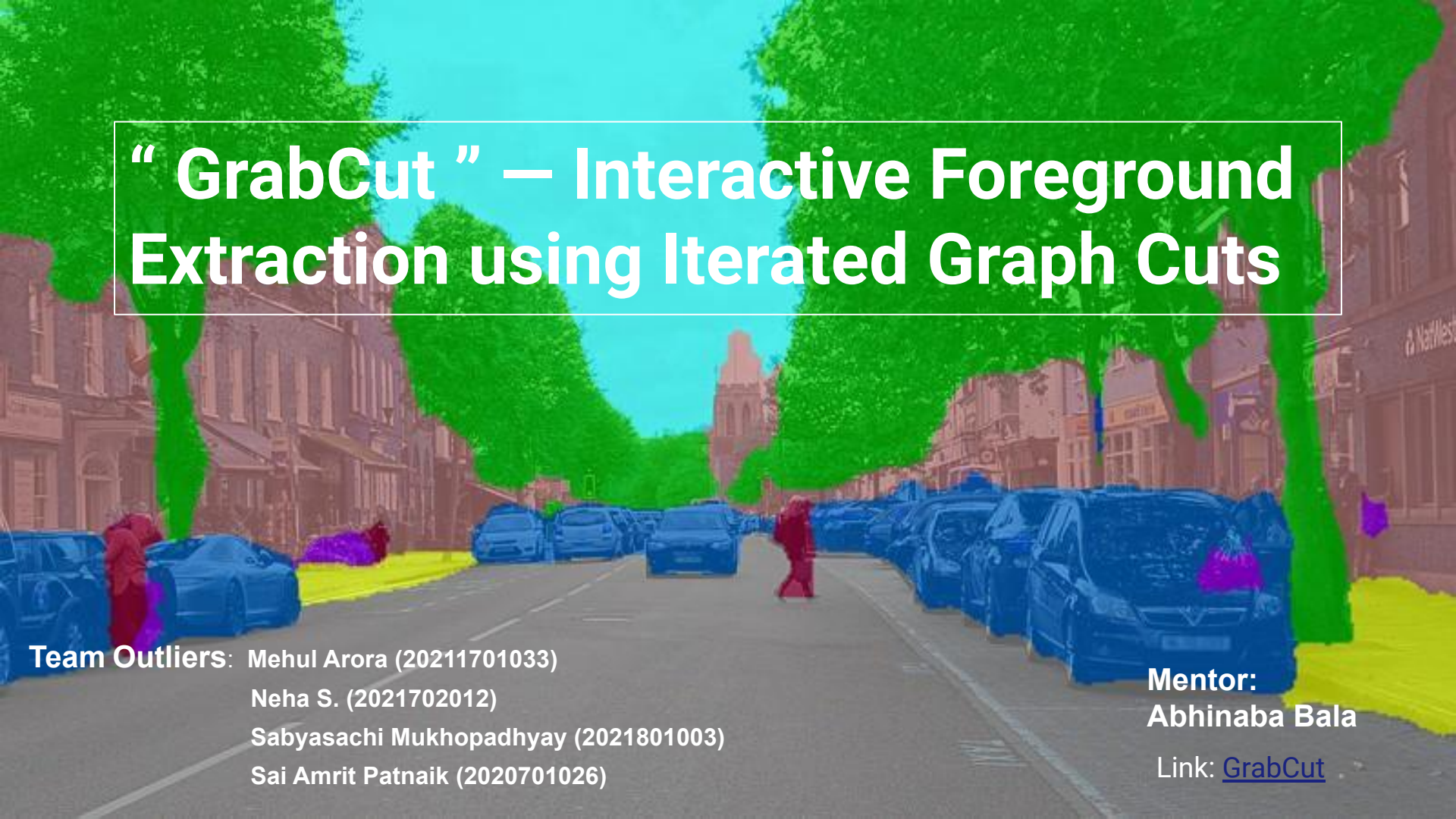
“ GrabCut ” — Interactive Foreground Extraction using Iterated Graph Cuts



