Image segments visualizer

BS Final Year Project Report

Prepared by: Muhammad Ali

Supervised by: Syed Muhammad Naqi

Department of Computer Science
Quaid-i-Azam University Islamabad

Session: (2014 -2018)

In the name of Allah, the Most Merciful, the Most Kind.

ACKNOWLEDGEMENT

First of all, I would like to extend my sincere and humble gratitude to ALMIGHTY **ALLAH** whose blessing and guidance has been a real source of all my achievements in my life, and who gave me the ability and knowledge to undertake this project and showed me the right path bestows me with his guidance.

I am especially grateful to my supervisor, **Mr. Syed Muhammad Naqi,** whose guidance made me complete this project. He helped me in improving my development skills with her helpful suggestions.

I would also like to thanks all my **friends** for their help, without which the completion of the project would not have been possible.

Muhammad Ali

ABSTRACT

We need to segment images so that it would become easier for the computer to understand image data. Image segmentation is the process of differentiating heterogeneous parts of the image or group similar neighboring pixels together, that could be used for the further applications such as: Image understanding model, Robotics, Image analysis, Medical diagnosis, etc.

The people working on image segmentation face a lot of problems in finding best segmentation algorithm according to their image dataset. Because segmentation tools available in market provide limited image segmentation techniques and also comparison of segmentation algorithms is visual not quantitative, these tools are not free and also mostly tools are online, dependent on internet connection so we are providing a platform where mostly segmentation algorithms are implemented and comparison of these algorithm results with groundtruth are visual as well as quantitative.

Image Segments Visualizer(ISV) is a platform which provide implementation of multiple image segmentation algorithms includes (Otsuthresh, Graythresh, Multithresh, Adaptivethresh, Grayconnected, Canny, Sobel, Prewitt, Log, Active contour, Fast marching, Grabcut, Lazysnapping, Imseggeodestic, Particle Swarm Optimization, Darwinian Particle Swarm Optimization, Fractional Order Darwinian Particle Swarm Optimization, Jaccard distance, Dice coefficient and Boundary F1 Score). Some of these algorithms take image and provide binary mask whereas some represent each segment with different color and some only display foreground. So a person can upload his desired image dataset and analyze the results of different available image segmentation algorithms. ISV also provide comparison of segmented image with groundtruth visually by highlighting un similar regions and also gives percentage of similarity based on Jaccard, dice coefficient, bfscore.

ISV has three main modules first one is for the segmentation of greyscale image. Second module allow user to segment colored(RGB) images and finally the third one is for comparing results of segmented image with ground truth so that user can find the quality of segmentation algorithm.

To implement this project matlab is used which provide us all the api’s to build GUI and backend of ISV. This project is implemented in professional way according to rules of software engineering and it is covering mostly available image segmentation algorithms.

Table of Contents

List of Figures

List of tables

Acronyms and abbreviations

Different acronyms and abbreviations are shown in table below.

Table 1: Acronyms and abbreviations

Acronyms	**Abbreviation**
ISV	Image Segmentation Visualizer.
ISA	Image segmentation algorithm.
QAU	Quaid-I-Azam university.
USER	People working on image segmentation.
UC	Use case.
IPA	Image processing application

Chapter 1 Software Project Management Plan

This chapter describes the introduction to Image Segmentation Visualizer (ISV) project and project management plan. Problem definition of project, its working and reasons to implement this project, proposed solution for problems, scope and objectives of ISV are described in detail. Project organization and project management plan is discussed for every task that in how much time a task could be completed. Project management plan is shown at the end of this section.

1.1 Problem definition

Image segmentation is the process of grouping similar neighboring pixel together. This is typically used to identify objects or other relevant information in digital images that could be used for the further applications such as: Image understanding model, Robotics, Image analysis, Medical diagnosis, etc. Each segmentation technique produces different results on different datasets. An algorithm performs better results in one type of dataset may cannot perform better in other type of dataset.

The people working on image segmentation have to read a lot of text but face a lot of problems in finding best algorithms according their desired datasets. Because segmentation tools available in market provide limited image segmentation algorithms and also comparison of segmentation algorithms is visual not quantitative and these tools are mostly not free.

1.2 Proposed solution

To resolve the problem faced by many people working on image segmentation we are going to develop a platform where user can upload his desired dataset(images) perform segmentation and then analyze the results of multiple image segmentation algorithms by comparing there results with groundtruth. Which help user in finding best algorithm according to the given input dataset. This system doesn’t require any internet connection it is a standalone application. it doesn’t need any authorization and authentication.

1.3 Scope

ISV provide interface to upload image(s) and user can choose image segmentation algorithm upon selecting algorithm user has to give desired parameters and segmented image will be displayed to the user. Also user can upload segmented image and groundtruth then select comparison algorithm in this case visual and quantitative comparison between segmented image and ground truth is displayed to the user.

1.4 Objectives

The main objective of the systems is to facilitate the people working on image segmentation in finding best algorithm according to their image dataset. So with the basic knowledge of image segmentation algorithm user can test it with its desired dataset and analyze the results by comparing it with groundtruth. Which help him to processed its work on image segmentation within very less amount of time.

1.5 Project Organization

Project organization describes the language in which project is implemented and during implementation of project which process model will be used and tools that will be used to implement the project.

1.5.1 Software process model

Rapid Application Development (RAD) software process model [1] is used in our project because Rapid application development is an incremental "software development process model" that emphasizes an extremely short development cycle. The RAD model is a "high-speed" adaptation of the linear sequential model in which rapid development is achieved by using component-based construction. Different components are independent of each other and are implemented independently. It is used for medium projects those could be implemented in duration of 2-3 months and at the end, modules could be integrated with each other.

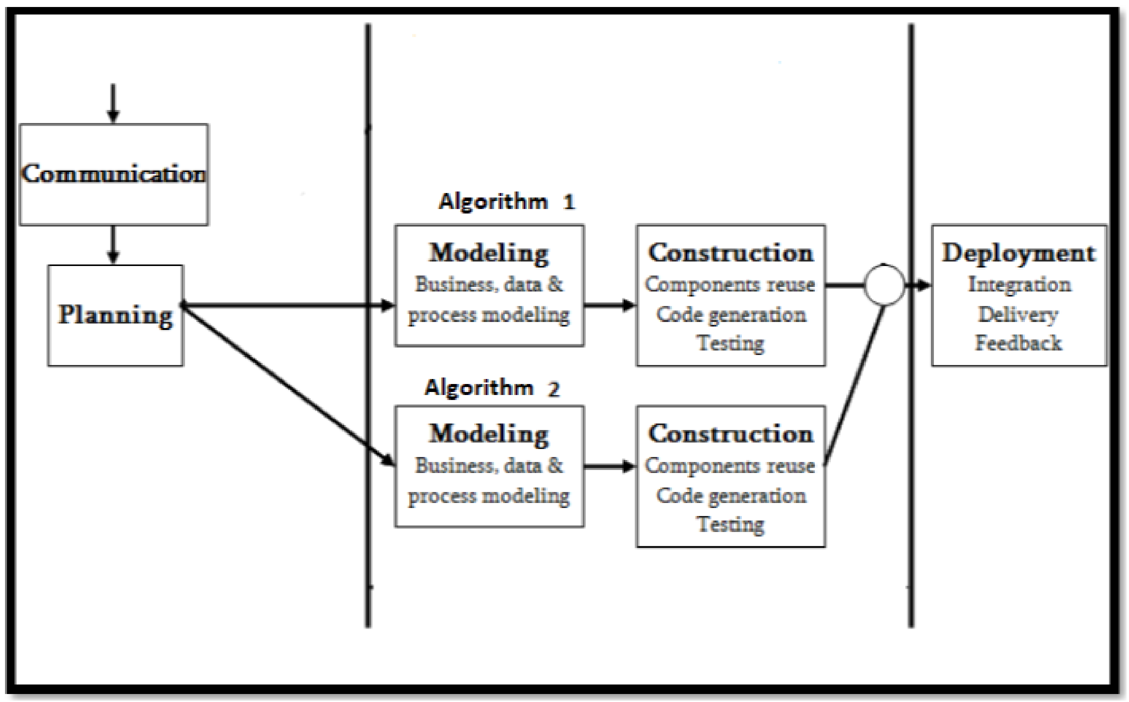


Figure 1 : Process model

1.5.2 Roles and responsibilities

ISV has only one type of users there is no authorization for performing any activity even user has no need to register for using this software it is open to everyone.

1.5.3 Tools and techniques

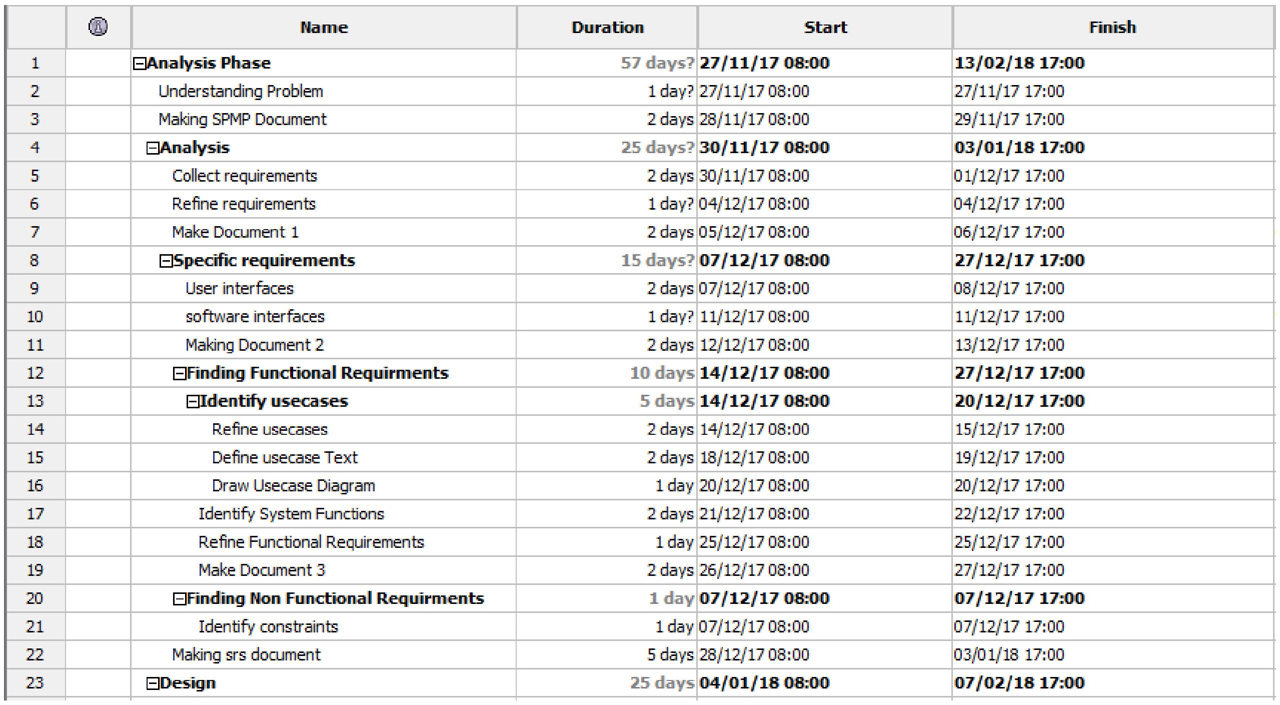
Different Tools to implement this project are used as follows.

Table 2: Tools and techniques

Tools	**Use**
Matlab R2018a	To implement algorithms and GUI.
Microsoft Visio 2016	To draw different diagrams e.g. Usecase.
Microsoft Word 2013	To write documentation.
Argo UML	To draw diagrams.
Project Libre	For project management plan.

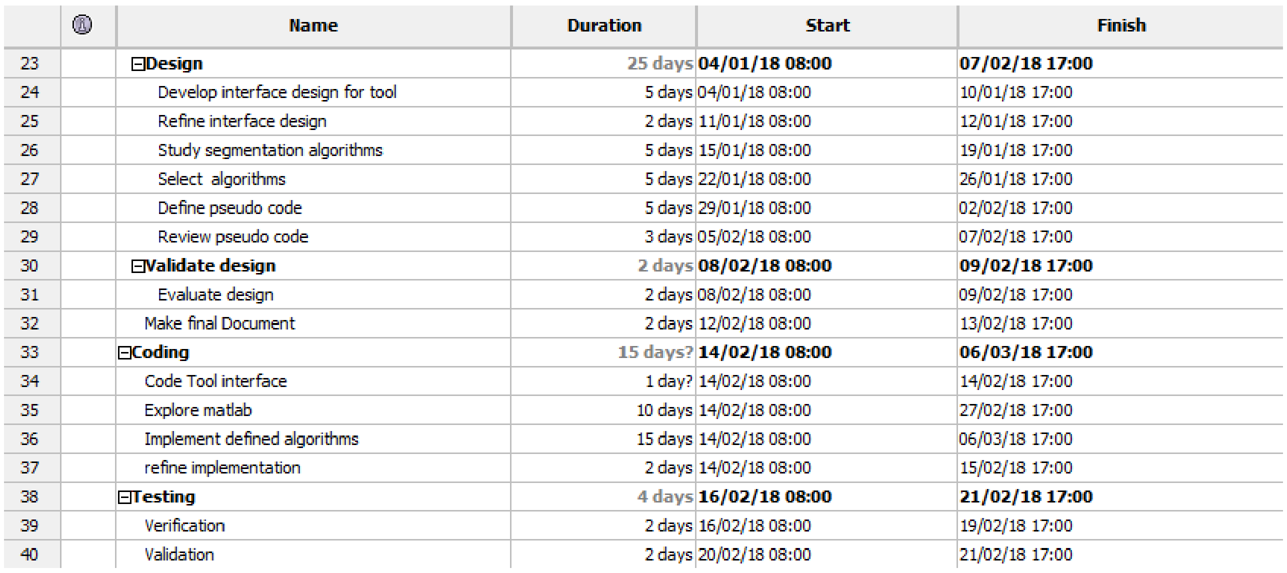
1.6 Project Management Plan

Project management plan is created in Project Libre. In project management plan resources name, start date, finish dates of every task are listed and Time line chart for every task with duration is shown.



