Medical Image Analysis

CS 593 / 791

Computer Science and Electrical Engineering Dept.
West Virginia University

2nd March 2007

Outline

1. Introduction
2. The min cut problem
3. Segmentation using graph cuts

Outline

1. Introduction
2. The min cut problem
3. Segmentation using graph cuts

Graph Cuts

- Reduce the problem of image segmentation to the problem of graph cuts.
- We can then use existing algorithms for graph cuts to perform segmentation.

We will present an **interactive** technique for segmenting an object from the background (only 2 regions). The user must specify some hard constraints : some pixels which are known to belong to the object, and some pixels which are known to belong to the background.

Graph

Let $G = \{V, E\}$ be a graph which consists of nodes V and edges E.
In general, the edges may be

- directed or undirected
- weighted or unweighted

The algorithm presented here uses a weighted, undirected graph.
Additionally, all weights are assumed to be positive.

Outline

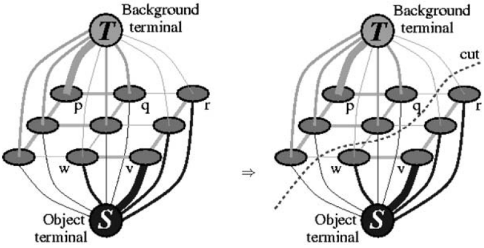
1 Introduction

2 The min cut problem

3 Segmentation using graph cuts

The min cut problem

- Let the set of nodes include 2 special nodes (terminal nodes) labeled S and T.
- An S/T cut (or just ”cut”) is a subset C of E such that there is no path between S and T when C is removed from E.
- The cost of a cut is the sum of all of the weights of the edges in C.
- The min-cut problem is to find the cut with minimum weight.



(b) Graph. (c) Cut.

Outline

1 Introduction

2 The min cut problem

3 Segmentation using graph cuts

Overview

- Convert an input to the segmentation problem to an input to the graph cuts problem:
  - Convert image and initialization into a graph, G.
- Find minimum cost cut, C, of the graph G.
- Convert the output C, into a segmentation.

Create the graph, G

Convert the image into an undirected, weighted graph.

- Create a node $v \in V$ for each pixel in the image.
- Pick a neighborhood system to use
  - for example : 4 or 8 neighbor system in 2D
  - for example : 6 or 26 neighbor system in 3D
- For each pair of neighbors create an edge (n-link).
- Set the weight for each n-link to $B_{p,q}$ (pixel similarity).

$B_{p,q}$ is a measure of the similarity of image intensities at pixels p and q.

$$B_{p,q} = c\exp(-\frac{(I_p - I_q)^2}{2\sigma^2})\frac{1}{dist(p,q)}$$

Create the graph, G

Incorporate the initialization - a set of pixels O and B in object and background.

- Create 2 terminal nodes, S and T.
  - S node represents the ”object”
  - T node represents the ”background”
- For each image vertex, p, create an edge to S (”obj terminal”)
  - If $p \in O$ then $w_{p,S} = K$ (a large constant)
  - If $p \in B$ then $w_{p,S} = 0$
  - Else, $w_{p,S} = \lambda R_p("bkg")$
- For each image vertex, p, create an edge to T (”bkg terminal”)
  - If $p \in O$ then $w_{p,T} = 0$
  - If $p \in B$ then $w_{p,T} = K$
  - Else, $w_{p,S} = \lambda R_p("obj")$

$R_p(A)$ is a region-based function reflecting how well the intensity at p fits into ”object” or ”background” initialization.

Find min cut

Known algorithms for finding the min cut:

- Ford-Fulkerson method (Edmunds-Karp algorithm) - $O(VE^2)$

The 'maxflow' algorithm optimized for graph cuts on grid graphs was presented in:
"An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Computer Vision.", Yuri Boykov and Vladimir Kolmogorov. In IEEE Transactions on Pattern Analysis and Machine Intelligence, September 2004.

- Outperforms other algorithms
- Seems to scale linear in the number of pixels.

The output

The function $A_p(C)$ returns the region of pixel, p, given the cut, C.

$$A_p(C) = \left\{ \begin{array}{ll} \text{"obj"} & \text{if } \{p, T\} \in C \\ \text{"bkg"} & \text{if } \{p, S\} \in C \end{array} \right.$$