Chapter 4
Image Segmentation

Distributed Algorithms
for Image and Video Processing

Content

- Image Segmentation
  - Clustering (K-means)*
  - Histogram based segmentation
  - Region growing*
  - Edge based segmentation
  - Graph partitioning (Graph Cuts) *

(* detailed discussion in exercises)

- Object Segmentation in Videos
  - Camera motion compensation
  - Image transformation
  - Background image / object segmentation
  - Results

Image Segmentation

Definition

- Input: Image with several regions (e.g., objects)
- Unknown: characterization of object regions (pixel positions)
- Characteristics of image regions:
  - connected
  - homogeneous

Clustering (K-means)

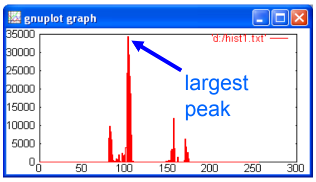
Partition an image into *K* cluster

1. Initialize *K* cluster centers
2. Allocate each pixel to the most similar cluster center
3. Update all cluster centers (average of all pixels assigned to a cluster)
4. Continue with 2., until all cluster centers are stable

→ **Detailed discussion during exercises**

Histogram based segmentation

1. Calculate image histogram
2. Detect maximum peak in histogram
3. Mark all pixels of this color (they define one or more regions)
4. Set selected histogram value to zero
5. As long as the number of selected pixels in the last iteration is larger than threshold, continue with 2.



Region Growing

1. Initialization: Mark all pixels as *undefined, and* region number to first region: *j=1*
2. Repeat, until no more undefined pixel exist:
   – Select next undefined pixel *P*
   – Assign pixel *P* to region *j* (pixel is no longer undefined)
   – Iteratively add neighbor pixel to region, if the difference to pixel *P* is below a threshold
   – Increase *j* by *1* (start with next region)

→ **Detailed discussion during exercises**

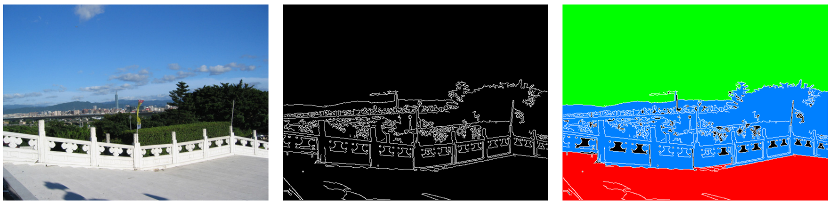
Edge based segmentation

Idea

- Calculate edge image (see *Canny*)
- Edges define the borders of the regions (*floodfill* can be used to identify the pixels of a region)

Problem

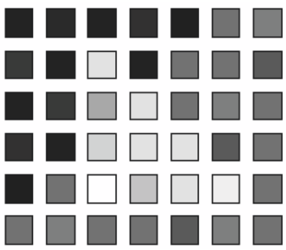
- Regions are merged in case of small gaps in the edges



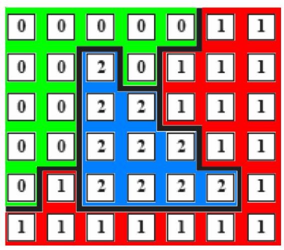
Graph partitioning (Graph Cuts) (1)

Goal

- Classify all pixels of an image (specify object number)



pixels of an image

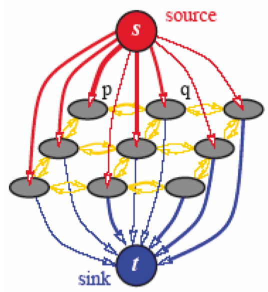


classified objects

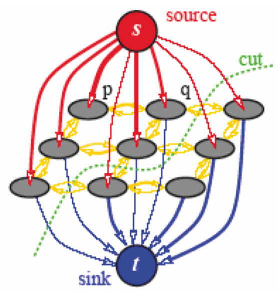
Source: Collins, „Graph Cut Matching In Computer Vision“

Graph partitioning (Graph Cuts) (2)