	ⓘ	Name	Duration	Start	Finish
1		**⊟Analysis Phase**	57 days?	**27/11/17 08:00**	**13/02/18 17:00**
2		Understanding Problem	1 day?	27/11/17 08:00	27/11/17 17:00
3		Making SPMP Document	2 days	28/11/17 08:00	29/11/17 17:00
4		**⊟Analysis**	25 days?	**30/11/17 08:00**	**03/01/18 17:00**
5		Collect requirements	2 days	30/11/17 08:00	01/12/17 17:00
6		Refine requirements	1 day?	04/12/17 08:00	04/12/17 17:00
7		Make Document 1	2 days	05/12/17 08:00	06/12/17 17:00
8		**⊟Specific requirements**	15 days?	**07/12/17 08:00**	**27/12/17 17:00**
9		User interfaces	2 days	07/12/17 08:00	08/12/17 17:00
10		software interfaces	1 day?	11/12/17 08:00	11/12/17 17:00
11		Making Document 2	2 days	12/12/17 08:00	13/12/17 17:00
12		**⊟Finding Functional Requirments**	10 days	**14/12/17 08:00**	**27/12/17 17:00**
13		**⊟Identify usecases**	5 days	**14/12/17 08:00**	**20/12/17 17:00**
14		Refine usecases	2 days	14/12/17 08:00	15/12/17 17:00
15		Define usecase Text	2 days	18/12/17 08:00	19/12/17 17:00
16		Draw Usecase Diagram	1 day	20/12/17 08:00	20/12/17 17:00
17		Identify System Functions	2 days	21/12/17 08:00	22/12/17 17:00
18		Refine Functional Requirements	1 day	25/12/17 08:00	25/12/17 17:00
19		Make Document 3	2 days	26/12/17 08:00	27/12/17 17:00
20		**⊟Finding Non Functional Requirments**	1 day	**07/12/17 08:00**	**07/12/17 17:00**
21		Identify constraints	1 day	07/12/17 08:00	07/12/17 17:00
22		Making srs document	5 days	28/12/17 08:00	03/01/18 17:00
23		**⊟Design**	25 days	**04/01/18 08:00**	**07/02/18 17:00**

Figure 2 : Project Plan Part 1



	ⓘ	Name	Duration	Start	Finish
23		**⊟Design**	25 days	**04/01/18 08:00**	**07/02/18 17:00**
24		Develop interface design for tool	5 days	04/01/18 08:00	10/01/18 17:00
25		Refine interface design	2 days	11/01/18 08:00	12/01/18 17:00
26		Study segmentation algorithms	5 days	15/01/18 08:00	19/01/18 17:00
27		Select algorithms	5 days	22/01/18 08:00	26/01/18 17:00
28		Define pseudo code	5 days	29/01/18 08:00	02/02/18 17:00
29		Review pseudo code	3 days	05/02/18 08:00	07/02/18 17:00
30		**⊟Validate design**	2 days	**08/02/18 08:00**	**09/02/18 17:00**
31		Evaluate design	2 days	08/02/18 08:00	09/02/18 17:00
32		Make final Document	2 days	12/02/18 08:00	13/02/18 17:00
33		**⊟Coding**	15 days?	**14/02/18 08:00**	**06/03/18 17:00**
34		Code Tool interface	1 day?	14/02/18 08:00	14/02/18 17:00
35		Explore matlab	10 days	14/02/18 08:00	27/02/18 17:00
36		Implement defined algorithms	15 days	14/02/18 08:00	06/03/18 17:00
37		refine implementation	2 days	14/02/18 08:00	15/02/18 17:00
38		**⊟Testing**	4 days	**16/02/18 08:00**	**21/02/18 17:00**
39		Verification	2 days	16/02/18 08:00	19/02/18 17:00
40		Validation	2 days	20/02/18 08:00	21/02/18 17:00

Figure 3 : Project Plan Part 2

Chapter 2 Software Requirement Specification

Software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform. SRS describes the complete information about system that what end user wants. What would be inputs of users to system and what would be response of system corresponding to user’s input. Functional and non-functional requirements of system are detailed completely in SRS document.

2.1 Overview

Requirements gathering and analysis clears the requirements before implementation of project. This chapter clears that what end users of project wants and what would be the output of system. So with different aspects requirements gathering and analysis is described below.

2.1.1 Purpose

The Purpose of this Requirement gathering and analysis is to clear the requirements of the system and to decide what the system should do and what the system should not do. Other purpose is to clear the requirements and analyze them that how those requirements would help in implementing the project with actual features that end users want.

2.1.2 Stakeholders

People those are working on image segmentation are main stakeholders of ISV.

2.1.3 Major functions

Major functions of ISV is to provide user interface with multiple image segmentation algorithms embedded in it so that people those are not aware of programming languages can upload images and apply image segmentation algorithms without writing any single line of code.

2.1.4 Major Inputs and Outputs

Main inputs to ISV are images whether these are grayscale or color images and segmentation algorithm with required parameters. While outs are segmented images maybe binary or labeled or colored images.

2.2 Overall description

Overall description of system describes how the system would perform with different operations or provides features. Overall description of system is described as below.

2.2.1 Product Perspective

ISV is standalone application. This is a desktop application in which user upload images select his desired image segmentation algorithm and give required parameters and perform image segmentation the resultant segmented image will be displayed to the user now if user want he can save it for future use.

2.2.2 User Interfaces

ISV is open software there is no authentication and authorization so each user can perform same tasks so it has no many user interfaces one main user interface is to upload images and selecting algorithms which will dynamically change based on user inputs.

2.2.3 Software interfaces

Software interfaces are different software to implement and run an application. In this application, software interfaces for client and admin are matlab and any operating system. Software interfaces on developing side are matlab R2018a for development environment. Different toolboxes are also required to handle algorithms and create GUIs like image processing toolbox introduced by matlab.

2.3 Product Functions

Product functions describe the functions or operations that a system will perform. Different functions with details are described below.

2.3.1 Upload Image

System will provide a function to upload image from windows file management system for further image processing.

2.3.2 Upload Multiple Images

System allows user to apply image segmentation algorithm on multiple images of different dataset at once. But this feature has a limitation, user will be able to apply only automatic segmentation algorithm if user select multiple images at once.

2.3.3 Select Folder

System will provide a function to select folder from windows file management system and upon selecting a folder all images in it will be loaded into memory for segmentation. Only automatic segmentation algorithm will be provided in this case.

2.3.4 Chose Algorithm

Once image is selected system allows user to select his desired algorithm for segmentation.

2.3.5 Set Algorithm parameters

ISV provide a functionality for setting parameters of image segmentation algorithms according to user's needs this will give user more control for better understanding of algorithm working.

2.3.6 Measure similarity

System also provide 3 functions for comparing results with ground truth to rank image segmentation algorithms or to measure quality of segmentation but this feature require ground truth.

Example

1. Jaccard similarity coefficient.
2. bfscore.
3. dice similarity coefficient.

2.3.7 Save Results

System will also provide interface for saving resultant segmented images into window file's management system for future use.

2.4 User Characteristics

This is assumed that the users know English language and can read and write it. User has laptop, or computer to perform image segmentation.

2.4.1 Constraints

User should have matlab runtime 9 version installed on it to use this system. User should have its images dataset for segmentation. System is implemented in english language so the user should have proper knowledge of english to use the system.

2.4.2 Assumptions and Dependencies

It is assumed that the user has matlab runtime 9 installed on his system and user has dataset of images to perform segmentation task. User should have basic how know of image segmentation.

2.6 Use cases

Table 3: Use cases

Id	**Name**	**Goal**
UC1	Upload image	Get image from file system.
UC2	Upload multiple images	Get multiple images from file system.
UC3	Select desired directory	Get folder containing images.
UC4	Choose desired algorithm’s	Perform segmentation through selected algorithm’s
UC5	Set algorithm’s parameters	Initialize algorithm for segmentation.

2.7 Use case Diagram

Use case diagram is graphical representation of user interaction with system. Use case diagram represents that how users interact with the components of system. So all use cases are mentioned in diagram and how the users interact with use cases is shown by diagram.

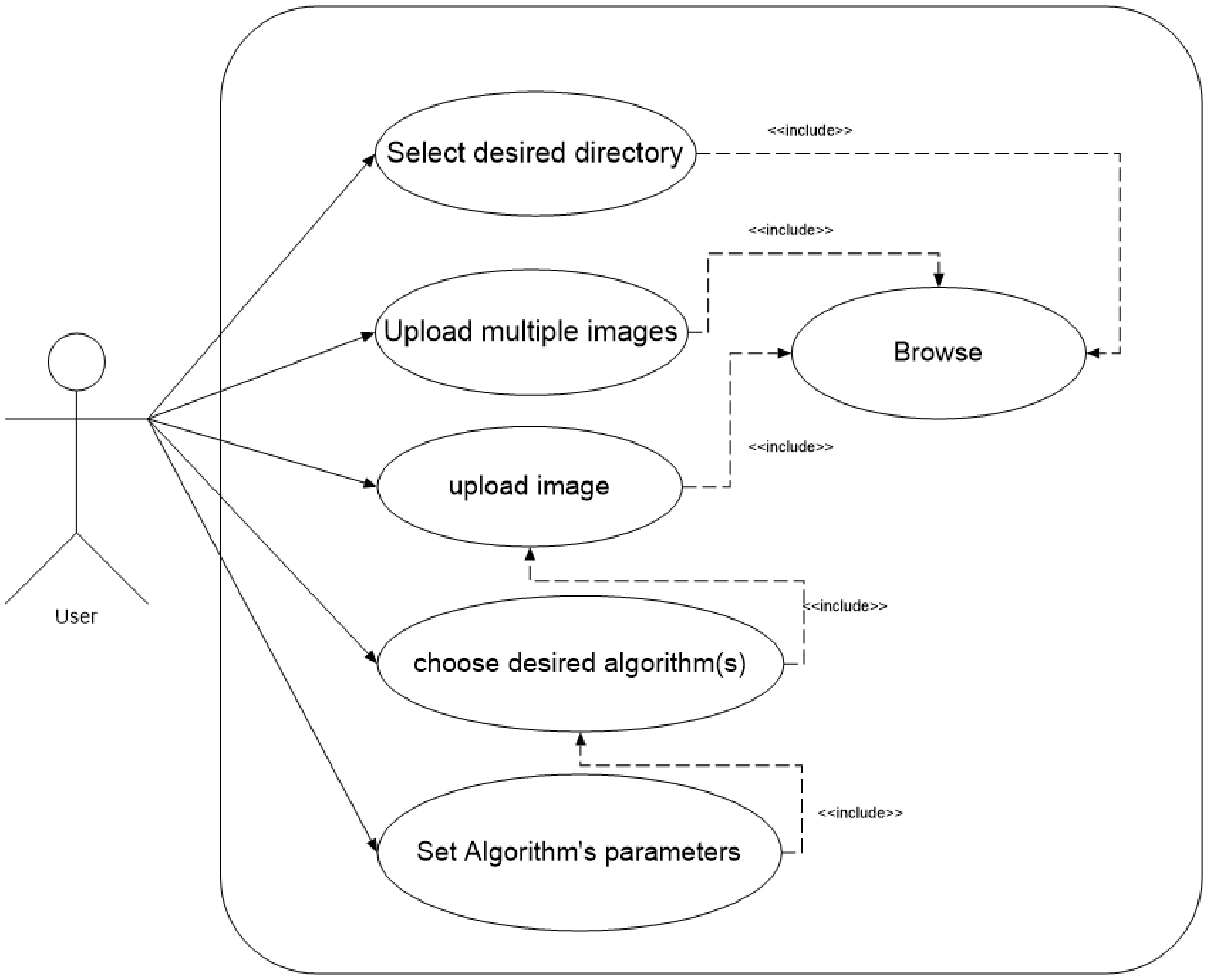


Figure 4: use case diagram

2.8 Usecase Descriptions

2.8.1 Use case 1: Upload image

Table 4: Use case 1

ID	**UC1**
Name	Upload image
Primary Actor	Developer, Researcher.
Stockholder's and interests:	**Developer:** who wants to choose an algorithm to use in development of an IPA. **Researcher:** To find an ISA in which he wants to proceeds his research.
Input	Image.
Pre-conditions	None.
Post-conditions	Image is selected from file system and uploaded into memory.
Main Success Scenario:	1. User start browsing file system by pressing upload button. 2. User select desired directory. 3. User choose desired image. 4. User press open button.
Alternatives:	1. Upload button doesn't response to user. 2. Desired directory not found. 3. Desired image not found. 3.a Desired image is corrupt.
Frequency of occurrence:	Every time user wants to perform segmentation on a new image.

2.8.2 Use case 2: Upload multiple images

Table 5: Use case 2

ID	**UC2**
Name	Upload multiple images.
Primary Actor	Developer, Researcher.
Stockholder's and interests:	**Developer:** who wants to choose an algorithm to use in development of an IPA. **Researcher:** To find an ISA in which he wants to proceeds his research.
Input	Images.
Pre-conditions	None.
Post-conditions	Images are selected from file system and uploaded into memory.
Main Success Scenario:	1. User start browsing file system by pressing upload button. 2. User select desired directory. 3. User check all desired images. 4. User press open button.
Alternatives:	1. Upload button doesn't response to user. 2. Desired directory not found. 3. Desired images not found. 3.a Any selected image is corrupt.
Frequency of occurrence:	Every time user wants to perform automatic segmentation on multiple images at once.

