragmented edges via adaptive thresholding or curvature analysis.

•**Active Contours (Snakes)**: Energy-minimizing curves guided by edge gradients and internal smoothness constraints.

•**Neural Integration**: CNNs predict edge maps, which are refined using graph-based grouping algorithms.

3. Adaptive Multichannel Modeling for Texture-Based Segmentation

Texture segmentation benefits from multichannel filtering (e.g., Gabor, wavelet) to capture spatial-frequency content.

•**Adaptive Filter Banks**: Dynamically adjust filter scales/orientations based on local texture properties.

•**Feature Fusion**: Combine responses from multiple channels using weighted fusion (learned via ANNs).

•**Bovik's Framework**: Emphasizes joint optimization of filter parameters and region labels for inhomogeneous

Adaptive and Neural Methods for Image Segmentation

4. An Optimization Framework

Segmentation is often posed as an energy minimization problem:

- $E(S) = \text{Data Term}(S) + \lambda \cdot \text{Smoothness Term}(S)$
    - **Data Term**: Measures fidelity to observed features (e.g., intensity, texture).
    - **Smoothness Term**: Penalizes label discontinuities (e.g., MRFs, graph cuts).
    - **Neural Solvers**: CNNs approximate energy functions, enabling real-time solutions.

5. Image Segmentation via Adaptive Clustering

Clustering algorithms group pixels based on feature similarity, with adaptivity improving robustness.

- **Adaptive k-means**: Dynamically adjusts cluster centers and counts using entropy criteria.
- **Fuzzy C-Means (FCM)**: Assigns soft membership probabilities, handling overlapping regions.
- **Spatial Constraints**: Incorporate spatial proximity into clustering (e.g., spatial FCM).

Adaptive and Neural Methods for Image Segmentation

6. Oscillation-Based Segmentation

Inspired by synchronized neural oscillations in the visual cortex, this method segments regions via phase coherence.

- **Coupled Oscillator Networks**: Pixels oscillate at frequencies tied to local features (e.g., color). Regions synchronize if features are similar.
  - **Phase Locking**: Segments emerge as groups of pixels with coherent oscillatory phases.
  - **Applications**: Medical imaging for tumor boundary detection.

7. Integrated Segmentation and Recognition

Jointly optimizing segmentation and recognition improves both tasks by sharing contextual information.

- **Recurrent Architectures**: Systems like Mask R-CNN predict segmentation masks and object labels simultaneously.
- **Feedback Loops**: Recognition results refine segmentation (e.g., semantic segmentation guiding edge detection).

Adaptive and Neural Methods for Image Segmentation

8. Feature Extraction, Classification, and Pattern Recognition

Feature Extraction:

- **Low-Level**: Edges (Canny), textures (GLCM), color histograms.
- **Mid-Level**: Superpixels (SLIC), region proposals.
- **High-Level**: Semantic features (CNN activations).

Classification Techniques:

- **SVMs**: Separate regions using hyperplanes in feature space.
- **Random Forests**: Ensemble learning for pixel-wise labeling.
- **Deep Learning**: Transformers for global context modeling (e.g., Vision Transformers).

Applications and Challenges

- **Medical Imaging**: Tumor segmentation via adaptive ANN architectures.
- **Autonomous Driving**: Real-time road-scene parsing using integrated CNNs.
- **Challenges**: Scalability to high-resolution images, handling occlusions, and interpretability.

Gradient-Based Edge Detection

Gradient-based methods detect edges by measuring intensity changes in an image. The gradient vector points in the direction of maximum intensity change, and its magnitude indicates edge strength.

2.1 Continuous Gradient

- The gradient of a continuous 2D image $f(x, y)$ is defined as:

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

- **Magnitude**: $|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$
- **Direction**: $\theta = \arctan\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

2.2 Discrete Gradient Operators

Discrete approximations of derivatives using convolution kernels:

- **Sobel Operator**:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- **Prewitt Operator**:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- **Roberts Cross Operator**:

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

These operators compute horizontal (G_x) and vertical (G_y) gradients, which are combined to find edge magnitude and direction.

Laplacian-Based Edge Detection

The Laplacian is a second-order derivative operator that highlights regions of rapid intensity change (zero-crossings).