Team Outliers: Mehul Arora (20211701033)
Neha S. (2021702012)
Sabyasachi Mukhopadhyay (2021801003)
Sai Amrit Patnaik (2020701026)

Mentor:
Abhinaba Bala

Link: GrabCut

1. Problem Statement
2. Objective
3. Graph Cut
4. Grab Cut
5. Scope and Deliverables
6. Midterm Progress
7. Future Works
8. References

Project Details

Problem Statement

The problem of efficient, interactive foreground/background segmentation in still images is of great practical importance in image editing.

All previous approaches demand considerable user interaction and yet the quality of segmentation obtained is poor, mostly because most of these approaches use texture or edges information.

These approaches also don’t addresses the problem of extraction of a foreground object in a complex environment whose background cannot be trivially subtracted.

Objective

The objective of the project is to implement GrabCut which is a powerful and robust algorithm that builds over Graph Cut algorithm(proposed by Boykov and Jolly).

The user is requested to mark a single rectangle around the object, defining the outer part of the rectangle as definite background, and the inner part of it as an unknown combination of the object (foreground) and some background.

These constraints are used as initial solution to the problem, leading to an iterative method which in conclusion aims to assign each pixel in the image its label - Background or Foreground.

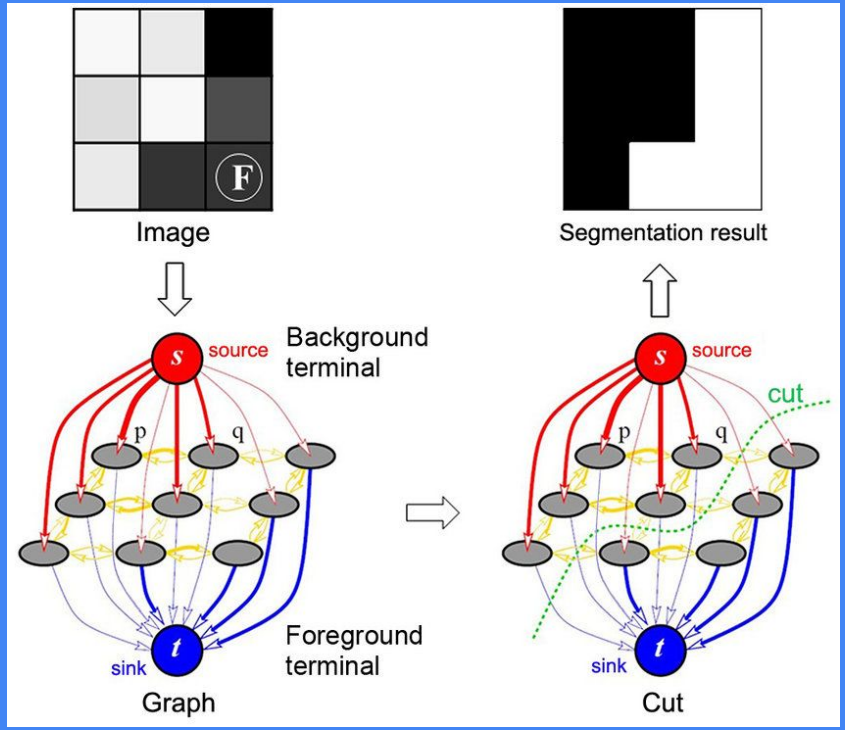
Graph Cut

- The framework of Graph cut consists of 2 parts: First, a network of flow graphs is built based on the input image. After that a max-flow algorithm is run on the graph in order to find the min-cut, which produces the optimal segmentation.
- The problem can be transferred to energy minimization. Its minimum should correspond to a good segmentation, as it is guided by both observed foreground and background grey-level histograms.
- The cost function to optimize is:

$$\arg\min_{w_{1\dots N}} \sum_{n=1}^{N} U_n(w_n) + \sum_{(m,n)\in\mathcal{C}} P_{mn}(w_m, w_n),$$

