IMAGE REGISTRATION AND FUSION APPLICATION

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Shreya Singh

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

November 2021

Fargo, North Dakota

North Dakota State University
Graduate School

Title

IMAGE REGISTRATION AND FUSION APPLICATION

By

Shreya Singh

The Supervisory Committee certifies that this *disquisition* complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Kenneth Magel
Chair

Dr. Simone Ludwig

Dr. Zhao Zhang

Approved:

November 18, 2021
Date

Dr. Simone Ludwig
Department Chair

ABSTRACT

Image Registration and Fusion is a process of manipulating two or more input images through a combination of geometric transformation operations and fusion rules to obtain the information from the input images into one fused image. Currently, most existing image registration and fusion techniques are appropriate for grayscale images. There are a few techniques which support colored images but cannot be used for a large input data set.

The aim of this research is to propose an image registration and fusion technique that supports large input data sets of both grayscale and colored images. To this end, we explored various existing image registration and fusion techniques. Based on the implementation results, we developed Image Registration and Fusion Application (IRFA). IRFA is a MATLAB based application which performs automatic image registration and fusion of large input data set of colored images, thus solving the two main challenges posed by existing techniques.

ACKNOWLEDGEMENTS

This research would not have been possible without the constant support and guidance of Dr. Kenneth Magel. He has been an extremely valuable resource during my entire Master's degree program. I would also like to express my sincere gratitude to my committee members Dr. Simone Ludwig and Dr. Zhang Zhao for their time and support on my research. Thirdly, I would like to thank the faculty and staff of the Computer Science Department for their unconditional support throughout my master's program and my research. Special thanks to Dr. Paulo and Department of Agriculture and Biosystems Engineering for allowing me to use their image data as my input data set for this research. Finally, I am grateful to my parents who are the reason for all my achievements and have supported and motivated me throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF CODE SNIPPETS

CHAPTER 1. INTRODUCTION

1.1. Overview

With the rapid emergence of artificial intelligence (AI) and machine learning, AI Chatbots and virtual assistant which appeared to be a thing of the future has been knocking right at our doors. AI has changed the face of software development, it has been gaining importance in almost every sector now, be it government – changing public administration, reducing paperwork or medical imaging and health care centers – making services more affordable and accessible, for example there are various medical Chatbot applications available which can accurately guide you what precautions and medicines to take just by knowing your symptoms. Have you ever wondered how we are able to unlock our phones just by our face, how exactly face recognition and pattern recognition works? How exactly Siri is almost able to answer almost all our question. Recently, Walmart changed their self-checkout machines, now if you are buying tomatoes and you must select from the list, the default list shows all the fruits and vegetables which are red in color, how exactly is this working? The answer is AI and Image Processing.

Artificial Intelligence combined with Image processing can be considered as a driving force powering various of our industries. The one of the biggest consumers of the above combination is medical industry and health sectors for detection of tumors, finding their exact location, for revealing cardiovascular related abnormalities, wound healing assessment etc. With advances in image processing in medical industry, other sectors such as navigation and automobile industry, military, natural resource management are also exploring image processing techniques to enhance image road vision during extreme weather conditions, automated target recognition in battlefield, environmental monitoring. In past few years, a spike has been observed for use of these image processing techniques in agriculture industry coupled with deep learning techniques and

Artificial Intelligence for application like weed detection, plant disease detection, prediction of best crop results.

Image Processing can be defined as a process of manipulating image which contains a series of steps, such as identifying out main region of interest, denoising it, enhancing the image, combining the information obtained from images taken at an interval of time. One of the important steps of Image Processing are – Image Registration and Image Fusion which in a single statement could be explained as combing the information from various sources.

For this thesis we are focusing on image registration and fusion of colored images using MATLAB. We have tried implementing existing techniques for our data set in MATLAB and demonstrated why there was a need of proposing a new method for image registration and fusion of colored images. Finally, we have developed an application in MATLAB that registers and fuses image using our proposed method.

