MIC : Project Report
Image Segmentation using Graph cuts

Motivation:

Segmentation is an important part of Image Analysis. The goal of segmentation is to simplify an image into something that is more meaningful and easier to analyse. Image Segmentation is used in Image Recognition and Image feature extraction. Many real world applications can strongly benefit from algorithms that can segment out objects in images by finding their precise boundaries. So, we decided to implement a technique of segmentation as project in this course.

Introduction:

Finding the *max-flow* given a graph is an age old problem for which many polynomial time algorithms are known. In this technique, we reduce the problem of segmenting an image into foreground and background to a *max-flow* (*min-cut*) problem which can then be solved efficiently in polynomial time. This project *i.e* Segmentation using Graph Cuts is a general approach for segmenting images. In most of the cases, this approach gives best solution. In our case, segmentation is interactive *i.e* based on the seeds provided by the user.

Segmentation Technique:

To segment an image we create a graph with nodes corresponding to pixels of the image. There are two additional terminal nodes : an “object” terminal (a source) and a “background” terminal (a sink). The source is connected by edges to all nodes identified as object seeds and the sink is connected to all background seeds with infinite cost. The remaining nodes are connected to the source and sink based on the probability calculated using the histograms constructed with the user provided seeds.

Pairs of neighboring pixels are connected by weighted edges. In our implementation, undirected links are constructed between neighboring pixels with a cost which is a decreasing function of the difference of their intensities.

We draw the boundary between the object and the background by finding the minimum cost cut on this graph. A cut is a subset of edges that separates the source from the sink. The cost of the cut is the sum of its edge costs.

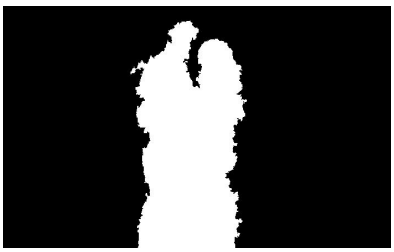
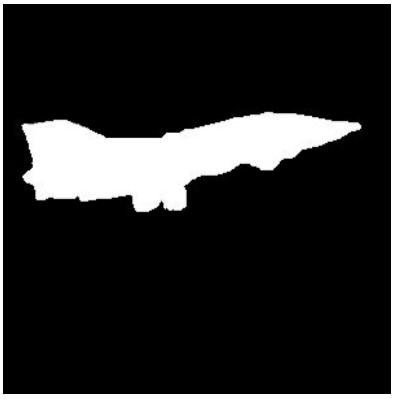
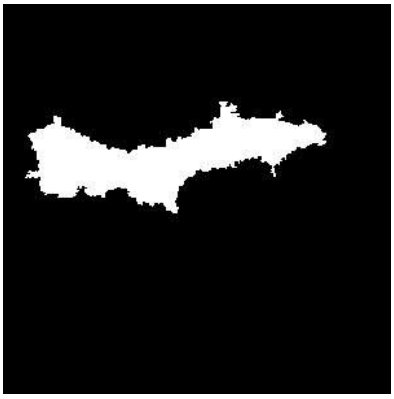
Locations with high intensity gradient between neighbouring pixels are attractive choices for the optimal segmentation boundary. The minimum cut can be computed