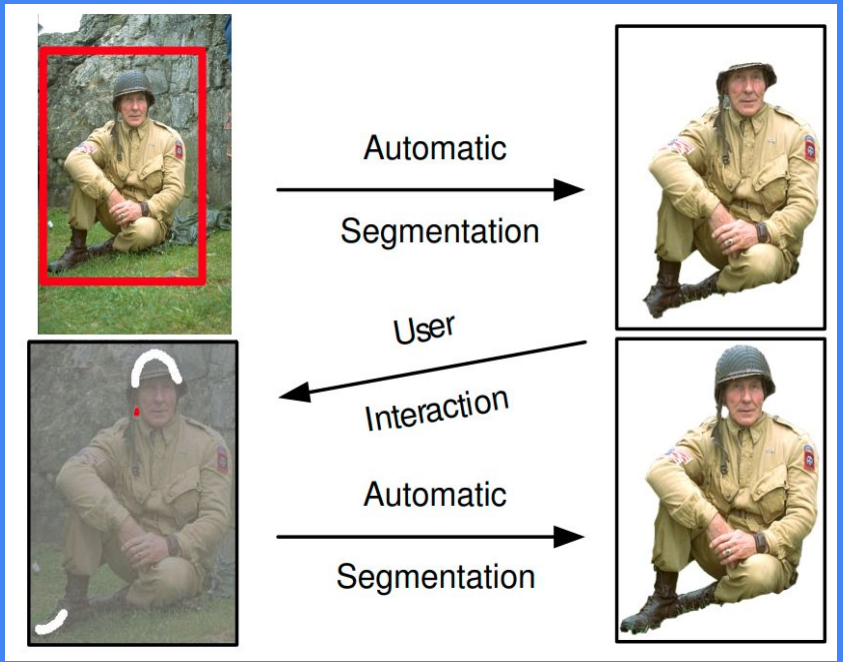
Unary terms (compatability of data with label y)

Pairwise terms (compatability of neighboring labels)



GrabCut

- Grabcut is a more powerful, iterative version of the optimization of Graph-cut. This substantially simplifies the user interaction needed for a given quality of result.
- In Grabcut, the monochrome image model is replaced for colour by a Gaussian Mixture Model (GMM) in place of histograms
- The one-shot minimum cut estimation algorithm is replaced by an iterative procedure that alternates between estimation and parameter learning and the demands on the interactive user are relaxed by allowing incomplete labelling.



GrabCut Algorithm

Initialisation

- User initialises trimap T by supplying only T_B. The foreground is set to $T_F = \emptyset$; $T_U = \overline{T}_B$, complement of the background.
- Initialise $\alpha_n = 0$ for $n \in T_B$ and $\alpha_n = 1$ for $n \in T_U$.
- Background and foreground GMMs initialised from sets $\alpha_n = 0$ and $\alpha_n = 1$ respectively.