2.8.3 Use case 3: Select desired directory

Table 6: Use case 3

ID	**UC3**
Name	Select desired directory.
Primary Actor	Developer, Researcher.
Stockholder's and interests:	**Developer:** who wants to choose an algorithm to use in development of an IPA. **Researcher:** To find an ISA in which he wants to proceeds his research.
Input	Directory containing images.
Pre-conditions	None.
Post-conditions	Path of selected directory and all the images in that directory are uploaded in memory.
Main Success Scenario:	1. User start browsing file system by pressing upload button. 2. User select desired directory. 3. User press open button.
Alternatives:	1. Upload button doesn't response to user. 2. Desired directory not found. 3. Desired directory contains corrupt images.
Frequency of occurrence:	Every time user wants to perform automatic segmentation on multiple image at once.

2.8.4 Use case 4: Choose desired algorithm

Table 7: Use case 4

ID	**UC4**
Name	Choose desired algorithm.
Primary Actor	Developer, Researcher.
Stockholder's and interests:	**Developer:** who wants to choose an algorithm to use in development of an IPA. **Researcher:** To find an ISA in which he wants to proceeds his research.
Input	Image(s) and parameter values.
Pre-conditions	Image(s) are selected.
Post-conditions	Segmented image displayed to user and saved to root directory.
Main Success Scenario:	1. User select algorithm category from colored or greyscale categories. 2. User check desired algorithm.
Alternatives:	1. Desired algorithm doesn't support uploaded image type. 2. Result image is not displayed to user.
Frequency of occurrence:	Every time user wants to perform segmentation on uploaded image(s).

2.8.5 Use case 5: Set algorithm's parameters

Table 8: Use case 5

ID	**UC5**
Name	Set algorithm's parameters.
Primary Actor	Developer, Researcher.
Stockholder's and interests:	**Developer:** who wants to choose an algorithm to use in development of an IPA. **Researcher:** To find an ISA in which he wants to proceeds his research.
Input	Image and algorithm.
Pre-conditions	Algorithm is selected and input image is displayed.
Post-conditions	Foreground and background are differentiated, segmentation is performed and results are displayed.
Main Success Scenario:	1. User select the foreground areas of image. 2. User select the background areas of uploaded image.
Alternatives:	1. Foreground area is not properly selected. 1.a rest and again select foreground area. 2. Background and foreground overlap. 2.a rest and again select background otherwise results of algorithm will be not expected.
Frequency of occurrence:	Every time user wants to perform segmentation on uploaded image(s).

2.9 Database requirement

This application doesn't require any database because it will load images from operating systems file system.

Chapter 3 Software Design Description

This chapter reviews the design description for the system of ISV. In this chapter introduction of system design description, system architecture design, detailed description of components and user interfaces are described in detail. Sequence diagrams are also designed in this chapter.

3.1 Design overview

System design describes the system at architecture level, services of system, data management of system and complete structure of system. In design phase of system, user interfaces or screen images of system are designed. In addition, system sequence diagram that is an interaction diagram, which shows sequence of interaction with system and user, State chart diagram of system is designed that describes multiple states of system.

3.2 System Architecture Design

Architectural design is a creative process where we design a system organization that will satisfy the functional and non-functional requirements of a system. System architecture design is the set of significant decisions about how multiple components of a software system interact with each other, the selection of the structural elements and their interfaces by which the system is composed. System architecture describes organization, styles, patterns, responsibilities, collaborations, connections, and motivations of a system and major subsystems.

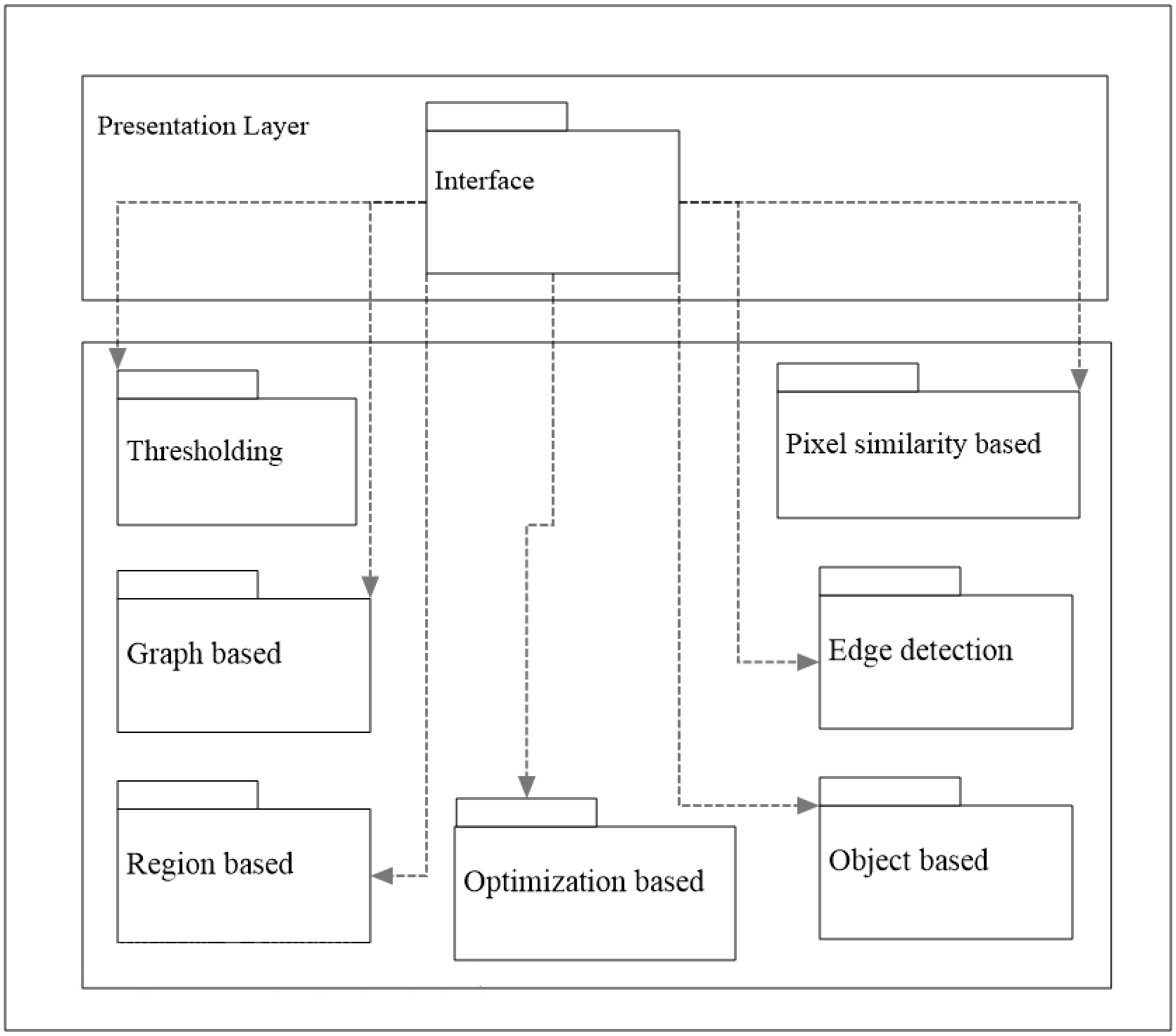


Figure 5 : Architecture Design

3.3 Architecture diagram description

In the description of architecture diagram all the modules those are used in architecture diagram are further explained in details for better understanding that how we have reduced coupling and increased cohesion.

3.3.1 Interface module

Interface module contains all coding related to user interfaces of the tool. Interface module will send all the user selected information to the corresponding selected functions.

3.3.2 Thresholding module

Thresholding module contains thresholding algorithms as a functions. In which we first find and optimal threshold and then by using that threshold value we binarize and image.

Some of thresholding algorithms are following

Graythresh

Gray thresh computes a global threshold level, that can be used to convert an intensity image to a binary image with imbinarize. The graythresh function uses Otsu's method, which chooses the threshold to minimize the intra class variance of the black and white pixels [2].

Otsuthresh

Otsuthresh counts computes a global threshold T from histogram counts, using Otsu's method [2]. T is a normalized intensity value that lies in the range [0, 1] that can be used with imbinarize to convert an intensity image to a binary image. Otsu's method chooses a threshold that minimizes the intra class variance of the thresholded black and white pixels.

Adaptthresh

Adapt thresh computes a locally adaptive threshold that can be used with the imbinarize function to convert an intensity image to a binary image. The result is a matrix the same size as I containing normalized intensity values in the range [0,1]. Adaptthresh chooses the threshold based on the local mean intensity (first-order statistics) in the neighborhood of each pixel.

3.3.3 Graph Based algorithms

In this module we have implemented two algorithms used grabcut and lazysnapping.

The basic working of graph based image segmentation is that firstly we will select terminal nodes s and t and then we create a graph by inserting an edge between each pixel and terminal nodes. We also assign weights to each edge based on pixel similarity with terminal node [3]. After graph is constructed properly we will remove edges by using min-cut/max-flow algorithm and in this way our background and foreground will be differentiated to each other.

Grabcut and Lazysnapping

Lazysnapping separate foreground from background. Grabcut algorithm was designed by Carsten Rother, Vladimir Kolmogorov & Andrew Blake from Microsoft Research Cambridge, UK [4].

Background process in grabcut image segmentation.

1. User draw a freehand shape. Everything outside this shape will be taken as sure background. Everything inside shape is unknown. Similarly, any user input specifying foreground and background are considered as hard-labelling which means they won't change in the process.
2. Computer does an initial labelling depending on the data we gave. It labels the foreground and background pixels (or it hard-labels)
3. Now a Gaussian Mixture Model(GMM) is used to model the foreground and background.
4. Depending on the data we gave, GMM learns and create new pixel distribution. That is, the unknown pixels are labelled either probable foreground or probable background depending on its relation with the other hard-labelled pixels in terms of color statistics (It is just like clustering).
5. A graph is built from this pixel distribution. Nodes in the graphs are pixels. Additional two nodes are added, **Source node** and **Sink node**. Every foreground pixel is connected to Source node and every background pixel is connected to Sink node.
6. The weights of edges connecting pixels to source node/end node are defined by the probability of a pixel being foreground/background. The weights between the pixels are defined by the edge information or pixel similarity. If there is a large difference in pixel color, the edge between them will get a low weight.
7. Then a mincut algorithm is used to segment the graph. It cuts the graph into two separating source node and sink node with minimum cost function. The cost function is the sum of all weights of the edges that are cut. After the cut, all the pixels connected to Source node become foreground and those connected to Sink node become background.
8. The process is continued until the classification converges.

3.3.4 Optimization based segmentation

In this module we used particular swarm optimization algorithm to find optimal threshold based on which we separate multiple regions in an image but particular swarm optimization has a drawback it sometimes stuck in local solutions so we have also used Darwinian particular swarm optimization and also we have Fractional-Order Darwinian particular swarm optimization for image segmentation [5].

3.3.5 Edge detection module

Segmentation methods based on discontinuity find for abrupt changes in the intensity value. These methods are called as Edge or Boundary based methods. Edge detection is the problem of fundamental importance in image analysis, Edge detection is the most common approach for detecting meaningful discontinuities in the gray levels in an image. Image segmentation methods for detecting discontinuities are boundary based methods. Edges are local changes in the image intensity. Edges typically occur on the boundary between two regions. Important features can be extracted from the edges of an image (e. g., corners, lines, curves). Edge detection is an important feature for image analysis. These features are used by higher-level computer vision algorithms (e. g. recognition). Edge detection is used for object detection which serves various applications like medical image processing, biometrics etc. Edge detection is an active area of research as it facilitates higher level image analysis. There are three different types of discontinuities in the grey level like point, line and edges. Spatial masks can be used to detect all the three types of discontinuities in an image.

All the edge detection operators implemented in this module are grouped under two groups as

- 1st order derivative
    - Prewitt operator
    - Sobel operator
    - Canny operator
- 2nd order derivative
    - Laplacian operator

The Process of Canny edge detection algorithm can be broken down to 5 different steps [6]:

1. Apply Gaussian filter to smooth the image in order to remove the noise

2. Find the intensity gradients of the image
3. Apply non-maximum suppression to get rid of spurious response to edge detection
4. Apply double threshold to determine potential edges
5. Track edge by hysteresis : Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges.

3.3.6 Region based segmentation module

This module contains two algorithms imseggeodestic [7]. which segment image into two or three regions using geodesic distance-based color segmentation.

3.3.7 Object based segmentation module

This module contains two algorithms provided by matlab.

1. Active contour.
2. Fast marching.

Active contour
Segments the image A into foreground (object) and background regions using the active contour algorithm, also called *snakes*, you specify curves on the image that move to find object boundaries. The active contour function evolves the segmentation using an iterative process and, by default, active contour performs 100 iterations [8].

To obtain faster and more accurate segmentation results, specify an initial contour position that is close to the desired object boundaries.

Fast marching
This method gives a segmented binary image, which is computed using the Fast Marching Method [9].

3.4 Description of User Interfaces

Screen images are interfaces through which user interacts with system. In this section simple prototypes according to analysis of system are drawn. In implementation of the system, following prototypes of system will be used.