1.2. Image Registration

Image Registration [26] is a process of transforming an input image into same coordinate system of the reference image. These two images could be of same object taken at different time intervals, or taken from different sources, or from different angles or viewpoints. This process of image alignment helps in better interpretation and comparison of information obtained from individual images.

In this process, we generally take two images, moving image – the image that needs to be registered or aligned and the fixed image - the image which is used as a reference image. Depending on how the images were taken, different geometric transformations could be used for image registration. For example, if the images were taken from two different viewpoints, we could use one of the simple geometric transformations – cropping, rotating, resizing, etc. If the images

were taken from two different sources, we can use one of the complex geometric transformations such as affine, projective, similarity etc.

1. Affine: Affine transformation is a linear geometric transformation which is generally used to correct distortions caused from poor angles from cameras. This transformation preserves points of the moving image, ratio between these points, parallelism, and creates a new 2-D/3-D affine object which has pixel intensity mapped from pixel intensity of each point of the moving image.
2. Projective: Projective transformation is totally opposite to affine transformation. In projective transformation, angles parallelism, point to point ratio is not protected. If suppose we take a rhombus and use projective transformation, the resulting figure could be any quadrilateral according to the chosen parameters and rules applied during transformation.
3. Similarity: Similarity Projection as the name suggest, ensures that the input image and the transformed image are similar in nature, angles are invariant, but the translation, scale, and rotation of the image could be distorted.

Image registration is a pre-requisite process for image fusion. We would be learning more about image fusion in the next section. Depending on the application of resultant fused image, image registration could be of different types.

1. Intensity based registration: In this type of registration, we register the moving image with respect to the similarity metric obtained from fixed image. This similarity metric is either maximized or minimized according to the transformation applied using an optimizer object. This process is generally iterated quite a few times to obtain best results.

2. Feature Based Registration: In this type of registration, initially we extract features like corners, sharp edges, blobs from our fixed image and then find matching features in the moving image and register it using the object created from the points obtained from matching features.
3. Control Points: In this type of registration, we manually select points in both images and determine the parameters required for transformation and register the moving image accordingly.

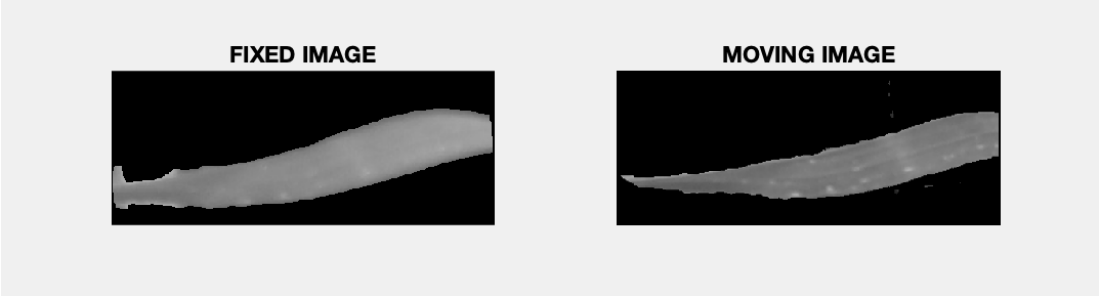

Figure 1. Image Registration Example

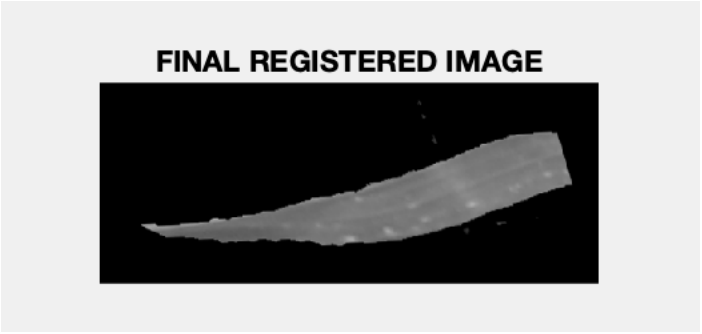

Figure 2. Image Registration Example

Let's try to understand the process by taking an example. In the above figure, there are two images fixed and moving which are taken from two different sources and angles. We want to align our moving image with respect to fixed image so that they can be easily fused later into a single image. The below image is our registered image, you can see our moving image has been transformed to the same coordinate system of fixed image.