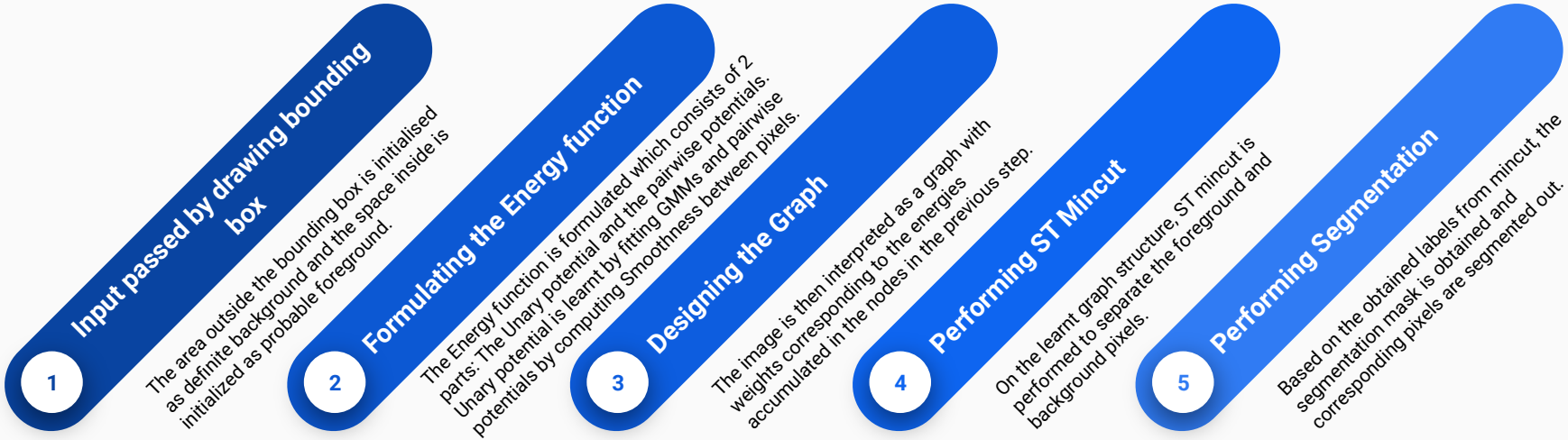
Iterative minimisation

1. *Assign GMM components to pixels:* for each n in T_U,
$$k_n := \arg\min_{k_n} D_n(\alpha_n, k_n, \theta, z_n).$$
2. *Learn GMM parameters* from data $\mathbf{z}$:
$$\underline{\theta} := \arg\min_{\underline{\theta}} U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z})$$
3. *Estimate segmentation:* use min cut to solve:
$$\min_{\{\alpha_n:\, n \in T_U\}} \min_{\mathbf{k}} \mathbf{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}).$$
4. Repeat from step 1, until convergence.
5. Apply border matting (section 4).

User editing

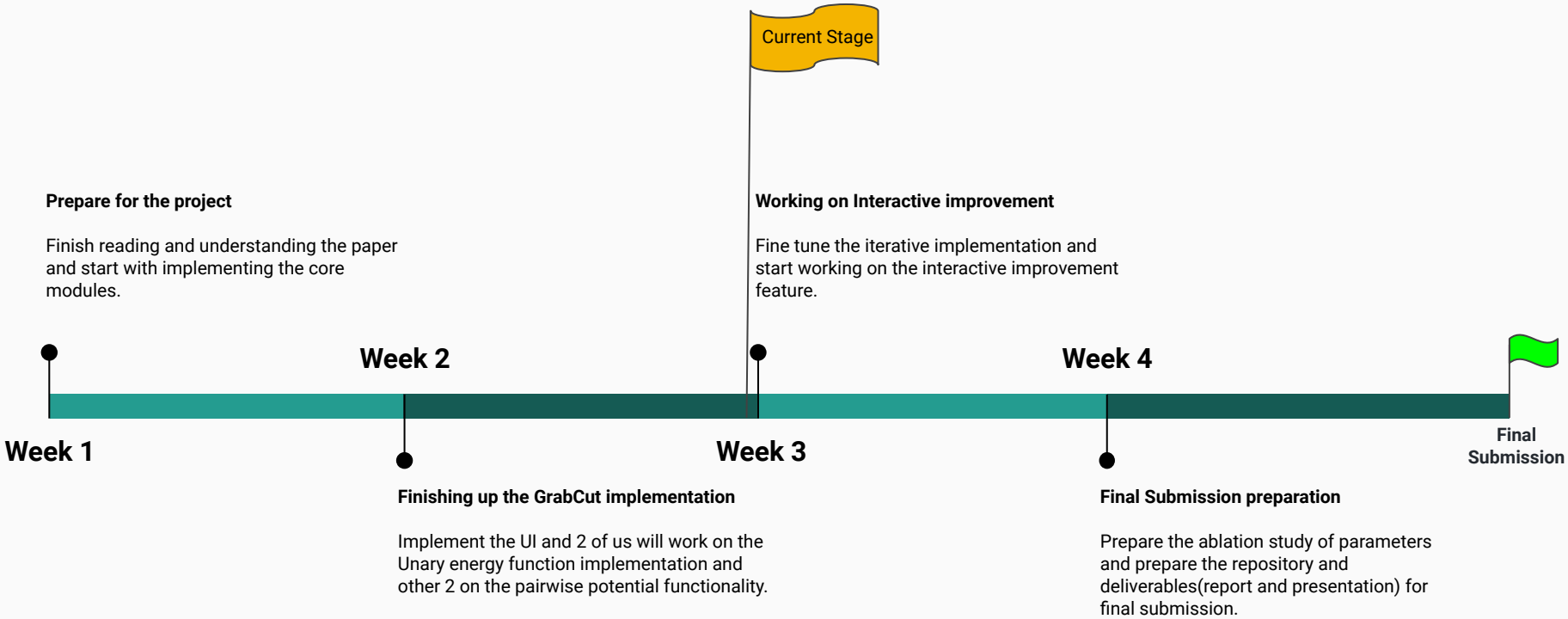
- *Edit:* fix some pixels either to $\alpha_n = 0$ (background brush) or $\alpha_n = 1$ (foreground brush); update trimap T accordingly. Perform step 3 above, just once.
- *Refine operation:* [optional] perform entire iterative minimisation algorithm.