3.1 Continuous Laplacian

For a continuous image $f(x, y)$:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

It is sensitive to noise and often paired with Gaussian smoothing.

3.2 Discrete Laplacian Operators

Common discrete kernels:

- **3x3 Laplacian**:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- **Diagonal Laplacian**: Includes diagonal elements for better edge detection.

3.3 Laplacian of Gaussian (LoG – Marr-Hildreth Operator)

1. **Gaussian Smoothing**:

   Apply a Gaussian filter $G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$ to reduce noise.

2. **Laplacian Operation**:

   Compute $\nabla^2(G * f)$.

- The combined LoG operator is:

$$\nabla^2 G(x, y) = \frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- **Zero-crossings** in the output indicate edges.

3.4 Difference of Gaussian (DoG)

- Approximates LoG by subtracting two Gaussians with different variances (σ_1 and σ_2):

$$\mathrm{DoG}(x, y) = G(x, y, \sigma_1) - G(x, y, \sigma_2)$$

- Computationally efficient alternative to LoG.

Laplacian-Based Edge Detection

The Canny edge detector is a multi-stage algorithm that processes an input image to produce a binary edge map. It leverages the discrete gradient operator (as discussed previously) and incorporates noise reduction, hysteresis thresholding, and edge thinning to achieve robust results.

4. Canny's Edge Detection Method

A multi-stage optimal edge detector:

1. **Noise Reduction**: Apply Gaussian filter.
2. **Gradient Calculation**: Use Sobel/Prewitt operators.
3. **Non-Maximum Suppression**: Thin edges by retaining only local maxima in the gradient direction.
4. **Double Thresholding**:
   - **High threshold**: Strong edges.
   - **Low threshold**: Weak edges (retained if connected to strong edges).
5. **Edge Tracking**: Hysteresis to link discontinuous edges.

5. Edge Detection in Color and Multispectral Images

Approaches:

1. **Per-Channel Processing**: Detect edges in individual color channels (e.g., RGB, HSV) and combine results.
2. **Vector-Based Methods**:
   - Compute gradients in multichannel space (e.g., Euclidean distance across channels).
   - Use **Di Zenzo's Multispectral Gradient**:

$$\nabla f = \sqrt{\sum_{i=1}^{n} \left(\frac{\partial f_i}{\partial x}\right)^2 + \left(\frac{\partial f_i}{\partial y}\right)^2}$$

3. **PCA-Based Methods**: Reduce dimensionality and detect edges in principal components.

Challenges:

- Color space selection (RGB vs. perceptually uniform spaces like CIELAB).
- Handling spectral redundancy in multispectral images.

Diffusion-Based Edge Detectors

Diffusion in image processing refers to a process that evolves an image over time to reduce noise while preserving structural features like edges. It is modeled as a partial differential equation (PDE) that governs how pixel intensities propagate spatially.

2. Scale Space and Diffusion

2.1 Scale Space and Isotropic Diffusion

•**Scale Space Theory**: Represents an image at multiple resolutions (scales) by convolving it with a Gaussian kernel of increasing variance σ.

$$I(x, y, \sigma) = I(x, y) * G(x, y, \sigma)$$

where $G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$.

- **Isotropic Diffusion**: Smooths the image uniformly in all directions. Governed by the heat equation:

$$\frac{\partial I}{\partial t} = \nabla^2 I$$

  - **Limitation**: Blurs edges indiscriminately, degrading image structure.

Diffusion-Based Edge Detectors

2.2 Anisotropic Diffusion

Proposed by Perona and Malik (1990), anisotropic diffusion selectively smooths regions based on local image gradients, preserving edges:

$$\frac{\partial I}{\partial t} = \operatorname{div}\left(g(|\nabla I|)\nabla I\right)$$

- $g(|\nabla I|)$: **Diffusion coefficient**, a decreasing function of the gradient magnitude.
- **Key Idea**: Reduces smoothing near edges (high $|\nabla I|$) and enhances smoothing in homogeneous regions (low $|\nabla I|$).

3. Implementation of Diffusion

3.1 Diffusion Coefficient

Defines how diffusion adapts to local gradients. Common choices:

1. **Perona-Malik Functions**:

$$g(|\nabla I|) = \frac{1}{1 + \left(\frac{|\nabla I|}{K}\right)^2} \quad \text{or} \quad g(|\nabla I|) = e^{-\left(\frac{|\nabla I|}{K}\right)^2}$$

   - K: Contrast parameter controlling edge sensitivity.