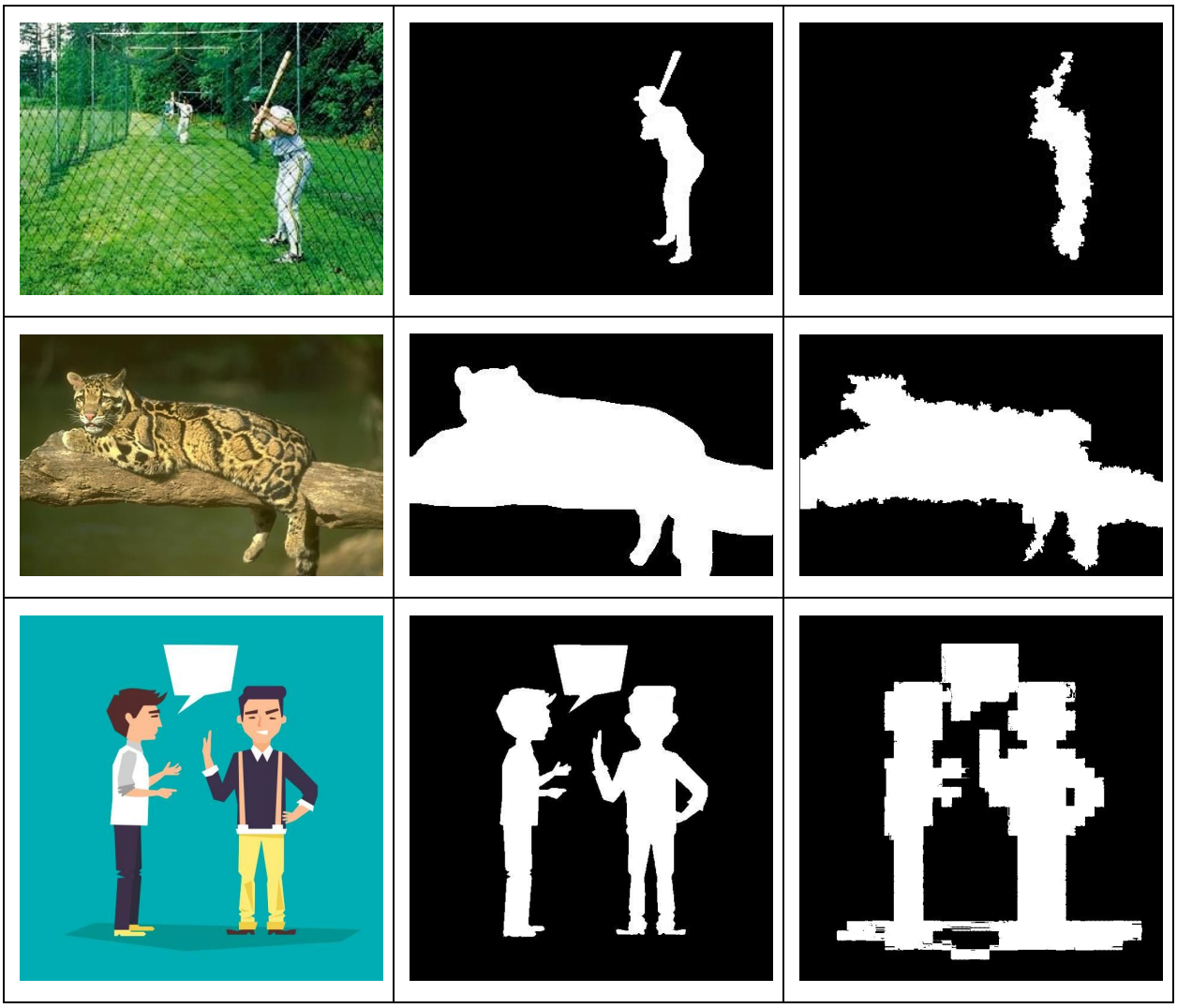
exactly in polynomial time using well known algorithms for two terminal graph cuts such as max-flow.

Working:

1. The libraries required for python 3.0 are numpy, cv2, matplotlib, networkx and pillow.
2. Three files are written as a part of this assignment: *project.py*, *foreground.m* and *background.m*.
3. Name the input file as *test.jpg* and place it in same folder as above three files.
4. Now, run *foreground.m* file in Matlab. An interface opens showing given input image where you have to draw lines thrice on object of interest in the image.
5. Then close the images and run *background.m* file in Matlab. Similarly, draw three lines with cursor on background of object in the given image.
6. Close the images. Now, two new files named *foreground.jpg* and *background.jpg* are created in the folder.
7. Now run *project.py* file in terminal which creates required output image named *final.jpg* in the same folder.

Results:

Original Image	Ground truth	Our Segmentation
		
		



References:

1.http://www.csd.uwo.ca/~yuri/Papers/miccai00.pdf