3.4.1 UI:1 Home screen

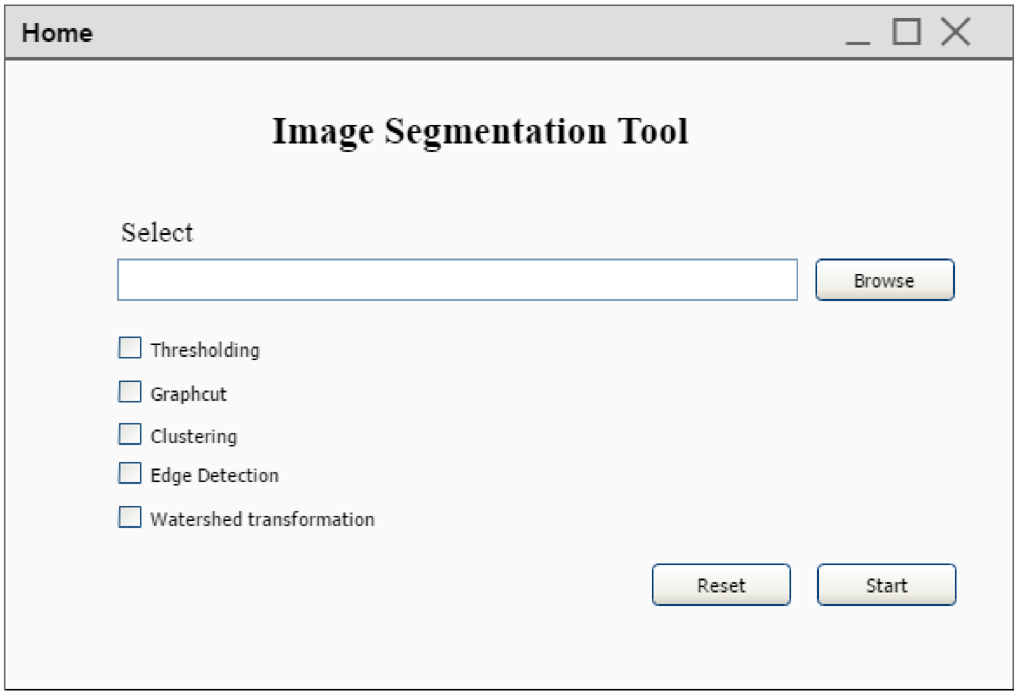


Figure 6 :home screen

When user want to use this system user has to browse the window file management system upload an image from file system.

3.4.2 UI:2 Select Image

To select an image or directory of images.

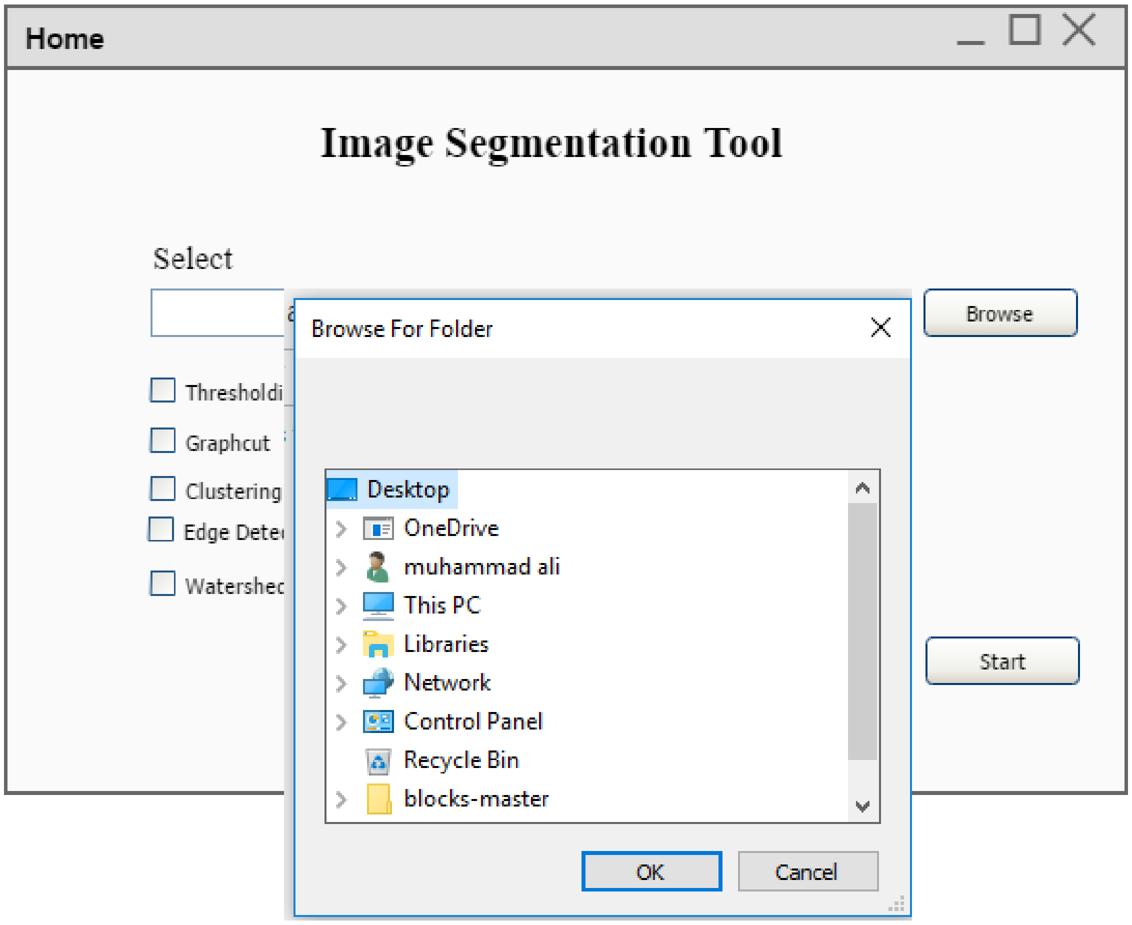


Figure 7 : Select Image

3.4.3 UI : 3 Set Background and Forground

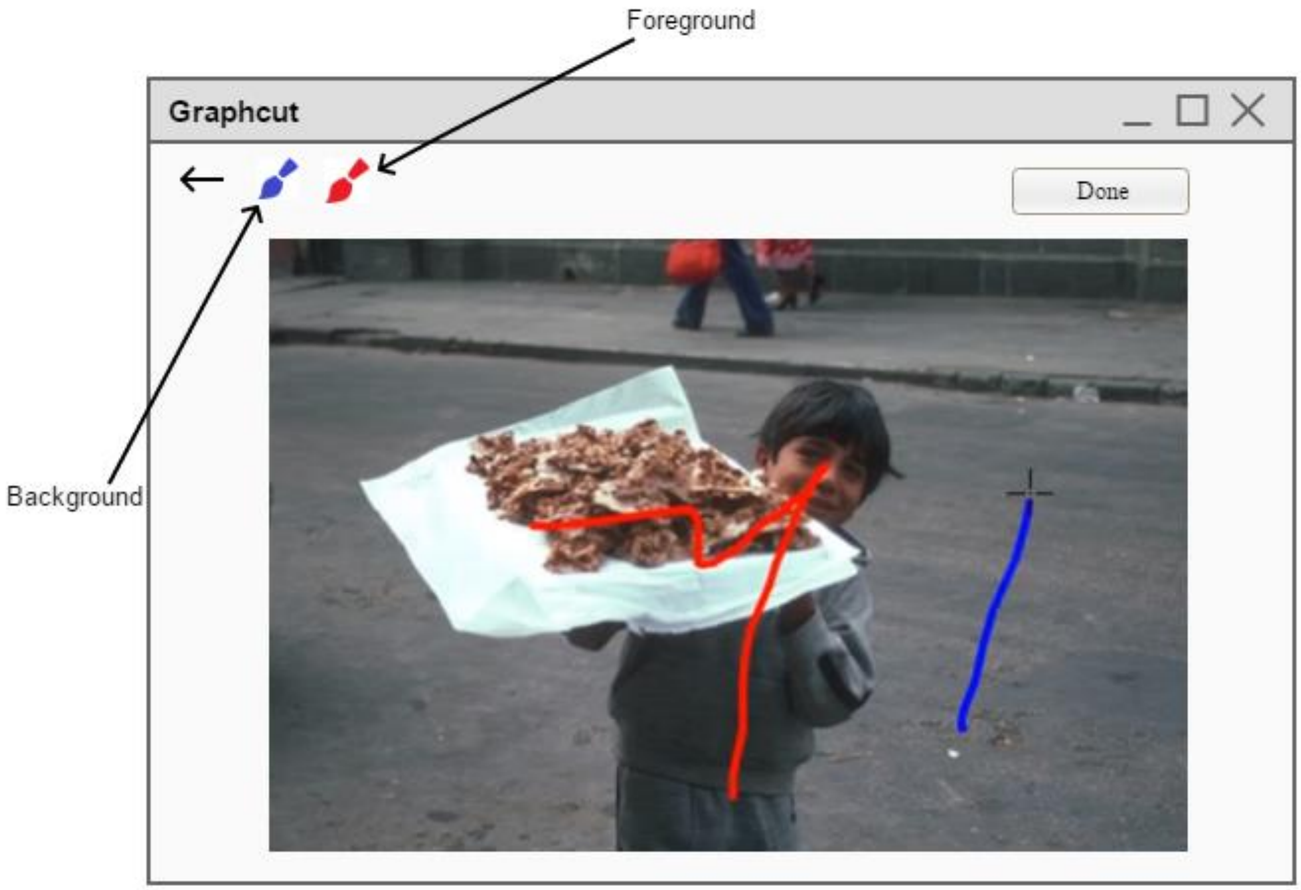


Figure 8 : Set Foreground and Background

3.4.4 UI: 4 Results Interface

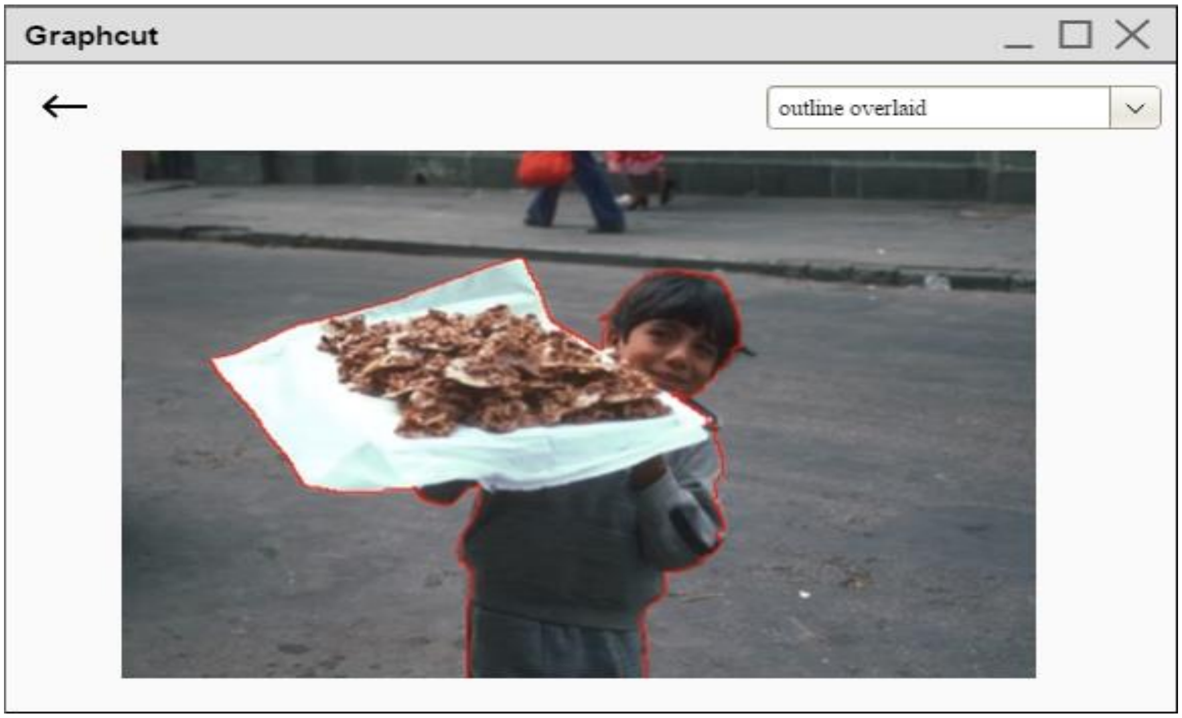


Figure 9 : Result Interface

3.4.5 UI: 5 Foreground Only

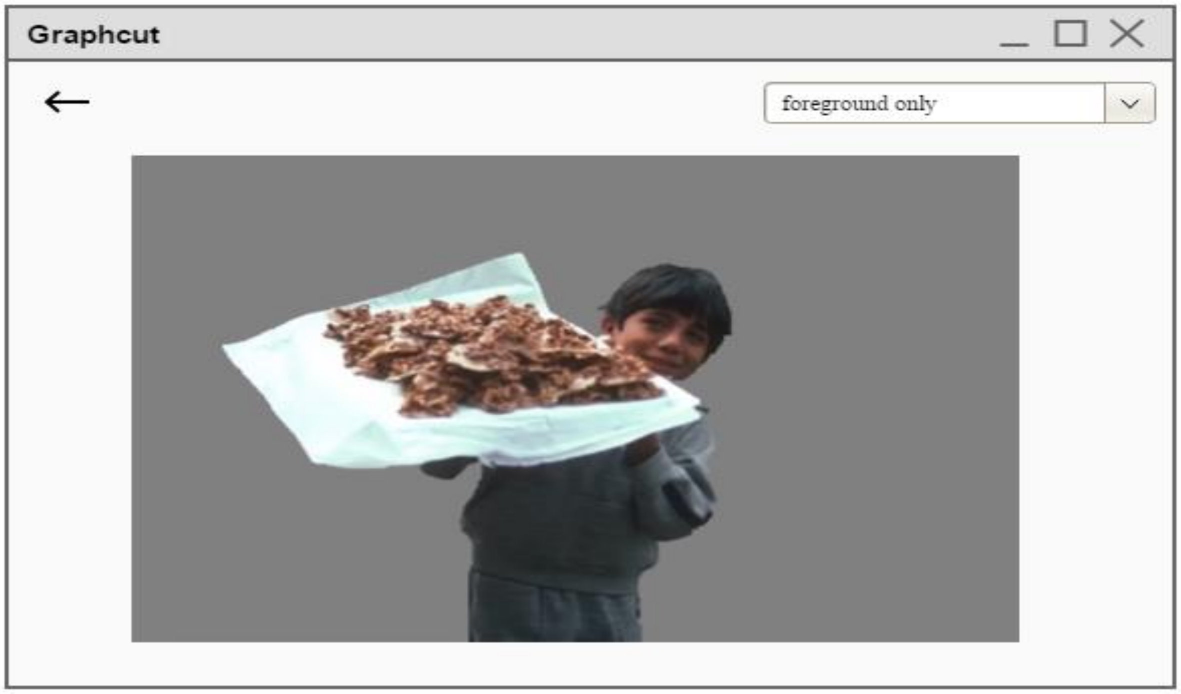


Figure 10 : Foreground only

3.5 System sequence diagram

System sequence diagrams are ideal for demonstrating when and how tasks are completed in a system, how user will interact with the system especially as those tasks related to use cases. Followings are some necessary system sequence diagrams for Image Segmentation Visualizer.