3.2 Diffusion PDE

Discretized anisotropic diffusion equation:

$$I^{t+1}(x,y) = I^t(x,y) + \Delta t \cdot \operatorname{div}\left(g(|\nabla I|)\nabla I\right)$$

- Δt: Time step (typically $0.1 \leq \Delta t \leq 0.25$).

Diffusion-Based Edge Detectors

3.3 Variational Formulation

Anisotropic diffusion can be derived by minimizing an energy functional:

$$E(I) = \int_\Omega \left(\lambda \cdot \rho(|\nabla I|) + \frac{1}{2}(I - I_0)^2 \right) dx\, dy$$

- ρ: Edge-stopping function (e.g., $\rho(s) = \sqrt{1 + s^2}$).

3.4 Multiresolution Diffusion

Applies diffusion across multiple scales (e.g., Gaussian pyramid or wavelet decompositions) to handle edges at different resolutions.

3.5 Multispectral Anisotropic Diffusion

Extends diffusion to color/multispectral images by coupling channels:

- **Vector-Valued Diffusion**:

$$\frac{\partial I_i}{\partial t} = \mathrm{div}\left(g(|\nabla I|) \nabla I_i \right), \quad i = 1, 2, ..., N$$

  - $|\nabla I|$: Gradient magnitude computed across all channels (e.g., Euclidean norm).

4.1 Edge Detection by Thresholding

1. Apply anisotropic diffusion to denoise the image.
2. Compute gradient magnitude $|\nabla I|$.
3. Threshold $|\nabla I|$ to detect edges.

4.2 Edge Detection From Image Features

- Use features from the diffused image (e.g., zero-crossings of the Laplacian) for edge localization.

4.3 Quantitative Evaluation

Metrics to assess performance:

- **Precision/Recall**: Compare detected edges with ground truth.
- **Edge Localization Error**: Measure deviation from true edges.
- **Noise Suppression Ratio**: Quantify noise reduction in homogeneous regions.

Software for Image and Video Processing

Image and video processing software environments provide frameworks for developing, testing, and deploying algorithms. Applications span medical imaging, computer vision, remote sensing, and multimedia. Bovik's handbook establishes foundational concepts (e.g., filtering, compression, feature extraction), which these tools operationalize. These environments balance usability, performance, and flexibility.

2. Development Environments

2.1 MATLAB

- **Description**: High-level language with toolboxes (e.g., Image Processing Toolbox, Computer Vision Toolbox).
- **Features**: Interactive prototyping, Simulink for model-based design, and GPU acceleration.
- **Applications**: Academia, industry (automotive, robotics).

2.2 IDL (Interactive Data Language)

- **Description**: Data analysis and visualization language.
- **Features**: Array-oriented syntax, support for large datasets.
- **Applications**: Astronomy, earth sciences (satellite data processing).

Software for Image and Video Processing

3. Compiled Libraries

3.1 Intel IPP (Integrated Performance Primitives)

- **Description**: Optimized functions for Intel CPUs (image compression, computer vision).
- **Use Case**: Real-time video processing, leveraging SIMD instructions.

3.2 IMSL (International Math & Statistics Library)

- **Description**: Numerical routines (FFT, optimization).
- **Applications**: Scientific computing, statistical analysis.

4. Source Code & Standards

4.1 Numerical Recipes

- **Description**: Book series with code samples (C, Fortran).
- **Limitations**: Restrictive licensing for commercial use.

4.2 Encoding Standards

- **Image**: JPEG, PNG (spatial/transform compression).
- **Video**: H.264/HEVC (motion compensation).

Software for Image and Video Processing

5. Specialized Processing & Visualization Environments

5.1 PCI Geomatica

- **Focus**: Remote sensing (satellite imagery, GIS integration).

5.2 ENVI

- **Focus**: Hyperspectral data analysis (e.g., mineralogy, environmental monitoring).

Other Software

- **OpenCV**: Real-time computer vision (object detection, ML).
- **ImageJ**: Open-source microscopy/image analysis.
- **ParaView/VisIt**: Large-scale scientific visualization.