1.3. Image Fusion

Image Fusion [25] as the name suggest is a process of fusing two or more images of an object into a single image. The sole purpose behind this process is to obtain information content from both the images into one fused image, thus reducing the amount of data by two. This enhanced fused image could be later use for different application such as medical imaging – finding exact locations of tumors, training, and testing AI models, etc.

Image registration is a precedent process for image fusion. In this process we take our registered image obtained from image registration and the fixed image and fuse them together into a single image using user-defined fusion rules. These fusion rules could be determined using various techniques such as Wavelet transformation, Laplacian transformation, fuzzy logic etc. Most of the image registration and image fusion techniques are applied to grayscale images. The below figure is an example of image fusion. The fixed image from Figure 1 and registered image from Figure 2 are fused together to combine information from both the images into one.

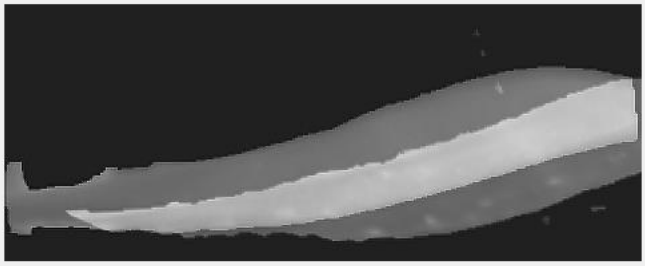

Figure 3. Image Fusion Example

1.4. MATLAB

1.4.1. Overview

MATLAB is a high-performance language used for performing linear algebra and numerical computation especially when dealing with matrix and vector formulations. MATLAB stands for Matrix laboratory. It has applications across several functional fields including finance, business, and agriculture sector. It provides a command line interface and text editor for carrying out the operations. It coalesces computation, visualization, and programming in an environment where it is easy to use while the problem statements and their solutions are presented in well-known mathematical notation. Some of the use case include developing various algorithms, analyzing enormous amounts of data for exploration and visualization and for developing Graphical user interface. MATLAB Editor also known as M-File/MATLAB Script file is used for writing a sequence of executable MATLAB commands for performing tasks. It also comprises of Command window for executing the code that are written by the user. MATLAB is a licensed software

The MATLAB system comprises of five parts. The MATLAB language, which is a high-level array language comprising of functions, data structures, control flow statements with the object-oriented programming principles, MATLAB working environment that has all the tools and abilities that are needed for the programmer of MATLAB to work on, A graphics system to design and handle two-dimensional, three-dimensional graphics using high level commands. It also provides low level command which allows the programmer to customize the graphics appearance according to the needs. The mathematical function library which has a collection of computational algorithms that includes elementary functions (sum, sine, cosine) as well as complicated functions like matrix inverse, eigenvalues, Bessel functions and Fourier transforms. Finally comes

Application Program Interface which allows us to code in C and Fortran Programming languages. Apart from these it also has calling routines from dynamic linking and for performing reading and writing MAT-files.

Figure 4 demonstrates the default environment of MATLAB. It comprises four main sections

1. Current folder: This shows the recent directory used in MATLAB program.
2. Workspace: The variables that are stored in MATLAB whenever a command is executed is displayed in Workspace window.
3. Command Window: As the name suggest, here we enter all our commands.
4. Command History: This displays all the commands that were executed in the command window.