- The pixels of an image define a graph
- Calculate the maximum flow in a graph



construction of the graph

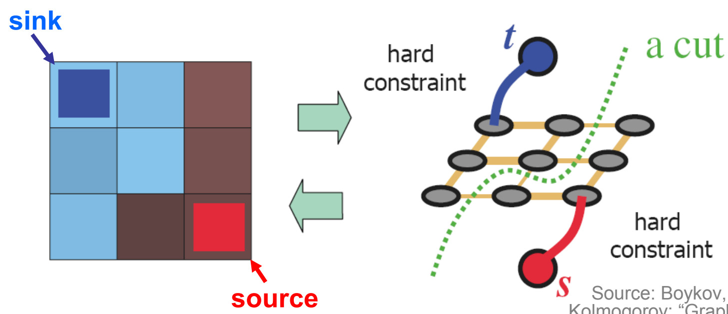


minimal cut of a graph

Source: Collins, „Graph Cut Matching In Computer Vision“

Graph partitioning (Graph Cuts) (3)

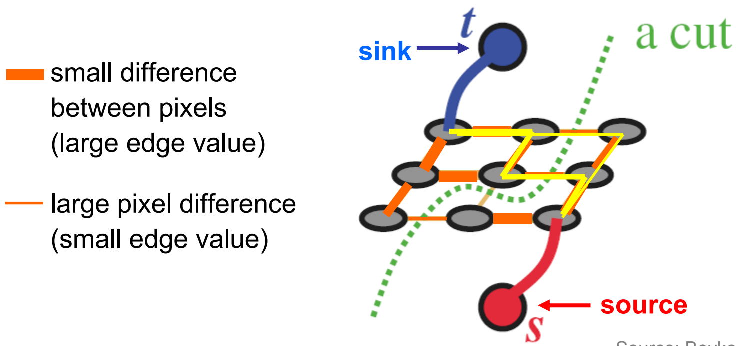
- Initialize nodes *source* and *sink* manually
- If gradient magnitude is large: low value of the graph edges
- If gradient magnitude is low : high value of the graph edges



Source: Boykov, Cremers, Kolmogorov: "Graph Cuts vs. Level Sets", 2006

Graph partitioning (Graph Cuts) (4)

- **Idea**: maximize flow from source to sink in the graph
- Theorem: Maximum flow is equivalent to minimum cut



Source: Boykov, Cremers, Kolmogorov: "Graph Cuts vs. Level Sets", 2006

→ **Detailed discussion during exercises**

Image segmentation in videos (1)

Goal

- Segment foreground (moving) object

Assumptions

- Motion of object and image background differs
- Luminance or color difference between object and background
- At least 50 percent of the pixels describe image background
- Continuous object motion

Image segmentation in videos (2)

Segmentation algorithm

1. Align image background of all frames:
   – Calculate the parameters of the camera model between all frames of a shot
   – Transform all frames into the same coordinate system
2. Generate background image
3. Segment moving objects by comparing the transformed frames to the background image

Camera motion compensation (1)

Approach

- The parameters of the camera model $T_{i,i+1}$ between two adjacent frames *i* and *i+1* are known
- Unknown: All other frame combinations: $T_{i,j}$

- Both images *i* and *j* are aligned if image *i* is transformed with $T_{i,j}$
- A reference frame *j* is selected
- To create a background image, all images *i* of a shot are transformed with $T_{i,j}$

Camera motion compensation (2)

Calculation of camera model parameters $T_{i,j}$ ($i<j$)

1. Transformation $T_{i,i+1}$ is given
   Select coordinate (x, y) in image i and transform it with $T_{i,i+1}$ to new position (x',y')
2. (x', y') is transformed with $T_{i+1,i+2}$ and gives position (x'', y'') in image $i + 2$.
3. The motion vector from (x, y) to (x'', y'') describes the motion of a background pixel from frame i to $i+2$

Camera motion compensation (3)

Calculation of camera model parameters $T_{i,j}$ ($i<j$)

