Enabling Technologies for Sports (5XSF0) Module 6

Segmentation

Jungong Han, Dirk Farin, Sveta Zinger

(s.zinger@tue.nl)

Outline

- Introduction
- Color Segmentation
  - region-growing
  - *region-merging*
  - *watershed*
- Background Subtraction
  - background generation
  - change detection
- Model-based Object Detection
  - graph model

Applications of Object Segmentation

- Surveillance
  - detect people in restricted areas
  - detect abnormal behavior (motion pattern)
- Video/Image Analysis
- Video Editing
- Intelligent Video Databases
  - object classification
- Sports Analysis
- Object-Oriented Video-Coding (MPEG-4)

Definition of Object Segmentation

- It's your turn. Segment this image !

Definition of Object Segmentation

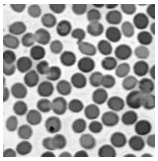
- Central question: what is an object ???
- Semantic problems of object definition:
  - shadows
  - occlusions
  - reflections
  - object status change (parking cars)
  - small movements (waving trees)
  - hierarchical objects (man in car)

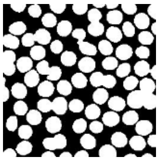
Definition of Object Segmentation

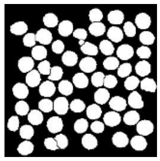
- The object definition and the segmentation algorithm to use depends on the context !
- There is no such thing as *"a general segmentation algorithm"*
- Possible definitions:
  - regions with uniform color → color segmentation
  - regions with uniform texture → texture segmentation
  - regions with uniform motion → motion segmentation

Color Segmentation: Example (1)

- Example medical application:
  - count number of blood cells


input image

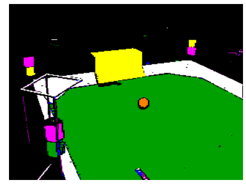

binarized image


separate connected cells (requires further model knowledge)

Color Segmentation: Example (2)

- Preprocessing for special applications
  - colors help to detect objects and markers


input image


segmentation

Images taken from CMVision realtime color segmentation library

Color Segmentation: Example (3)

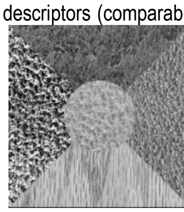
- Color segmentation on natural images:


input image


segmentation

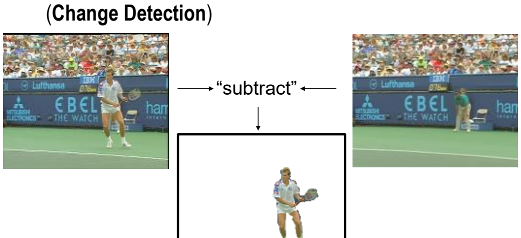
Texture Segmentation

- Group regions based on texture features.
- Usual approach:
  - extract texture descriptors (e.g., Gabor filter coefficients)
  - cluster similar descriptors (comparable to color segmentation)



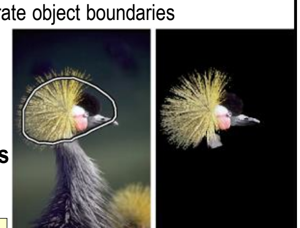
Background Subtraction

- Assume that a pure background image is known
  - detect changes between input image and background image (**Change Detection**)



Manual Segmentation

- Interactive segmentation
  - user controls the segmentation process
  - computer determines accurate object boundaries
- Edge based techniques
  - intelligent scissors
- Region based techniques
  - marker based watershed



Kar-Han Tan, Narendra Ahuja: Selecting Objects with Freehand Sketches

Evaluating the Segmentation Result

* oversegmentation
  – result has ***more*** regions than expected
* undersegmentation
  – result has ***less*** regions than expected

oversegmented

undersegmented

Point detection

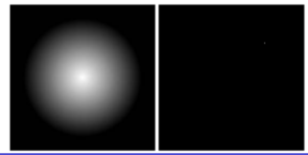
* Look for a point that is different from its neighborhood
* Apply an isolating mask to calculate:

$$R = w_1 z_1 + w_2 z_2 + ... + w_9 z_9$$

* A point is detected at the center of the mask if

$|R| \geq T$ where T is a threshold

−1	−1	−1
−1	8	−1
−1	−1	−1



Line detection

* Masks responding to lines of different orientations:

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

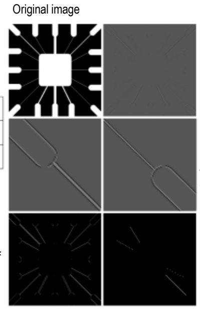
+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

2	−1	−1
−1	2	−1
−1	−1	2

−45°

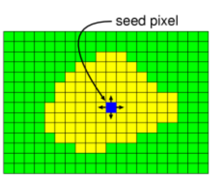


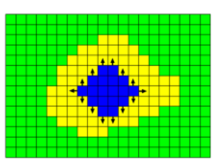
Outline

* Introduction
* Color Segmentation
  – region-growing
  – *region-merging*
  – *watershed*
* Background Subtraction
  – background generation
  – change detection
* Model-based Object Detection
  – graph model

Color Segmentation: Region Growing (1)

* Start a new region with a seed pixel
* Consider neighboring pixels
  – if their color is similar to the mean region color, add pixel to the region
  – continue growing until no more pixels can be added to the region





Color Segmentation: Region Growing (2)

* Implementation
  – input: seed (x_s,y_s), image $I(x,y)$
  – insert (x_s,y_s) into queue of positions Q
  – region color $C=I(x_s,y_s)$

  – while Q not empty
    • extract (x,y) from queue
    • if $\| I(x,y)-C \| < \tau$
      • add neighbors of (x,y) into Q
      • [update region color C with $I(x,y)$]

Color similarity threshold τ

Color Segmentation: Region Growing (3)

19

- Regions have already been defined

$$R = \bigcup_{i=1}^{S} R_i \qquad R_i \cap R_j = \emptyset \qquad i \neq j$$

- We hope $H(R_i) = TRUE \qquad i = 1, 2, ..., S$

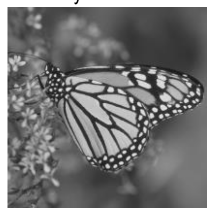
$$H(R_i \cup R_j) = FALSE \quad i \neq j, \quad R_i \ adjacent \ to \ R_j$$

homogeneous

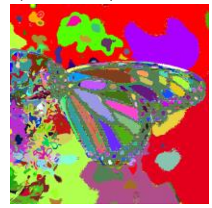
Color Segmentation: Region Growing (4)

20

- Seed pixels can be
  - placed manually, or
  - arbitrarily chosen from the unprocessed pixel



input image



segmentation result

Color Segmentation: Region Growing (5)

21

- Segmentation result is depending on
  - similarity threshold
  - choice of seed pixels
  - order in which seed pixels are processed
  - algorithm implementation (order in which neighboring pixels are processed)

Color Segmentation: Region Merging (1)

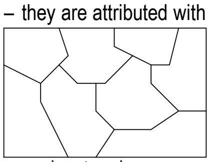
22

- Alternative approach: *Region Merging*
- Start with a set of regions (obtained by another algorithm)
- Consecutively join the two most similar neighboring regions

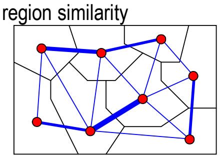
Color Segmentation: Region Merging (2)

23

- Data-structure:
  - neighborhood graph of image regions
  - each node represents a region
  - graph edges denote adjacency
  - they are attributed with the region similarity



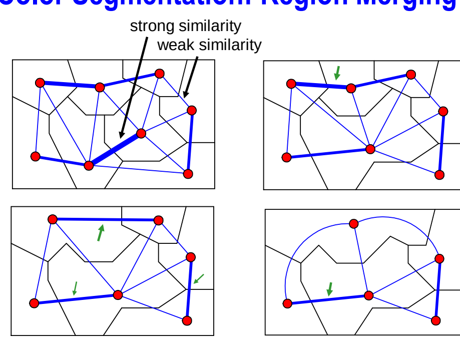
input regions



neighborhood graph

Color Segmentation: Region Merging (3)

24



Color Segmentation: Region Merging (4)

* Several merging criteria are possible
* Simple model:
  – describe region color l_i by average luminance.