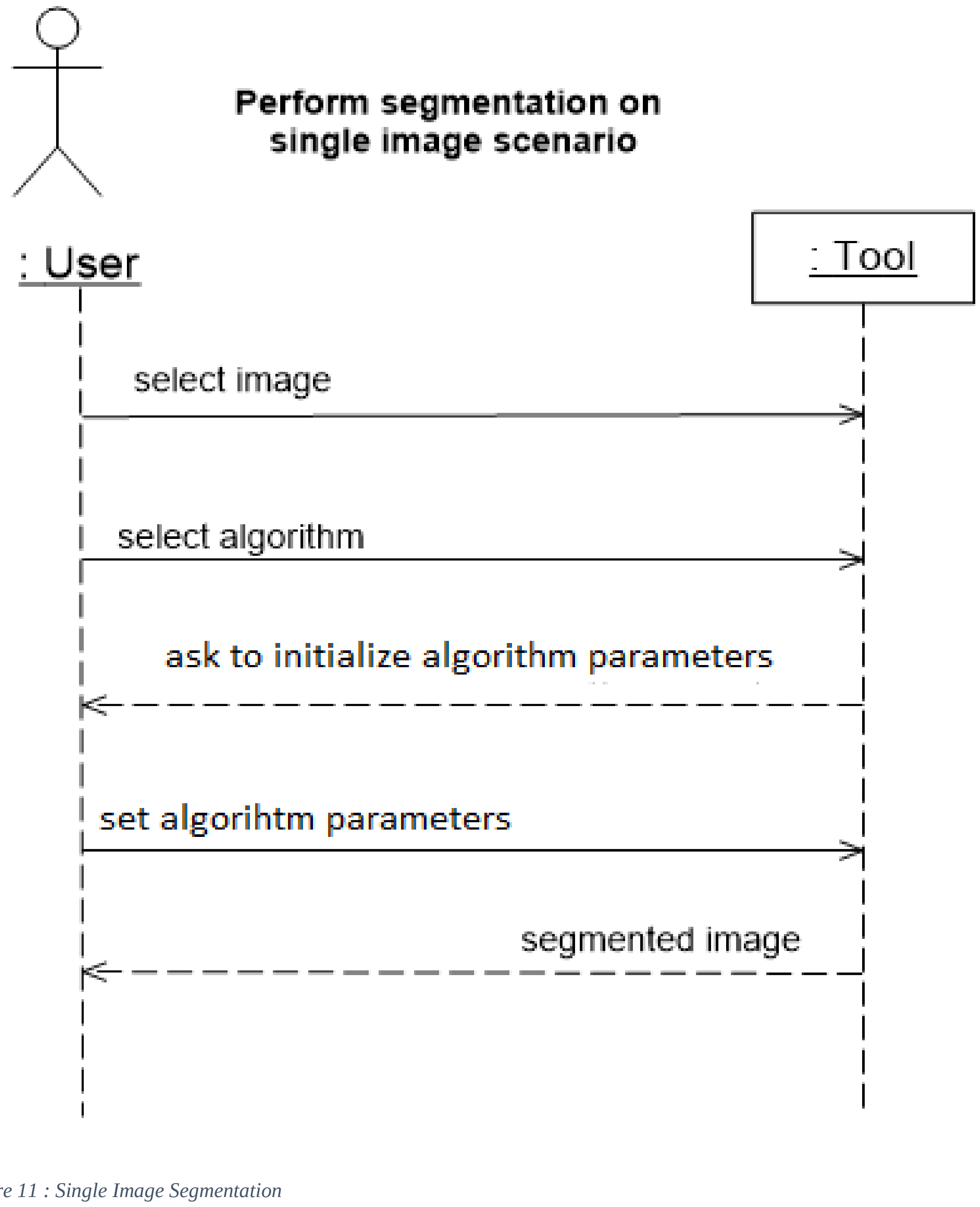

Perform segmentation on single image scenario
: User
: Tool
select image
select algorithm
ask to initialize algorithm parameters
set algorihtm parameters
segmented image

Figure 11 : Single Image Segmentation

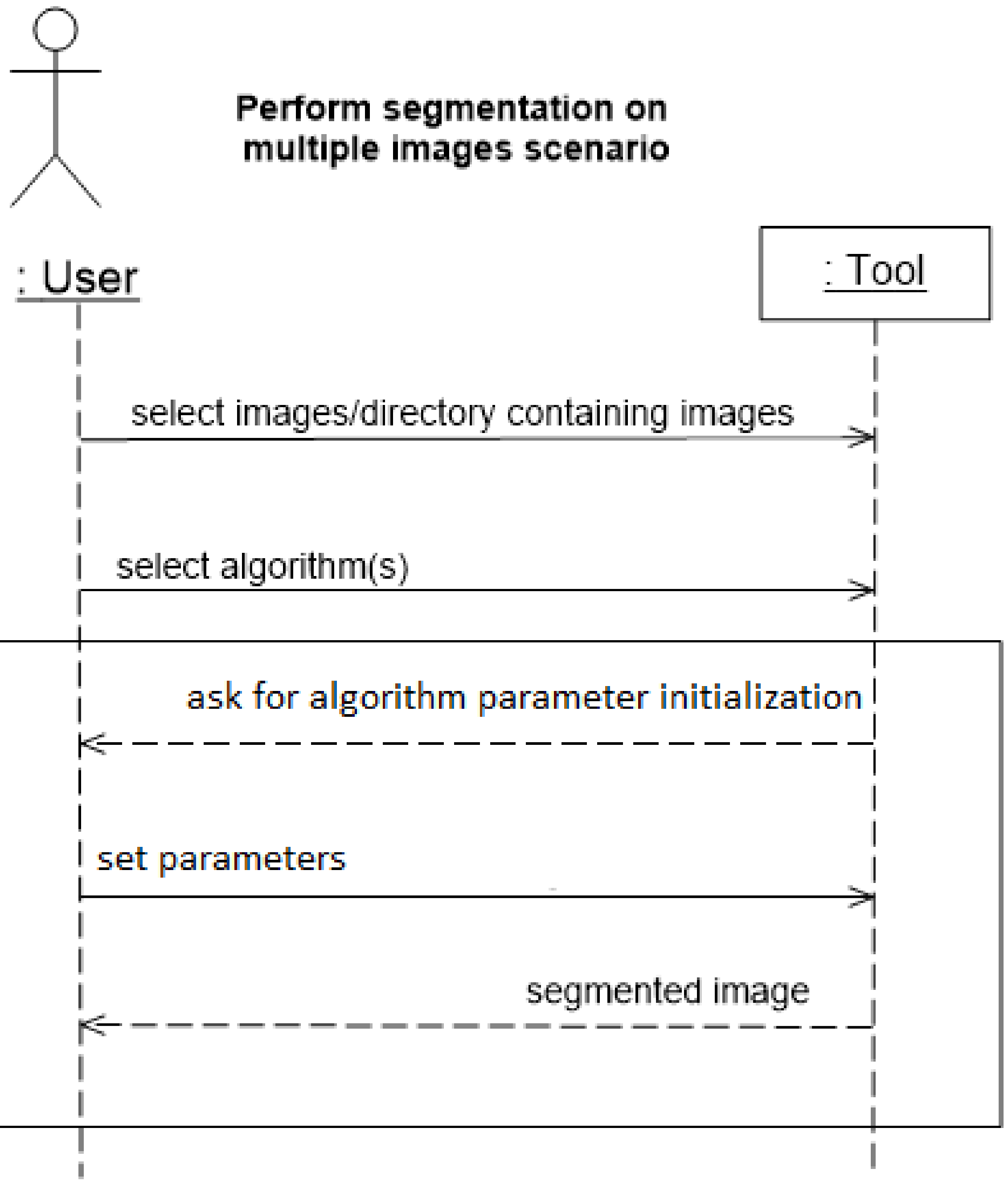

Perform segmentation on multiple images scenario
: User
: Tool
select images/directory containing images
select algorithm(s)
ask for algorithm parameter initialization
set parameters
segmented image

Figure 12 : Multiple Image Segmentation

3.6 State chart diagram

Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

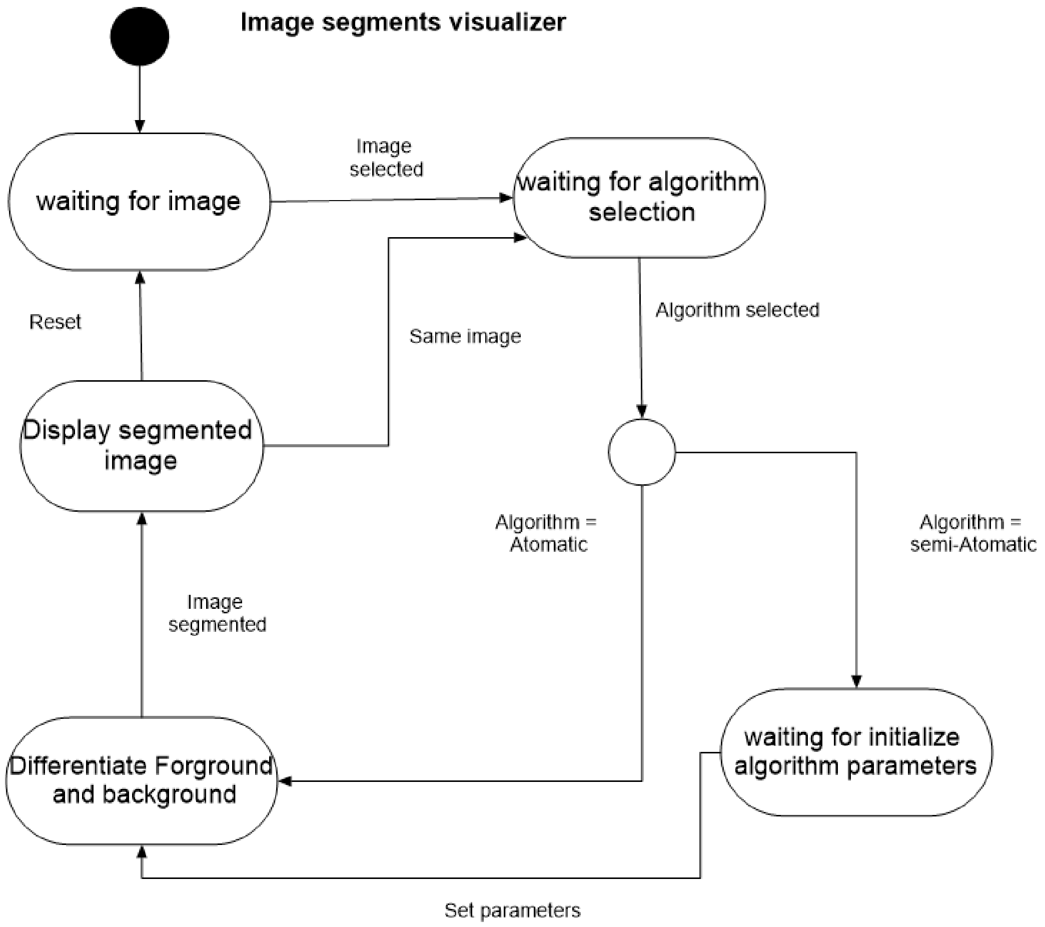


Image segments visualizer
waiting for image
Image selected
waiting for algorithm selection
Reset
Same image
Algorithm selected
Display segmented image
Algorithm = Atomatic
Algorithm = semi-Atomatic
Image segmented
Differentiate Forground and background
waiting for initialize algorithm parameters
Set parameters

Figure 13 : State Chart Diagram

3.7 Activity diagrams

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent. In both cases an activity diagram will have a beginning and end.