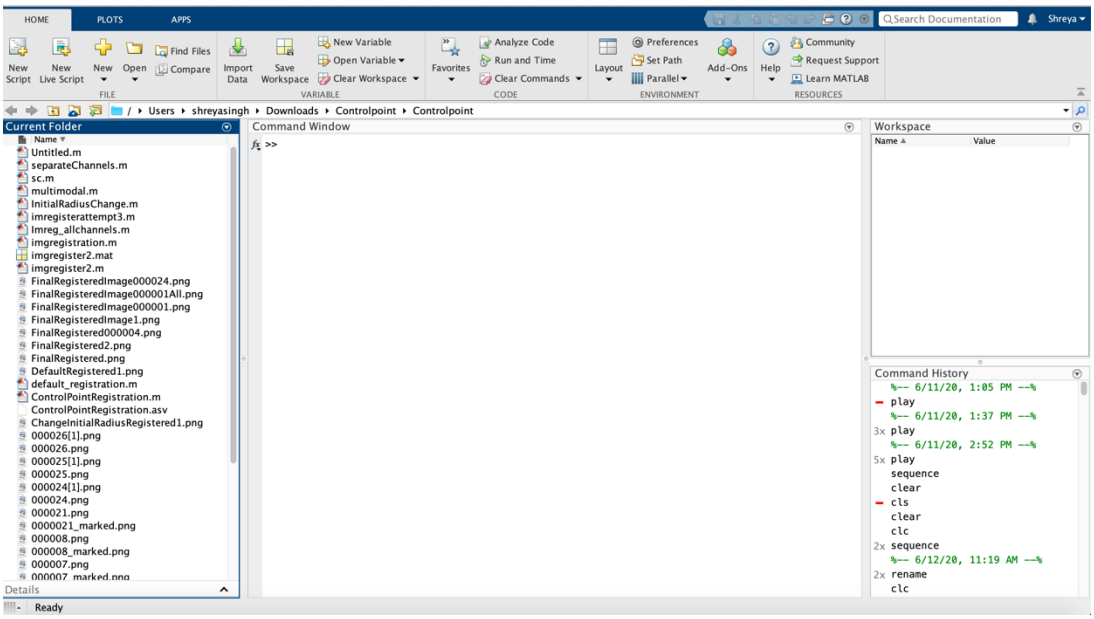

Figure 4. MATLAB Environment

1.4.2. Image Representation in MATLAB

Everything in MATLAB can be represented in form of matrix as the basic data structure in MATLAB is a rectangular matrix. The elements of these matrix could be real or complex.

Therefore, a grayscale image could also be represented in form of a two-dimensional array or matrices, each element of this array corresponds to each pixel of the image. Let's imagine an image as grid of 400 rows and 500 columns, with each cell having a color, now this could be represented by 400x500 matrix. A colored image also called RGB image could be represented in form of a three-dimensional array, each 2D plane represent the pixel intensity of R, G and B color. Image processing is simply manipulation of these matrix represented data for getting and enhanced image.

1.4.3. MATLAB App Designer

MATLAB has an interactive development environment for designing an application layout and how that should be programmed based on its behavior [25]. It's also integrated with complete version of the MATLAB editor and a humongous list of interactive user interface components. In addition to these, it also offers a grid layout manager for organizing the user interface, automatic reflow choices to ensure that our application will respond to the changes in the screen size and adjust its resolution accordingly.