4. We get 4 motion vectors from frame i to $i+2$ if we select 4 coordinates and transform them with $T_{i,i+1}$ and in a second step with $T_{i+1,i+2}$
5. The 4 motion vectors are used to calculate the transformation $T_{i,i+2}$
6. This technique is used to calculate any transformations $T_{i,j}$ from image i to j ($i< j$)

Camera motion compensation (4)

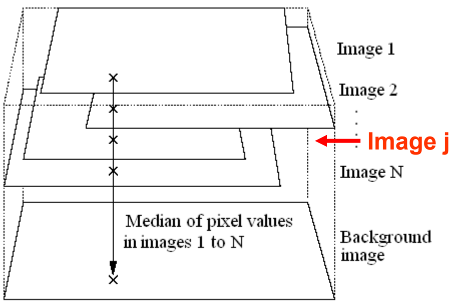
Calculation of camera model parameters $T_{i,j}$ ($i>j$)

- We can easily calculate the transformation $T_{j,i}$ (j<i)
  Unknown: inverse transformation $T_{i,j}$
- 4 motion vectors from *j* to *i* are calculated
- We change the direction of the motion vectors (switch start and end point)
- The new motion vectors are used to solve the linear system of equations and derive the parameters of the inverse transformation $T_{i,j}$

Transformation of an image (1)

Idea

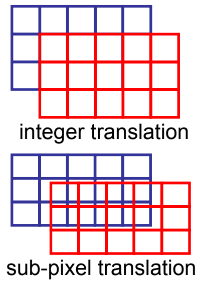
- Select one frame *j* as reference image
- Align all images to the reference image (use transformation $T_{i,j}$)
- The background of all images is aligned after the transformation



Transformation of an image (2)

Generation of a background image

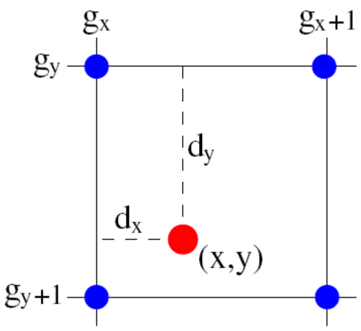
- For each pixel in the background image, one pixel value from the transformed images is selected
- The inverse transformation $T_{j,i}$ – starting with pixel position (x', y') in the background image – defines the position (x, y) in the original image
- The luminance value at position (x, y) in the original image is assigned to each pixel at position (x', y') in the transformed image I'
- In most cases, the position (x, y) is not an integer value but defines a position between 4 adjacent pixels



Transformation of an image (3)

Linear interpolation

- g_x and g_y: integer value of *x and y*
- Remainder:
  $d_x := x - g_x$
  $d_y := y - g_y$
- Luminance I' in transformed image



$$I'(x',y') = [(1-d_x)\cdot I(g_x,g_y) \quad + d_x \cdot I(g_x+1,g_y) \quad]\cdot(1-d_y) + [(1-d_x)\cdot I(g_x,g_y+1) + d_x \cdot I(g_x+1,g_y+1)]\cdot d_y$$

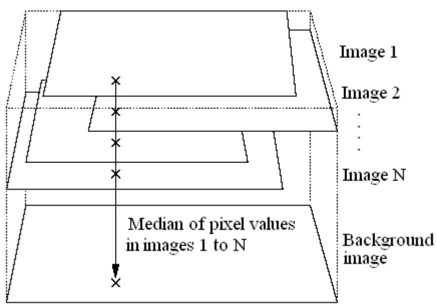
Transformation of an image (4)

Characteristics of linear interpolation

- Between 1 and 4 pixels define the value of the new pixel
- A translation of *0,5* pixels maximizes the blurring
- Linear interpolation increases the precision and enables a better segmentation of objects
- Linear interpolation should be avoided if high resolution panoramic images are created (blurring, erroneous colors)
- To avoid blurring, only the nearest pixel is used (avoid interpolation)
  → disadvantage: integer precision of the translation

Generation of a background image (1)

- Moving objects should not be visible in background (panoramic) images
- Transform all *N* frames of a shot:
  - Which frame defines the color of the background image?
  - The median of all pixels at a position is a good heuristic



Generation of a background image (2)