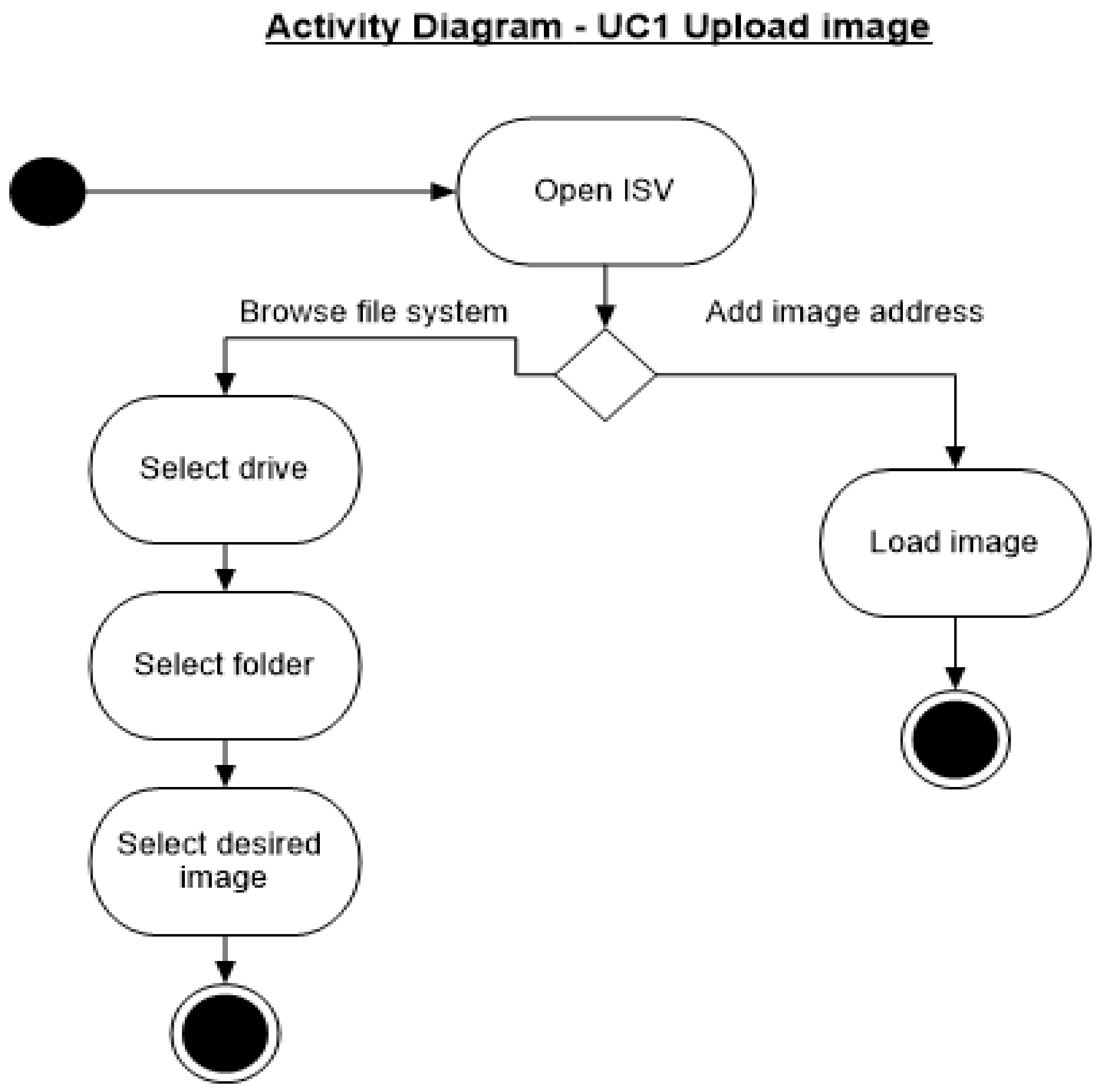


Figure 14 : Activity Diagram UC1

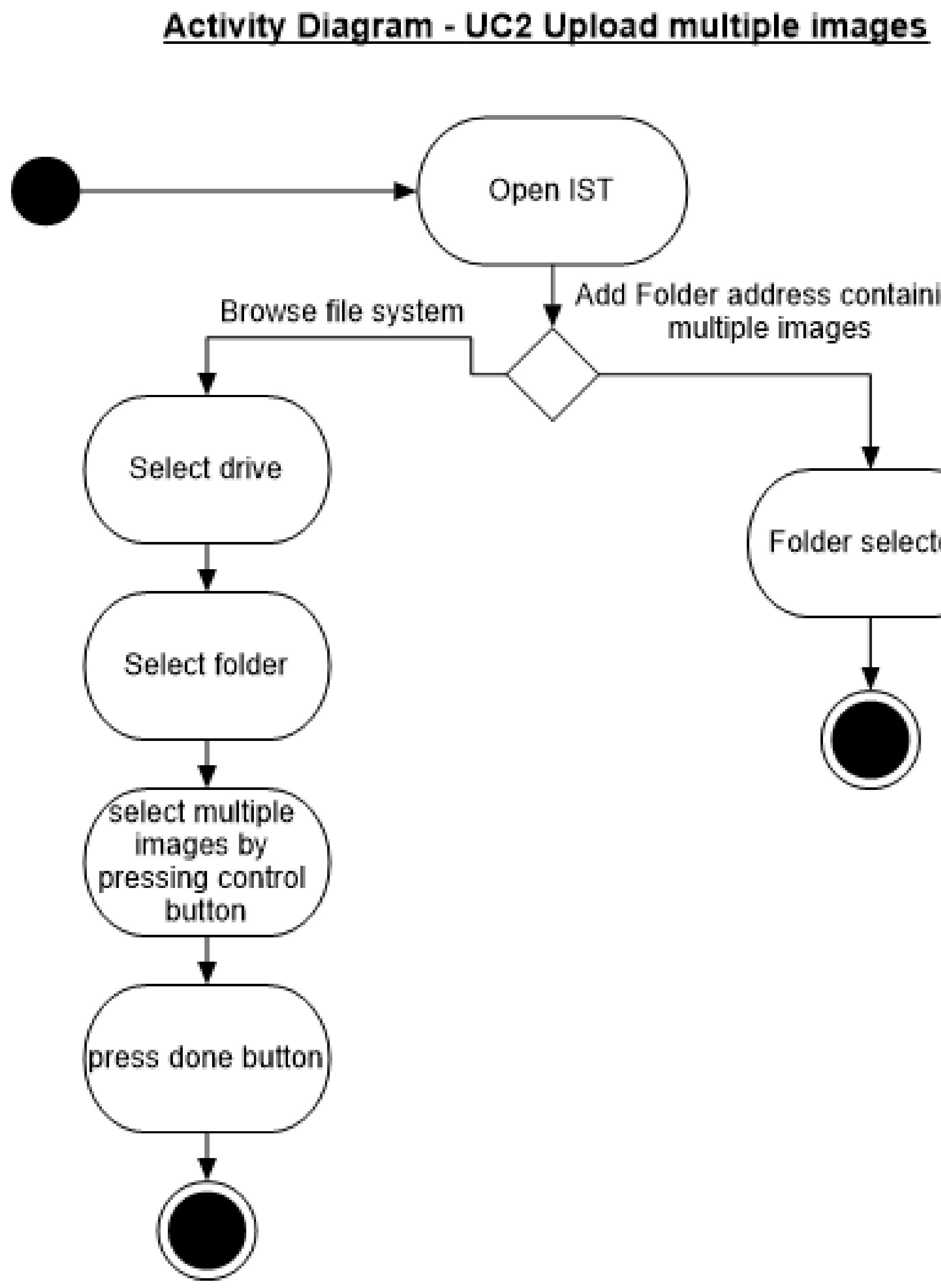

Activity Diagram - UC2 Upload multiple images
Open IST
Add Folder address containig multiple images
Browse file system
Select drive
Select folder
select multiple images by pressing control button
press done button
Folder selected

Figure 15 : Activity Diagram UC2

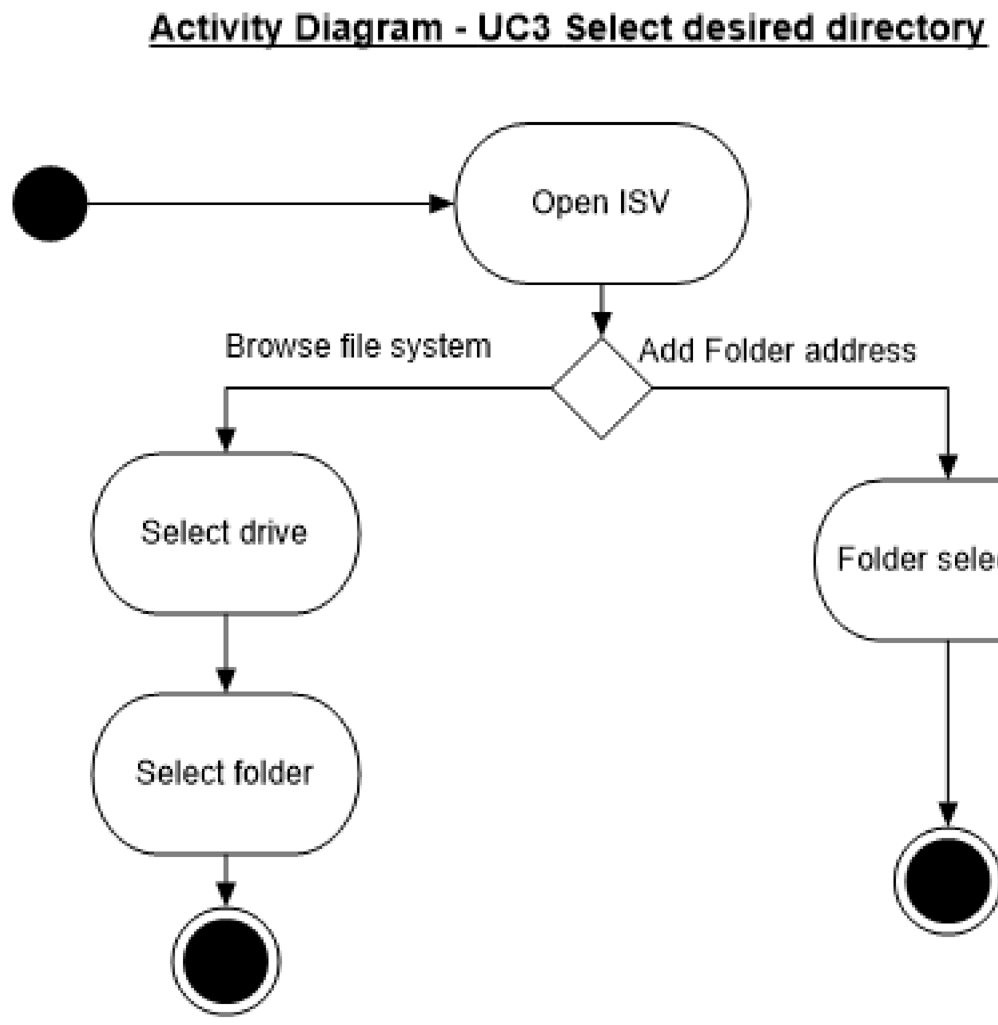


Figure 16 : Activity Diagram UC3

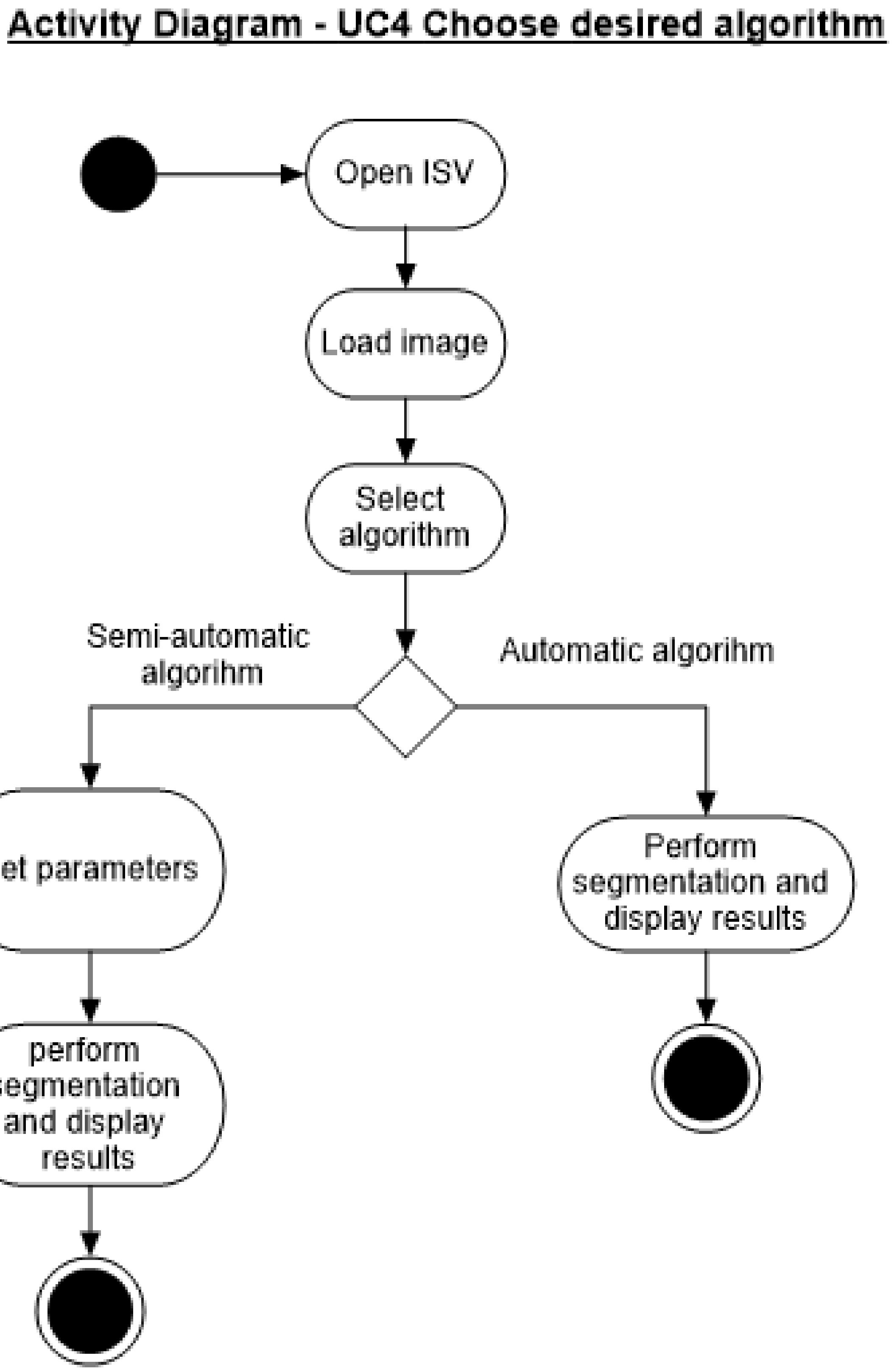

Activity Diagram - UC4 Choose desired algorithm
Open ISV
Load image
Select algorithm
Semi-automatic algorihm
Automatic algorihm
Set parameters
Perform segmentation and display results
perform segmentation and display results

Figure 17 : Activity Diagram UC4

Chapter 4 Implementation

This chapter is about the implementation of system. In this chapter selected platform, language and api's for the implementation of this project are described in detail.