Flow Chart


1
Input passed by drawing bounding box
The area outside the bounding box is initialised as definite background and the space inside is initialized as probable foreground.
2
Formulating the Energy function
The Energy function is formulated which consists of 2 parts: The Unary potential and the pairwise potentials. Unary potential is learnt by fitting GMMs and pairwise potentials by computing Smoothness between pixels.
3
Designing the Graph
The image is then interpreted as a graph with weights corresponding to the energies accumulated in the nodes in the previous step.
4
Performing ST Mincut
On the learnt graph structure, ST mincut is performed to separate the foreground and background pixels.
5
Performing Segmentation
Based on the obtained labels from mincut, the segmentation mask is obtained and corresponding pixels are segmented out.

Deliverables

- A fully working implementation of the iterative GrabCut Algorithm implemented from scratch.
- Implementation for the components of the energy function and optimization procedure.
- An interactive user interface that lets the user provide initial input by drawing bounding box.
- Generate and display the segmented output.
- Implement the interactive user editing feature to allow the user to improve segmentation.
- A detailed ablation study on various parameters of the algorithm and their effect on the output.
- Release the code open sourced and Presentation and description of the project along with well documented README instructions on run the demo.


Current Stage
Prepare for the project
Finish reading and understanding the paper and start with implementing the core modules.
Working on Interactive improvement
Fine tune the iterative implementation and start working on the interactive improvement feature.
Week 1
Week 2
Week 3
Week 4
Final Submission
Finishing up the GrabCut implementation
Implement the UI and 2 of us will work on the Unary energy function implementation and other 2 on the pairwise potential functionality.
Final Submission preparation
Prepare the ablation study of parameters and prepare the repository and deliverables(report and presentation) for final submission.

Mid term Progress

Progress Till Now

1. Finished Implementing a simple UI using OpenCV named windows.

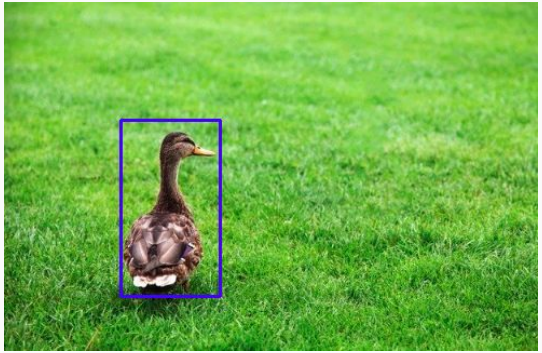
2. Implemented both the energy functions. For the GMM, sklearn's K-Means clustering is used to cluster points and assign the GMM component.