Calculation of the median

- Use aligned frames and add pixels at one image position *(x,y)* to a list
- Sort list and select element in the middle of the list (median)
  → high computational effort for each pixel: *O (N log N)*

Improvements

- Create a histogram of all pixel values at one position
- Calculate cumulative histogram (summarize pixel values)
- The median is defined as index position:
  index value > ½ number of all pixels

Generation of a background image (3)

Assumption

- At least half of all pixels describe image background
  → The median is a good heuristic to classify a background pixel

Problem

- Slow moving objects
- In more than 50 percent of the frames, an object is at the same position
  → incorrect regions in the background image

Generation of a background image (4)

Example: Background image (based on average pixel values)

original video

average of last 10 frames

Generation of a background image (5)

Example: Background image based on median

Error caused by slow object motion

Generation of a background image (6)

Improved selection of pixels for background images

- Comparison of two aligned frames
  → Object motion causes image differences in two regions
- Estimate position and size of an object by calculating the center of gravity (S_x , S_y) of all difference pixels *D(x,y)*:

$$S_x = \frac{1}{\sum_{x,y} D(x,y)} \sum_{x,y} x \cdot D(x,y) \qquad S_y = \frac{1}{\sum_{x,y} D(x,y)} \sum_{x,y} y \cdot D(x,y)$$

- (S_x , S_y) specifies the center of a rectangle; its size depends on the variance of the pixel positions

Generation of a background image (7)

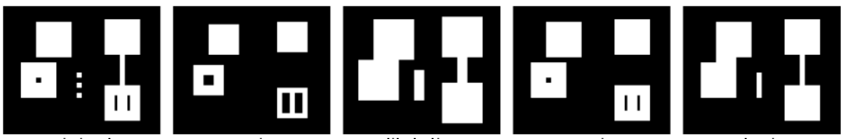
Moving object segmentation (1)

- The median removes moving objects
- Object segmentation is done by comparing the transformed images to the background image
- The position, size and shape of the object in the original frame is calculated by applying the inverse transformation
- The largest object is selected
- Morphological operators (*opening*, *closing*) are used to reduce errors at image boundaries
- Edges near object borders can be used to improve the accuracy of the segmentation

Moving object segmentation (2)

Morphological operators: *Opening* / *Closing*

$$\begin{aligned}
Dilatation: \quad D_B(I(x)) &= \max \{I(x+r) \mid r \in B\}, \\
Erosion: \quad E_B(I(x)) &= \min \{I(x+r) \mid r \in B\}. \\
Opening: \quad O_B(x) &= D_B\left[E_B\left(I(x)\right)\right], \\
Closing: \quad C_B(x) &= E_B\left[D_B\left(I(x)\right)\right].
\end{aligned}$$



original erosion dilatation opening closing

- Opening: small objects and thin lines are removed
- Closing: holes within and between objects are filled

Moving object segmentation (3)

difference image

morphological operators

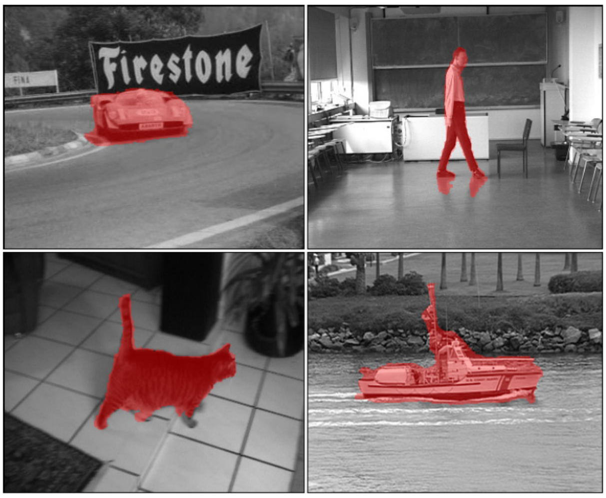
edges

Results

Typical errors

- Very large objects
- Slow object motion
- Object moves to the camera (pixel position of object does not change)
- No characteristic image features in background
- Shadow of object
- Luminance of object and background pixels are similar

Results: Example of segmented objects



Questions ?