4.1 Selected Platform

Matlab(MatrixLaboratory) is a multi-paradigm numerical computing environment and proprietary programming language developed by Math Works. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python.

4.2 Selected Language

While other programming languages usually work with numbers one at a time, Matlab operates on whole matrices and arrays. Language fundamentals include basic operations, such as creating variables, array indexing, arithmetic, and data types Matlab is a weakly typed programming language.

4.3 Tools

Tools used to implement this application are as described below.

4.3.1 Matlab App Designer

App Designer integrates the two primary tasks of app building – laying out the visual components and programming app behavior. Simply drag and drop visual components to the design canvas and use alignment hints to get a precise layout. App Designer automatically generates object-oriented code that specifies your app's layout and design. You can then use an integrated version of the Matlab Editor to define your app's behavior.

4.4 API's Used

Following APIs are used in this project.

This api is used to manipulate image data
https://www.mathworks.com/help/images/getting-started-with-image-processing-toolbox.html

4.5 Interfaces

Final Interfaces of system when the system is implemented completely are shown as below.

4.5.1 Segmentation Interface

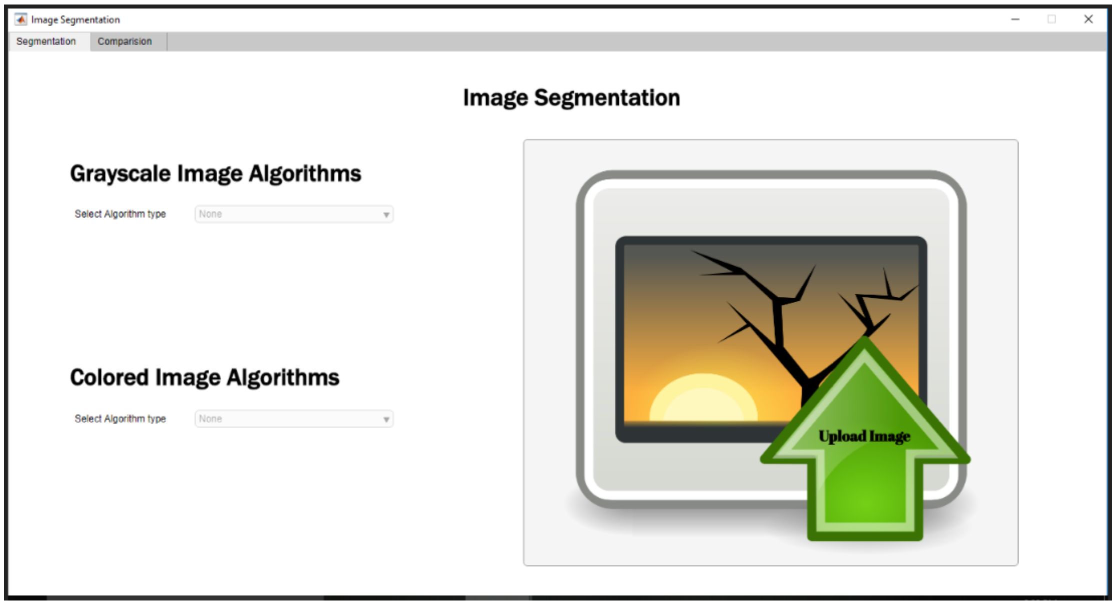


Figure 18 : Segmentation interface

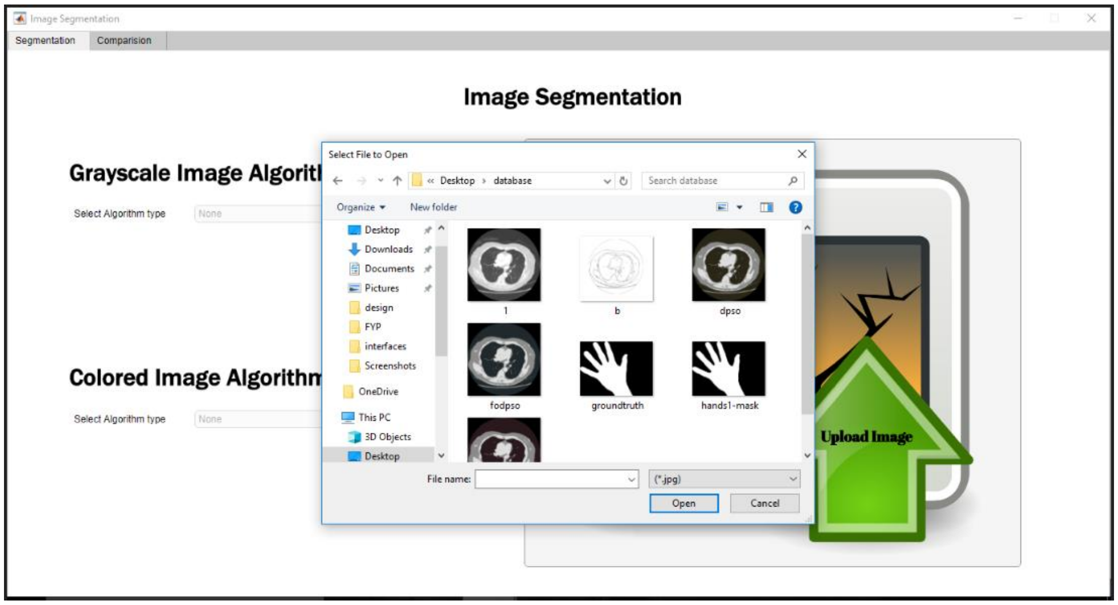


Figure 19 : uploading Image

When user run system this screen will appear. Now user has two tabs Segmentation and Comparison by default Segmentation tab is selected to perform image segmentation for further proceeding user has to upload image from windows file system upon uploading image user will be able to select available segmentation algorithms. Even if user upload colored image grayscale image segmentation algorithms will be available to him.

4.5.2 Comparison Interface

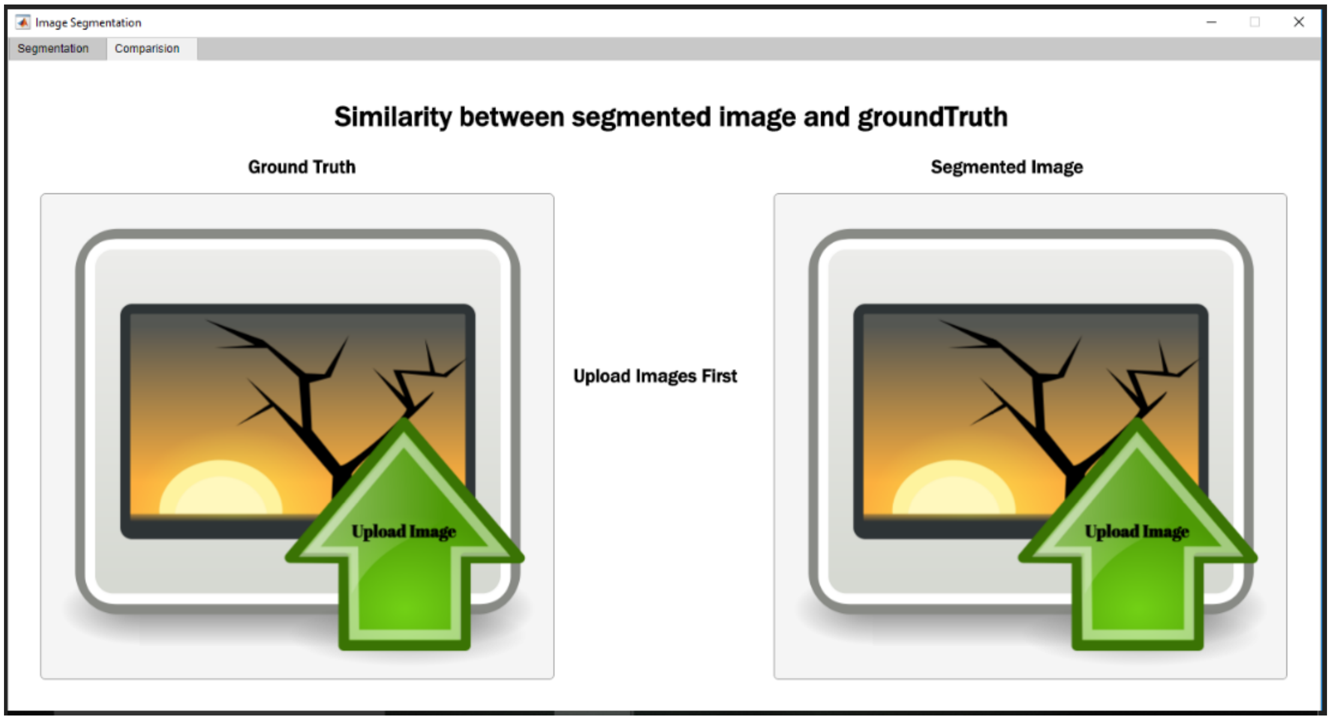
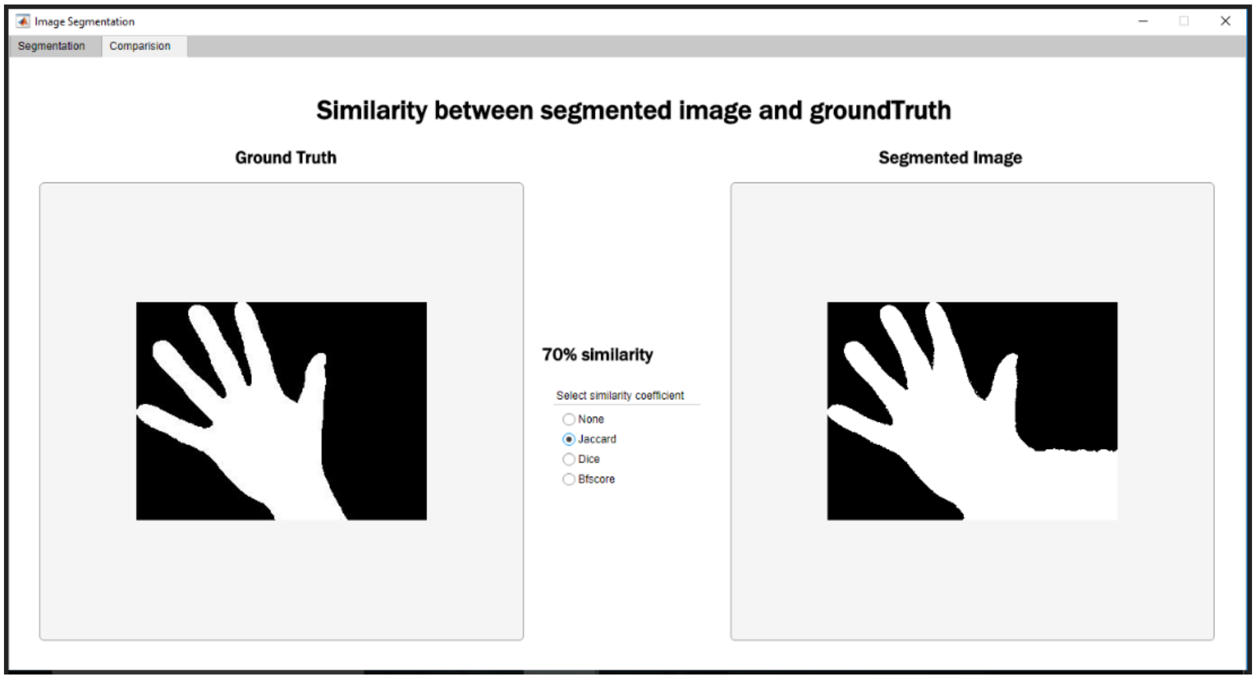


Figure 20 : Comparison interface

Comparison module is developed to check the quality of segmentation performed by any image segmentation algorithm here user has to upload both images Ground Truth and Segmented Image once user upload both images from window file system three similarity coefficient algorithms will be available to the user but these three algorithms will only work on binary, labeled or categorical images.