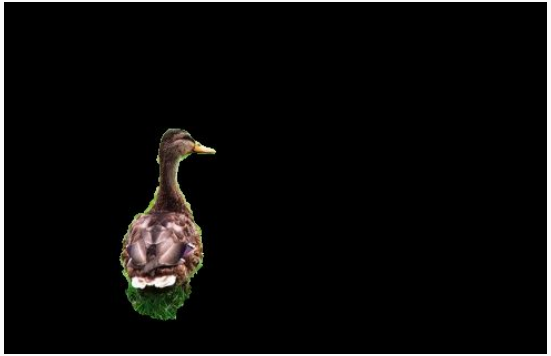
3. For formulating the image as a graph, igraph library is used.

4. igraph's ST Mincut functionality is used to perform mincut over the graph.

5. OpenCV is extensively used for all image related operations, bounding box visualization and generating the final output image.

Results

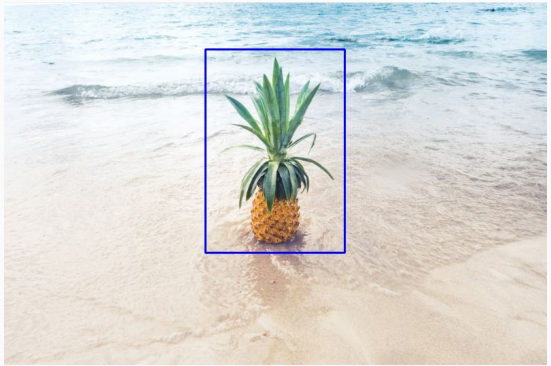




Results



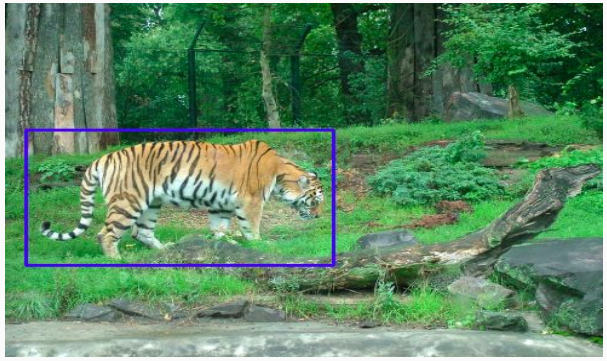





Failed Results








Future Works

1. Implement the Interactive improvement that handles the incorrect cases here with this simple grabcut.
2. Improve the performance of the current grabcut implementation.
3. Improve the UI to make it a full functional app with better functionalities.
4. Perform ablation study over the various parameters.
5. Prepare the code in a more generic and well documented form.
6. Prepare the final deliverables.

01	**Mehul Arora**	• Worked on the GMM module • Designed the UI to accept user input
02	**Neha S.**	• Module to prepare the image as graph • The Smoothness computation module
03	Sabyasachi Mukhopadhyay	• GMM module, computing unary potentials • Designed the UI to accept user input
04	**Sai Amrit Patnaik**	• The Smoothness computation module • Overall Code Organization