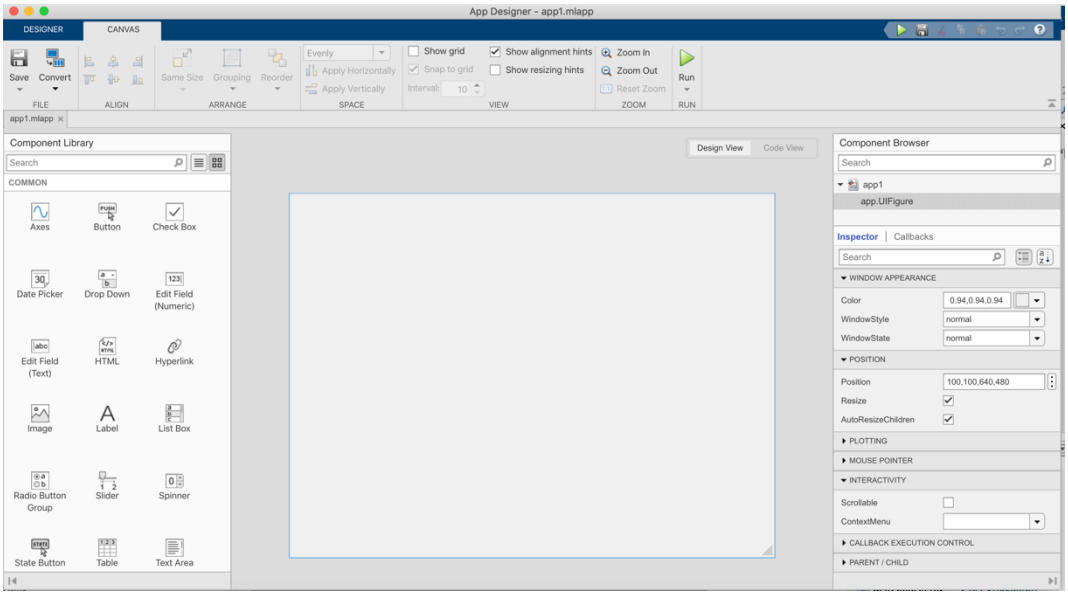

Figure 5. MATLAB App Designer Environment

The component library of the MATLAB includes container and instrumentation components. To view the code that has been created for the blank applications switch to code from the design view. The methods in this are eventually set for all the application functions including the call back functions. The App designer creates the file with mlapp extension.

MATLAB App Designer provides two separate views

1. Design View: Here you can create UI of the application by drag and dropping components from various available option.
2. Code View: In this view, you can add call backs to the components that you have added to UI of the application.

1.5. Problem Statement and Requirement Analysis

We have been intensively using image processing techniques in various fields such as medical science, satellite and navigation, military, and combat. Recently, there has been an increase in demand of image processing techniques in agriculture with only one motive to improve the quality and quantity of the harvest. Image processing techniques are being used as pre-requisite for developing, training, and testing frameworks, applications, Artificial Intelligence Models. In an agricultural state like North Dakota, these models can be used to detect weeds that take up extra space and resources, to find sick parts of the plants, kernel detection, disease detection.

Generally, most of these AI models use image processing techniques for detection of various objects. For example, medical sector uses image fusion techniques to detect the exact location of tumors, satellite and navigation uses it for detecting volume of traffic over different routes. Clearly process of detection doesn't depend on the nature of the images captured, whether they are grayscale in nature or colored. But let's talk about disease detection in crops, it could be only detected either if the leaves of the crops or physically damaged or if there is a color loss.

Most of current research revolves around image processing of grayscale images. The process of obtaining information from the input images rely mainly on converting color images into grayscale images. As talked before, image processing comprises of various steps for to achieve maximum information from the input images. These steps are noise removal, image segmentation, image registration, image fusion etc. The noise removal and image segmentation step remain inconsiderable to the nature of the image, but when we talk about image registration and image fusion, only few techniques support color images.

This research was mainly conducted to prepare most informative input image data set which would later be used to train and test AI models that enhance detection of disease such as tanspot and rust in various crops. The two challenges that needed to be accomplished during the research were:

1. Colored images: The proposed image registration and fusion method should protect the color information of the images, i.e., the resultant fused image should have color information from both the parent images.
2. Large Data Set: The other challenge was that the proposed technique should support automatic image registration and fusion for a large data set.

The few of the existing image registration and fusion techniques that did support color images were not automated, had to be manually done by the user, one at a time and the ones that did allow automatic registration and fusion did not support color images. The existing methods are explored more in Chapter 2, section 2.2. and their implementation using our input data set is done in Chapter 4.