$$l_i = \frac{1}{|r_i|} \sum_{(x,y)\in r_i} I(x,y)$$

$I(x,y)$ luminance image
r_i pixels in region

* Merging criterion:
  – **mean:** difference of mean region luminances

$$w_{mean}(i,j) = |l_i - l_j|$$

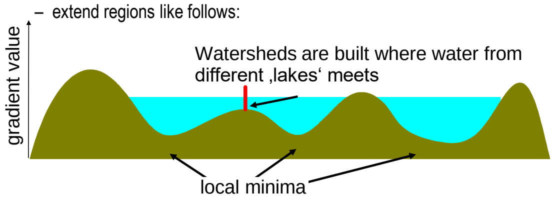
* Difference of mean luminances does not take region size into account

Color Segmentation: Watershed (1)

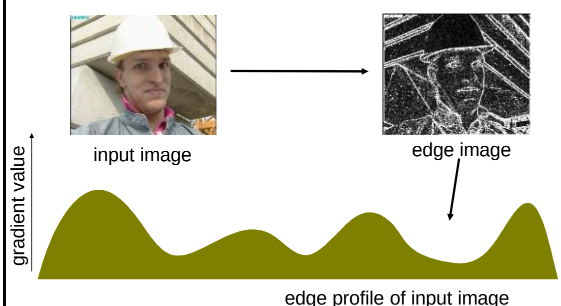
* The *watershed* algorithm
* Fast color segmentation algorithm
* Generally gives oversegmented results on natural images
* Well-defined output
* Often used as preprocessing operation
  – quickly convert a pixel image into a region-level description to speed-up further processing
* Variant: watershed with manually placed markers

Color Segmentation: Watershed (2)

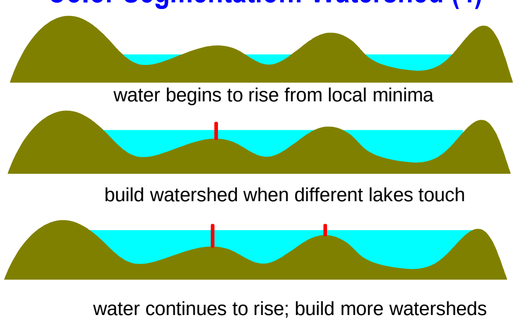
* Algorithm principle:
  – apply gradient filter on input image and work on the resulting edge image
  – search for local minima in edge image and initialize a new region for each minimum
  – extend regions like follows:



Color Segmentation: Watershed (3)

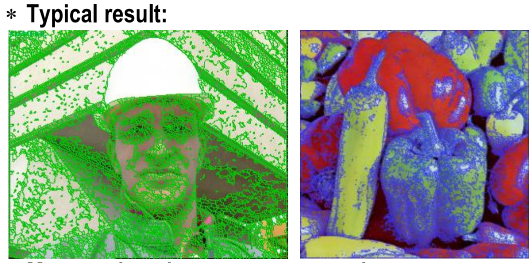


Color Segmentation: Watershed (4)



Color Segmentation: Watershed (5)

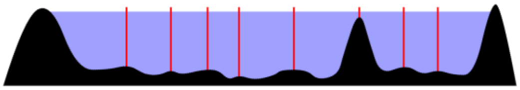
* Typical result:



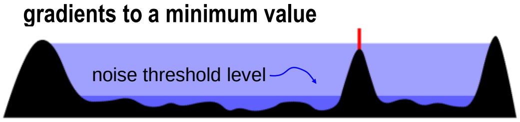
* Many regions because camera noise generates many local minima

Color Segmentation: Watershed (6)
31

* Noise in flat areas generates too many local minima

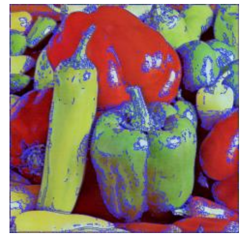


* Small noise regions can be avoided by clipping gradients to a minimum value



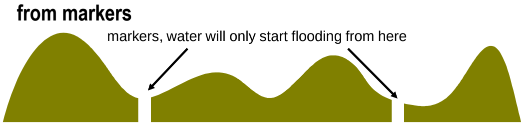
Color Segmentation: Watershed (7)
32

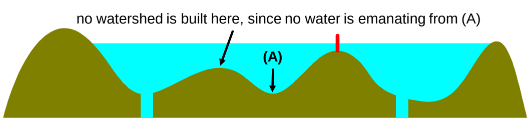
* Result with clipped gradient strength:



Color Segmentation: Watershed (8)
33

* Variant: manual watershed
* Instead of starting to flood from local minima, start flooding from markers





Color Segmentation
34

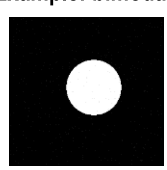
* Summary
  – Region Growing
    – easy implementation, low-quality results
  – Region Merging
    – difficult implementation, flexible control of segmentation process, high-quality results
  – Watershed
    – easy and fast implementation, predictable result, severe over-segmentation
    – no threshold to influence result (except gradient clipping)
  – Result only sufficient for specialized applications

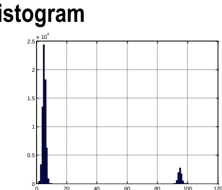
Outline
35

* Introduction
* Color Segmentation
  – region-merging
  – watershed
* **Background Subtraction**
  – background generation
  – change detection
* Model-based Object Detection
  – graph-models

Thresholding
36

* Simple and computationally efficient
* Threshold selection uses intensity information → histogram
* Example: bimodal histogram

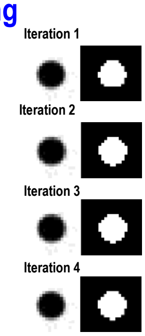




Optimal Thresholding

37

* What value of T will give us the best segmentation?
* Gonzalez and Woods:
  - Initialize T (e.g. halfway between min and max)
  - Iteratively set T=0.5*(μ1+μ2), where μ1, μ2 the mean values of pixels with value larger or smaller than T, respectively
  - Until T converges



Bkg. Subtraction: Principle - (1)

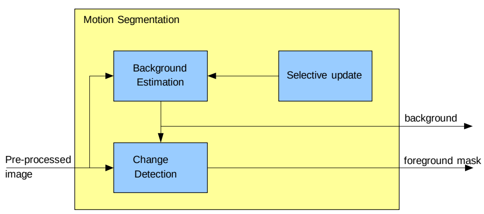
38

* Assumption: background is static
* Input image is compared with background image
  - if the difference is small: background content
  - if the difference is large: foreground object
  - difference is generally not zero due to noise in the sequences
* For some applications, the background image can be captured separately
* For other applications, it must be synthesized from the input sequence

Bkg. Subtraction: Principle (2)

39

* Background image generation
* Change detection



Bkg. Subtraction: Principle (3)

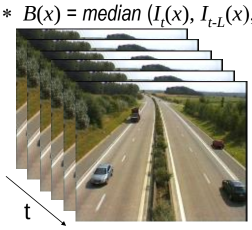
40

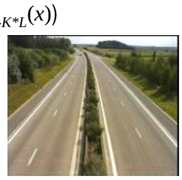
* Background image generation:
  - stack large number of input frames (camera motion must be compensated)
  - different techniques proposed in literature
    - weighted linear filter
    - temporal recursive filter
    - kalman filter
    - *Temporal Median*
    - *Gaussian Mixture Model*

Bkg. Sub: Background Generation (1)

41

* Temporal median filter
  - good performance can be achieved
  - relatively low computational complexity
* $B(x) = median\,(I_t(x), I_{t-L}(x), \ldots, I_{t-K*L}(x))$





background image

Bkg. Sub: Change Detection

42

* We have
  - current input frame I_t
  - background frame I_B
* We denote the image color channels as

$$I_t^R, I_t^G, I_t^B, \quad I_t^Y, I_t^U, I_t^V$$

  - and the vector combining all channels as $\mathbf{I}_t^{YUV}$
* Detect object, e.g., if