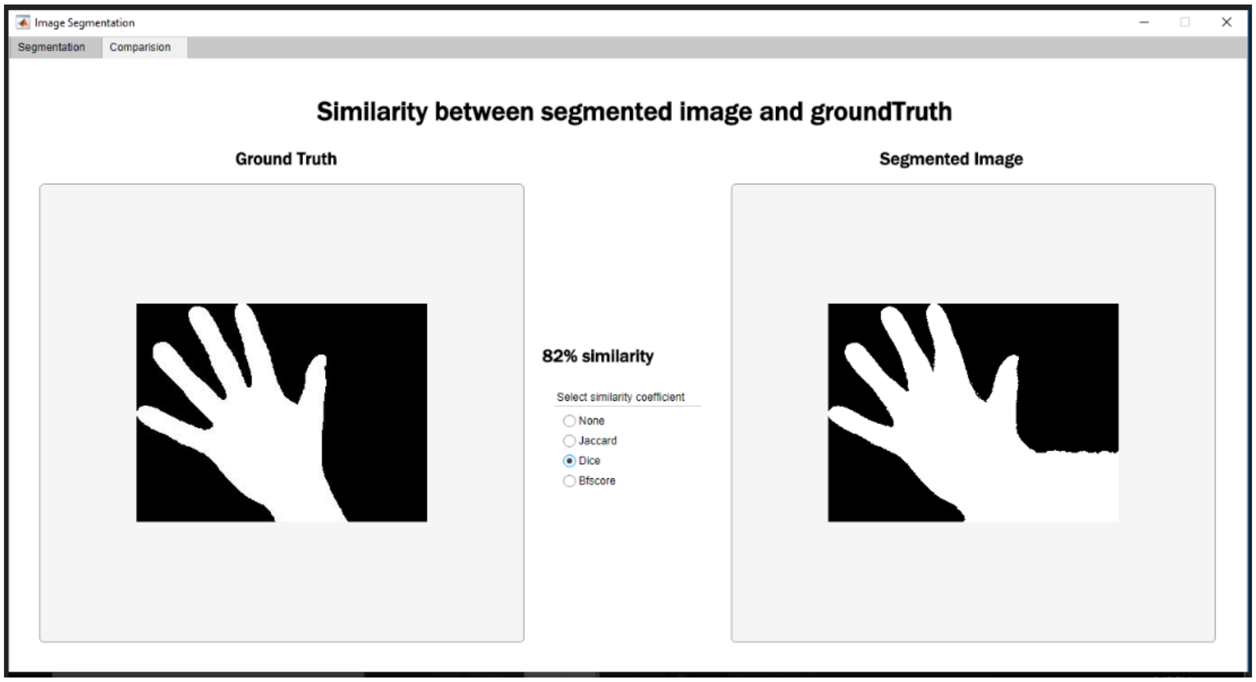
4.5.3 Similarity Coefficient Algorithm Selection Interface



70% similarity

Select similarity coefficient: None [✓ Jaccard] Dice Bfscore

Figure 21: Jaccard Similarity Coefficient



82% similarity

Select similarity coefficient: None Jaccard [✓ Dice] Bfscore

Figure 22 : Dice Similarity Coefficient

After uploading images when user select any of the given similarity coefficient algorithm the similarity in percentage will be displayed to the user.

4.5.4 Segmentation Algorithm Selection interface

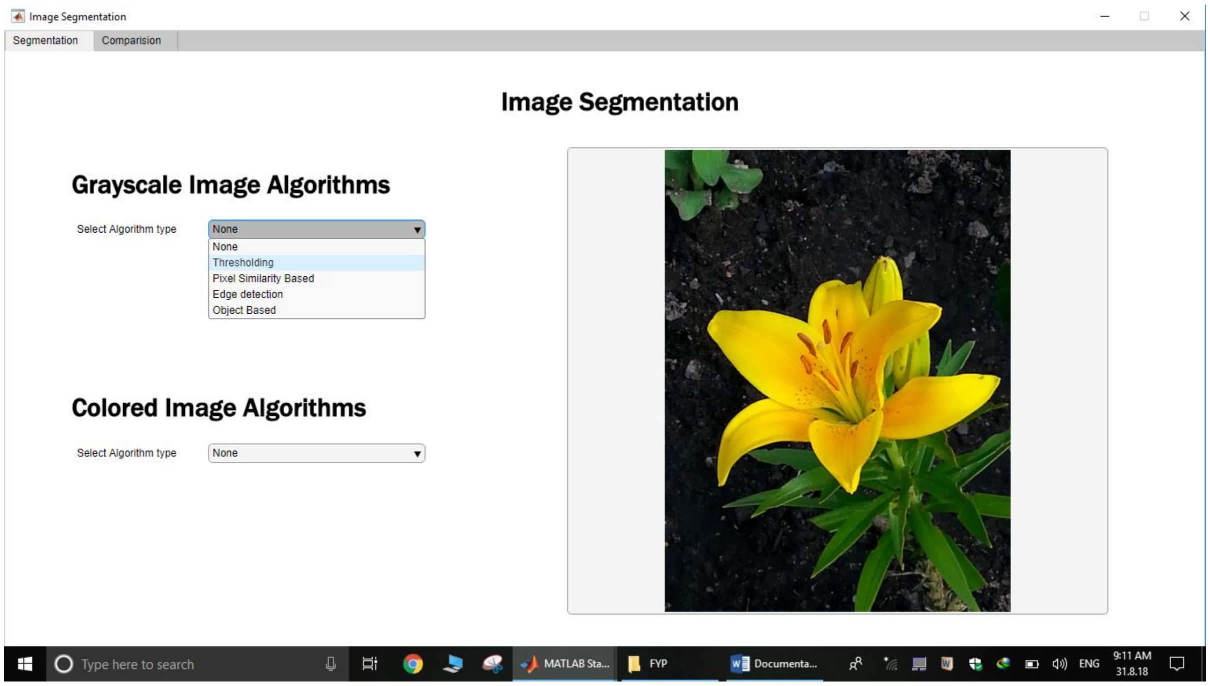


Figure 23 : Algorithm Category Selection

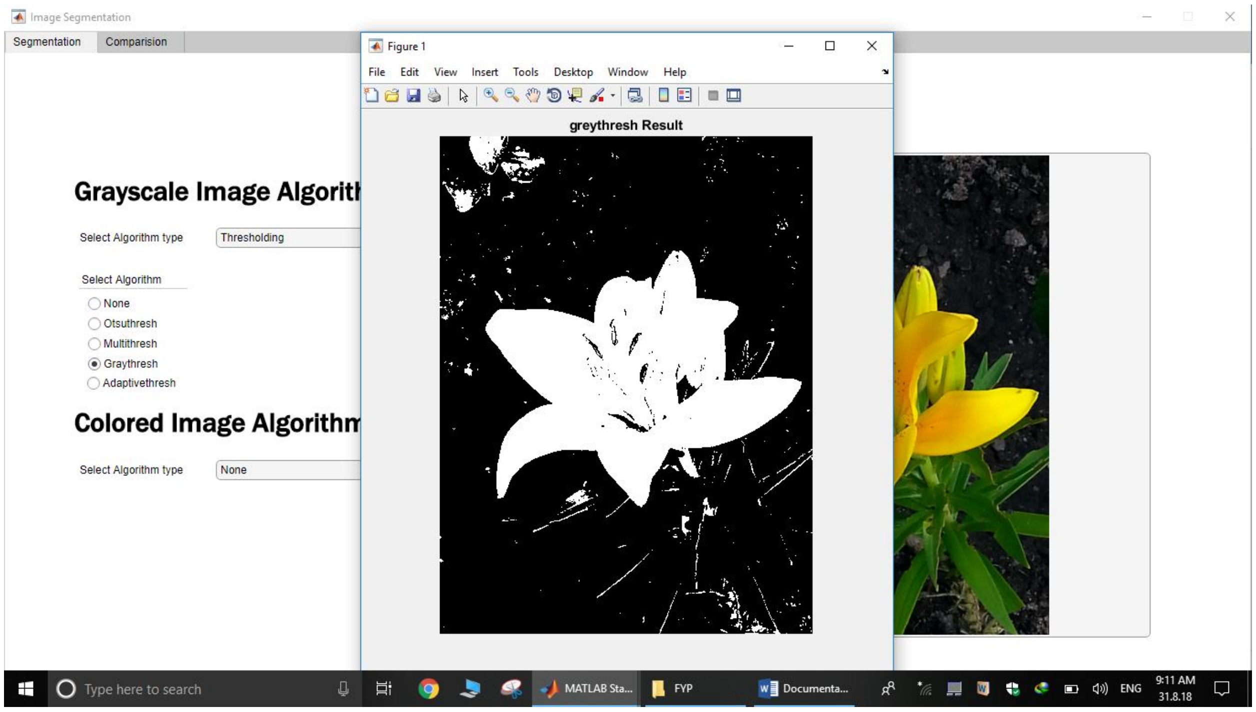


Figure 24 : Algorithm Selection

Upon selection of category all available algorithms will be displayed to the user. when user select any algorithm he has to give required parameter and segmented image be shown in new figure.

Chapter 5 Software testing

5.1 Introduction

Software testing is a process of verifying and validating a software application or program. Software testing also identifies important defects, flaws, or errors in the application code that must be fixed.

The system testing is to check the verification and validation of software. System Testing tests all components and modules that are new, changed, affected by a change, or needed to form the complete application. The system test may require involvement of other systems but this should be minimized as much as possible to reduce the risk of externally-induced problems. Testing the interaction with other parts of the complete system comes in Integration Testing. The emphasis in system testing is validating and verifying the functional design specification and seeing how all the modules work together.

There are two basics of software testing:

1. Black box testing
2. White box testing

5.2 Black box testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. When performing a black box test, a tester will interact with the system’s user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

5.3 White box testing

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called internal testing and glass box testing.

Black box testing is often used for validation and white box testing is often used for verification.

5.4 Test cases

1. Test for loading an image from file system.
2. Test for loading multiple images from file system.
3. Test for selecting desired directory containing multiple images.
4. Test to apply automatic segmentation algorithms.
5. Test to apply semi-automatic segmentation algorithms.
6. Test to compare results with ground truth and check similarity.

Table 9: Test case 1

Test Case 1: load an image from file system	
Purpose	To ensure that the image is loaded into the tool.
Setup	1. Install Image Segments Visualizer.
Instructions	1. Open tool. 2. Press upload button and select an image from file system and double click on it.
Expected Result	Image is displayed to the user on upload button and available algorithms becomes enabled to select.

Table 10: Test case 2

Test Case 2: load multiple images from file system	
Purpose	To ensure multiple images are loaded into the tool.
Setup	1. Install Image Segments Visualizer.
Instructions	1. Open tool. 2. Press upload button and select multiple image from file system and double click open button.
Expected Result	Automatic segmentation algorithms become enabled to the user for selection and message is displayed that images are loaded into memory.

Table 11: Test case 3

Test Case 3: Select desired directory containing multiple images.	
Purpose	To ensure that multiple images in a directory are loaded into the tool.
Setup	1. Install Image Segments Visualizer.
Instructions	1. Open tool. 2. Press upload button and select the desired directory containing images from file system.
Expected Result	Automatic segmentation algorithms become enabled to the user for selection and message is displayed that images are loaded into memory.

Table 12: Test case 4

Test Case 4: Apply automatic segmentation algorithms	
Purpose	To ensure that the automatic segmentation algorithms are working properly and segmented images are displayed to user.
Setup	1. Install Image Segments Visualizer.
Instructions	1. Open tool. 2. Load image. 3. Select automatic segmentation algorithm.
Expected Result	Segmented binary image should be displayed to the user.

Table 13: Test case 5

Test Case 5: Apply semi-automatic segmentation algorithms	
Purpose	To ensure that the semi-automatic segmentation algorithms are working properly and segmented images are displayed to user.
Setup	1. Install Image Segments Visualizer.
Instructions	1. Open tool. 2. Load image. 3. Select semi-automatic segmentation algorithm. 4. Set algorithm parameters.
Expected Result	Segmented image should be displayed to the user.

Table 14: Test case 6

Test Case 6: Perform comparison of segmented image with ground truth image	
Purpose	To check the quality of segmentation performed by algorithm.
Setup	Install Image segments visualizer.
Instructions	1. Open ISA. 2. Open comparison tab. 3. Upload segmented image from file system. 4. Upload ground truth image from file system. 5. Select comparison operator.
Expected Result	Similarity between segmented image and ground truth in terms of percentage is displayed to the user.

Chapter 6 Conclusion

This system is implemented for people working on image segmentation. System provide facility of performing image segmentation on multiple images once with multiple algorithms. By using this system, user can compare image segmentation results with groundtruth in order to find the best algorithm according to the corresponding dataset. Both grayscale and colored image segmentation algorithms are provided in this system. Three quantitative similarity measures are provided in this system so quality of segmentation can be judge on more than one aspect. This system is standalone application and open for everyone no authorization and authentication is required. People working on image segmentation can perform their task in better way as well as it helps them to reduce time and effort cost.

6.1 Future Work

In future we will add a module in system so that newly developed image segmentation algorithms can be added to the system by user.

References

[1] Software Engineering A PRACTITIONER'S APPROACH 5th Edition by Roger S. Pressman

[2] Otsu, N., "A Threshold Selection Method from Gray-Level Histograms," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 1, 1979, pp. 62-66.

[3] 2012 International Conference on Systems and Informatics (ICSAI2012) by Yantai Shandong

[4] Interactive foreground extraction using iterated graph cuts by Carsten Rother, Vladimir Kolmogorov, Andrew Blake SIGGRAPH '04 ACM SIGGRAPH 2004 Papers

[5] Multilevel Image Segmentation Based on Fractional-Order Darwinian Particle Swarm Optimization by Pedram Ghamisi, Student Member, IEEE, Micael S. Couceiro, Student Member, IEEE,

[6] https://en.wikipedia.org/wiki/Canny_edge_detector

[7] A. Protiere and G. Sapiro, Interactive Image Segmentation via Adaptive Weighted Distances, IEEE Transactions on Image Processing, Volume 16, Issue 4, 2007.

[8] https://www.mathworks.com/help/images/ref/activecontour.html?s_tid=doc_ta

[9] https://www.mathworks.com/help/images/ref/imsegfmm.html