Segmentation

Definition 1:

Segmentation = finding outline of object or region in image

Definition 2:

Segmentation = grouping of pixels into regions such that:

- Pixels in each region have a common property
- Pixels in adjacent regions do not share this property
- Exclusive Partitioning: Pi intersect Pj = {} for all i not equal to j
- Exhaustive Partitioning: Union of Pi = entire image

Definition 2:
Segmentation = grouping of pixels into regions such that:
Pixels in each region have a common property
Pixels in adjacent regions do not share this property
Exclusive Partitioning: Pi intersect Pj = {} for all i not equal to j
Exhaustive Partitioning: Union of Pi = entire image

Common Property:

Easy to test:

- Same (or similar) color
- Same grey value
- Brightness above some threshold
- Black versus white

Deep Learned:

"Typical" object appearance

"Semantic Segmentation"

common property =

same type of object, e.g. cars

"Instance Segmentation"

common property = object instance,
e.g. a specific car

Definition 2:
Segmentation = grouping of pixels into regions such that:
Pixels in each region have a common property
Pixels in adjacent regions do not share this property
Exclusive Partitioning: Pi intersect Pj = {} for all i not equal to j
Exhaustive Partitioning: Union of Pi = entire image

Common Property:

Same color
Same grey value
Brightness above some threshold
Black versus white

Segmentation Pipeline:

Color image -> e.g., R+G+B/3
Greyscale image ->
Segmentation algorithm e.g., thresholding
-> Binary image ->
Boundary Following Algorithm

Segmentation Methods

1. Absolute Thresholding

   Algorithm:

   1) Determine by inspection of the grey value histogram of the image which threshold T separates bright and dark pixels
   2) Scan image and assign foreground and background labels to objects/regions

   Disadvantage: Does not work well if

      object is illuminated unevenly

      MRI scanner has a bias

Segmentation Methods

2. Percentile Method

Assumption: Single (bright) object occupies p% of image

Algorithm:

1) n = number of pixels
2) m = number of object pixels = n ∗ p%
3) Scan image and find m brightest pixels and assign them to belong to desired object

Segmentation Methods

3. Mode Method

Assumptions:

n objects and background have different and relatively uniform grey values; objects are not "very small"

<u>Algorithm for 1 Object:</u>

1) Find local maxima at some minimum distance apart, e.g., maximum at g_i & g_j
2) g_k = minimum between g_i & g_j
3) Peakiness $p(i,j,k) = [\min \{H(g_i), H(g_j)\}] / H(g_k)$
4) Use (i,j,k) with highest peakiness p: $T = g_k$

<u>Algorithm for n Objects:</u>

Move through histogram to find n valleys based on their peakiness

Segmentation Methods

4. Adaptive Thresholding

Adapt T to image region

Algorithm: Create different thresholds for different image regions

Segmentation Methods

5. Iterative Threshold Method

- T = average grey value in image
- Partition image into sets of pixels with grey value above and below T
- Compute the average grey value in each set
- T = average of these averages
- Repeat procedure until averages do not change much anymore

What if you segmented the object of interest into several regions instead of one region?

Merging of Adjacent Regions

Merging: Combine adjacent regions with similar characteristics:

Mean of grey values of region 1: $\mu_1 = 1/n_1 \Sigma_{\text{i in region1}} g_i$

Mean of grey values of region 2: $\mu_2 = 1/n_2 \Sigma_{\text{i in region2}} g_i$

Method 1: Merge based on means

If $|\mu_1 - \mu_2| <$ threshold, merge regions.

Method 2: Merge based on likelihood ratio (use standard deviations)

$\sigma_1^{\,2} = 1/\, n_1 \Sigma_{\text{i in region1}} (g_i - \mu_1)^2$

$\sigma_2^{\,2} = 1/\, n_2 \Sigma_{\text{i in region2}} (g_i - \mu_2)^2$

Assume the regions in an image have constant grey values that are corrupted by independent additive zero-mean Gaussian noise.

Merging of Adjacent Regions

Method 2:

Hypothesis H_0: Both regions belong to the same object => gray values drawn from same distribution $N(\mu_0 , \sigma_0^2)$

Hypothesis H_1: The regions belong to different objects => gray values drawn from different distributions $N(\mu_1 , \sigma_1^2)$, $N(\mu_2 , \sigma_2^2)$

$P(g_1, g_2, \ldots, g_{n1+n2} | H_0) = \Pi_{i=1}^{n1+n2} p(g_i | H_0)$

$P(g_1, g_2, \ldots, g_{n1+n2} | H_1) =$

Merging of Adjacent Regions

Method 2:

The likelihood ratio is defined as the ratio of probability densities under the two hypotheses:

$L = P(g_1, g_2, \ldots, g_{n1+n2} | H_1) \; / \; P(g_1, g_2, \ldots, g_{n1+n2} | H_0) =$

$\sigma_0^{n1+n2} \; / \; (\sigma_1^{n1} \, \sigma_2^{n2})$

If $L <$ threshold, then it is highly likely that there is only one region => merge.

Boundary Following Algorithm

Goal: Given a segmented region (or object), find the pixels on the closed boundary of the region (or object).

Tool: Neighborhood definitions N_4

N_8

Notation: R=region, $\underline{R}$=background,

b=pixel at boundary (outside object/region)

c= “current pixel” inside object/region on boundary

Boundary Following Algorithm

1. Find start pixel s (through scanning binary image), c=s
2. Find N_4–neighbor b in R̲ to west of s
3. Among N_8–neighbors of c, starting with b, search clockwise for first n_i in R
4. $c = n_i$, $b = n_{i-1}$
5. Repeat steps 3 & 4 until c=s

Comparison of two boundaries (or curves): using the Hausdorff Distance

Idea: Two boundaries have a small Hausdorff distance if every pixel of either boundary is close to some pixel of the other boundary.

The Hausdorff distance is the largest distance between points on two curves when an adversary can select a point on one of the curves (goal to maximize the distance) while you are trying to minimize the distance from that chosen point to a point on the other curve

$$\max_{x \text{ in } X} \min_{y \text{ in } Y} \text{Euclidean distance}(x,y)$$

Comparison of two boundaries (or curves): Hausdorff Distance

Hausdorff distance between curves X and Y =

$$\max \left\{ \max_{x \text{ in } X} \min_{y \text{ in } Y} \text{Euclidean distance}(x,y),\ \max_{y \text{ in } Y} \min_{x \text{ in } X} \text{Euclidean distance}(x,y) \right\}$$