$$|I_t^Y - I_B^Y| > \tau$$

* τ is a threshold that depends on the noise level

Bkg. Sub: Difference Metrics

43

* What metric should be used for the image difference?
  – greyscale difference

$$d = |I_t^Y - I_B^Y|$$

  – sum of squared differences of RGB channels

$$d = |I_t^R - I_B^R|^2 + |I_t^G - I_B^G|^2 + |I_t^B - I_B^B|^2$$

  – sum of absolute differences of RGB channels

$$d = |I_t^R - I_B^R| + |I_t^G - I_B^G| + |I_t^B - I_B^B|$$

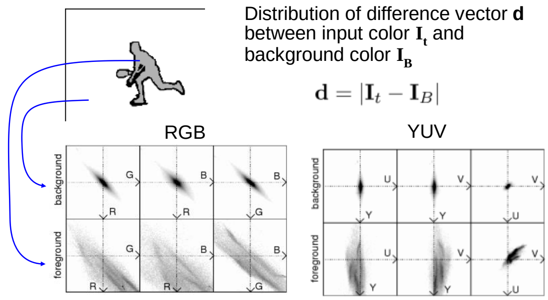
  – ... in YUV color-space
  – ... something else (L*u*v color-space, non-linear difference functions?)

Distribution of Color Differences (1)

44

Distribution of difference vector **d** between input color $\mathbf{I}_t$ and background color $\mathbf{I}_B$

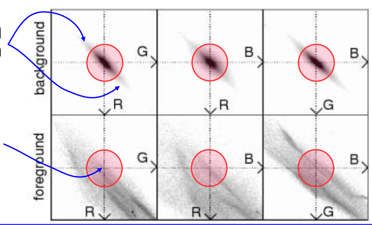
$$\mathbf{d} = |\mathbf{I}_t - \mathbf{I}_B|$$



Distribution of Color Differences (2)

45

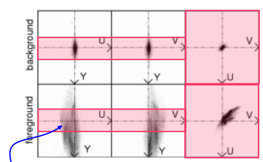
* Each difference function defines a decision boundary
  – here: Euclidean distance defines **sphere**
  – inside sphere: background, outside: foreground



Distribution of Color Differences (3)

46

* Greyscale difference only: $d = |I_t^Y - I_B^Y|$
* Defines slice in Y-dimension
* Better classification could be obtained by integrating color



YUV

many false classifications

example result

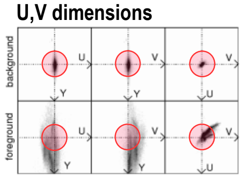
Distribution of Color Differences (4)

47

* Euclidean distance in YUV space

$$d = |I_t^Y - I_B^Y|^2 + |I_t^U - I_B^U|^2 + |I_t^V - I_B^V|^2$$

* Defines sphere
* However, background distribution has lower variance in U,V dimensions



YUV

example result

Background Subtraction

48

* Summary
  – change detection between input frame and background frame
  – independent pixel classification
    • RGB, YUV color difference
    • greyscale difference
    • sum of squared difference
    • Mahalanobis distance
  – classify groups of pixels
    • joint probability has less overlap
    • results are more robust to noise

Outline

* Introduction
* Color Segmentation
  – region-merging
  – watershed
* Background Subtraction
  – background generation
  – change detection
* Model-based Object Detection
  – graph model

GM Algorithm: Principle (1)

* How can we tell the computer what objects we want to extract?

← Find this !

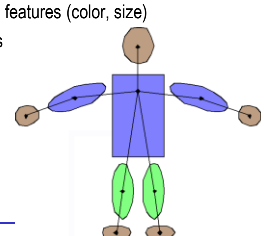
GM Algorithm: Principle (2)

* The same “object” can show in many deformed appearances

* Objects can be partly occluded

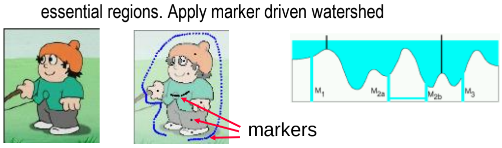
GM Algorithm: Principle (3)

* Idea: Describe objects with graph-based models
* The graph consists of:
  – **nodes** for the essential object regions
  – nodes have **attributes** with region features (color, size)
  – **edges** denote spatial relationships between regions



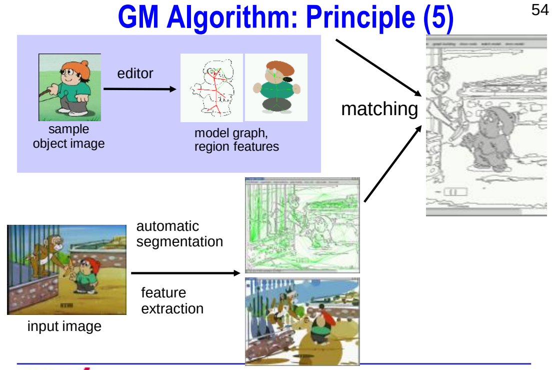
GM Algorithm: Principle (4)