Task Distribution

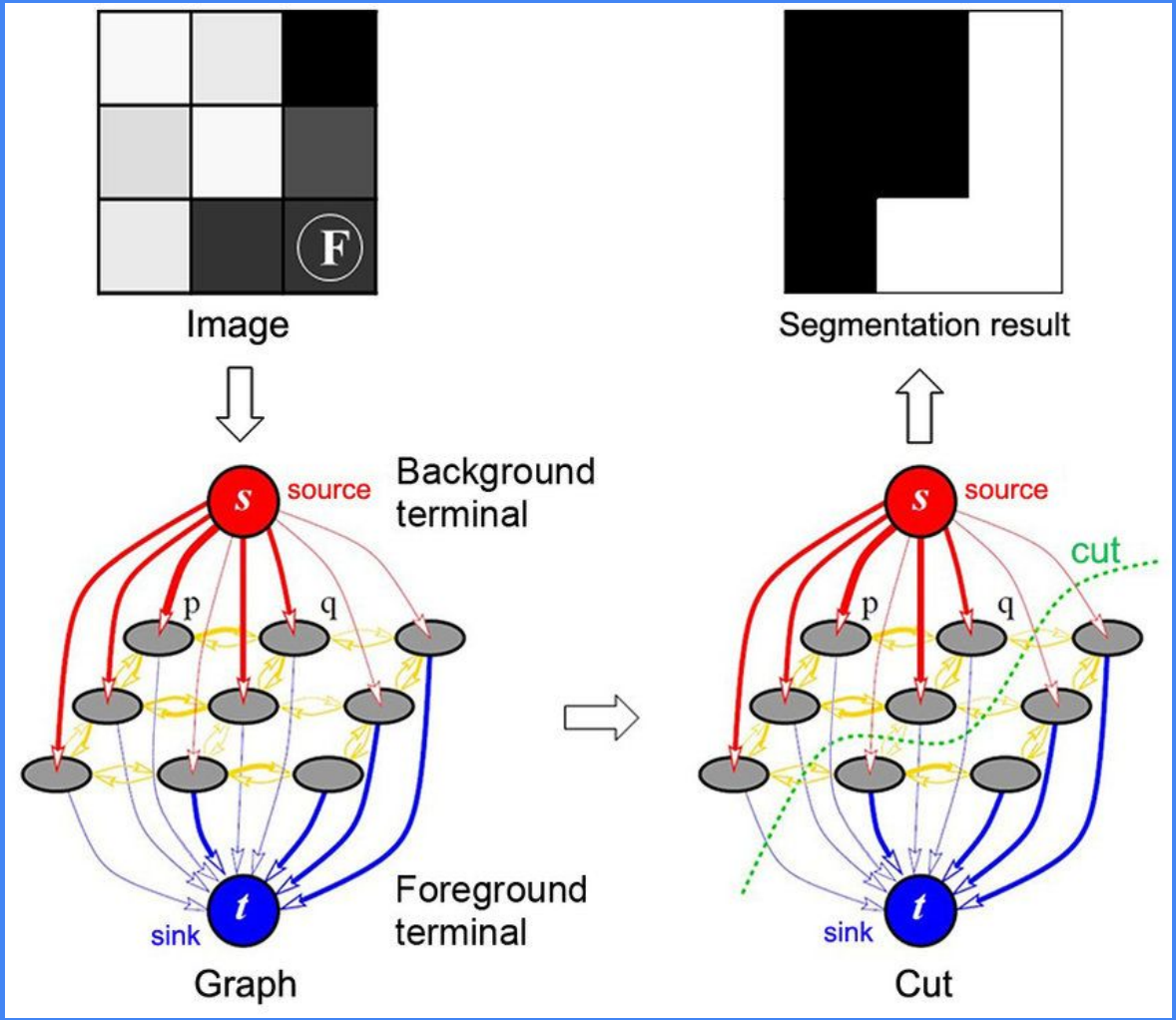
References

1. Graph Cut
   a. http://www.eecs.berkeley.edu/~efros/courses/AP06/Papers/boykoviccv01.pdf

2. GrabCut Interactive Foreground Extraction using Iterated Graph Cuts
   a. http://research.microsoft.com/apps/pubs/default.aspx?id=67890

3. Implementing Grabcut
   a. http://read.pudn.com/downloads94/doc/374106/Grabcut.pdf
   b. http://www.justintalbot.com/research/coursework/

Thanks!


F
Image
Segmentation result
s
source
Background
terminal
p
q
t
sink
Foreground
terminal
Graph
cut
Cut