* Generation of the object model:
  – user takes example image and places markers into essential regions. Apply marker driven watershed



  – user specifies spatial relationships between regions that must be fulfilled

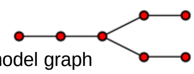
GM Algorithm: Principle (5)

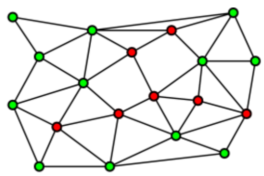


Graph-Model: Graph Matching (1)

55

- **Graph Matching**
  - model graph (V_M, E_M)
  - segmentation graph (V_S, E_S)
  - mapping $\mathcal{M} : V_M \rightarrow V_S$
  - cost function $C(\mathcal{M})$
  - determine $\mathcal{M} = \arg\min_{\mathcal{M}} C(\mathcal{M})$


model graph


segmentation graph

Graph-Model: Graph Matching (2)

56

- **Matching cost function consists of**

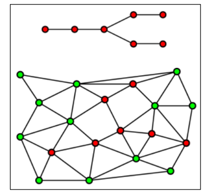
$$C(\mathcal{M}) = \sum_{v \in V_M} C_N(v, \mathcal{M}(v)) + \sum_{(v_1,v_2) \in E_M} C_E(v_1, v_2, \mathcal{M}(v_1), \mathcal{M}(v_2))$$

  - Node costs
    - similarity of regions (color, shape, ...)

$$C_N(v, \mathcal{M}(v))$$

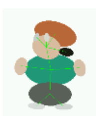
  - Edge cost
    - relations between regions (distances, size ratios)

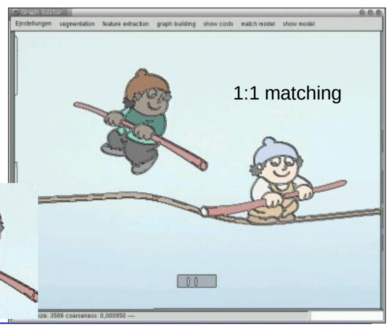
$$C_E(v_1, v_2, \mathcal{M}(v_1), \mathcal{M}(v_2))$$



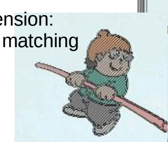
Graph-Model: Result (1)

57


model

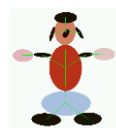

1:1 matching

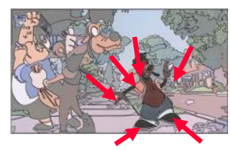
extension:
1:N matching



Graph-Model: Result (3)

58


model


1:1 matching


1:N matching

Graph-Model based Object Detection

59

- **Summary**
  - algorithm to detect general objects
  - object is described with attributed graphs (only tree-shaped)
  - manual definition of model graph
  - automatic color segmentation to obtain segmentation graph
  - efficient graph-matching algorithm to detect model graph in segmentation graph

References (1)

60

- **Color segmentation**
  - B. Dobrin, T. Viero, M. Gabbouj: "Fast watershed algorithms: analysis and extensions", SPIE Nonlinear Image Processing V, 209-220, 1994
  - K. Haris, S. Efstratiadis, N. Maglaveras, A. Katsaggelos: "Hybrid Image Segmentation Using Watersheds and Fast Region Merging", IEEE Transactions on Image Processing, Vol. 7, No. 12, 1684-1699, 1998.
- **Background subtraction**
  - T. Aach, A. Kaup, "Statistical model-based change detection in moving video", Signal Processing 31, pp. 165-180, 1993

References (2)

* **Intelligent Scissors**

  - E.N. Mortensen and W.A. Barrett, “Interactive Segmentation with Intelligent Scissors”, Graphical Models and Image Processing, Vol. 60, No. 5, pp. 349-384, Sept. 1998.

* **Segmentation based on feature clustering**

  - J. Shi, J. Malik, “Normalized Cuts and Image Segmentation”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 8, pp. 888-905, 2000.