1.6. Application

Image Registration and Fusion Application (IRFA) has been developed in accordance with the challenges described in section 1.3. Problem Statement and Requirement Analysis. The application provides the user a very simple and easy to use interface. IRFA has been developed using MATLAB App designer. MATLAB offers various add-on toolbox such as Image Processing Toolbox, Computer Vision Toolbox, Wavelet Toolbox, and Fuzzy Logic Toolbox. These toolboxes have in-built functions that helped designing the Application's algorithm used for image registration and fusion of colored images.

IRFA supports image registration and fusion of both colored and grayscale images. It also allows batch image registration and fusion. IRFA uses intensity-based automatic registration technique modified for colored images to register the input moving image with respect to input fixed image. Once the moving image has been registered, the new registered image and input fixed image is fused together using wavelet transform-based technique. This would be discussed in more detail in coming chapters.

Currently this application does not support preliminary steps of image processing such as noise removal or image segmentation. The input images must go through all those operations before they are uploaded in IRFA for image registration and fusion.

1.7. Organization of the Paper

The remainder of the paper is as follows: Chapter 2 provides a literature review of the existing work; detailed explanation of techniques currently being used. Chapter 3 explains details of application design by documenting various diagrams. It also focuses on the main logic behind the proposed image registration and fusion technique that the application uses for registering and fusing image. Chapter 4 provides the implementation of existing Image registration and fusion

techniques with our input data set and describes the rationale behind the proposed methods. Chapter 5 gives an overview of implementation of the developed application with our input data set and its functionalities. Chapter 6 shows results of various tests performed to analyze the logic against the limitations explained in Chapter 1. It also covers application's testing in terms of performance, amount of input data, etc. Chapter 7 summarizes the work done, and documents conclusions, limitations of the current system and suggestions for future work.

CHAPTER 2. BACKGROUND

2.1. Overview

In many years, several Image Processing tools, and techniques have emerged that have immensely helped and improved medical image representation. The purpose of these tools is to enhance the image information, characteristic, etc. These tools perform various operations over input images such noise removal, image segmentation, image registration, image sampling, image fusion, etc. Some of the medical image processing tools are Elastix, Camino and Gimias which have tremendously contributed towards improving clinical analysis.

Image Fusion is one of the most important process in Image processing. The process can be recognized as preliminary steps for preparing usable input data sets. The purpose of this process is to assess the information at each pixel of the input images and integrate the information into a single fused image. Since the fused image is a combination of two or more images, hence more accurate and informative as compared to individual input images.

2.2. Literature Review

2.2.1. Image Processing Toolbox

MATLAB consist of various in-built functions and add-ons that provide an ideal environment for interacting, exploring, designing, and visualizing data. One of such add-ons is Image Processing Toolbox [25]. It is a comprehensive set of algorithms and workflow of applications designed for enhancement of an image. It allows the process of noise removal, image segmentation, image registration, image fusion, etc. It also provides the ability to design and develop algorithms with the help of various utility functions for batch processing of images. Most of the existing and on-going work is contingent on Image Processing Toolbox in MATLAB. This toolbox supports various types of images such as binary, grayscale, true color which can be

represented by 2D arrays, where each element of the array stores information about a pixel in the image. Let's take a quick look to various Image Processing Toolbox utility functions that we would be using later for implementation of existing and proposed methods in Chapter 4 and Chapter 5.

Table 1. Image Processing Toolbox Methods

Functions	**Description**
imread	Reads an image to workspace.
imshow	Displays the image.
imwrite	Write the final processed image to a disk file.
rgb2gray	Converts a colored image into grayscale.
imresize	Resize an image.
imwarp	Transforms the numerical image according to given geometric transform and returns the transformed image.
imregister	Used for intensity-based image registration.

2.2.2. Computer Vision Toolbox

MATLAB consist of various in-built functions and add-ons that provide an ideal environment for interacting, exploring, designing, and visualizing data. One of such add-ons is Computer Vision Toolbox [25]. It is a comprehensive set of algorithms, computational functions, and applications for designing and testing computer vision. It allows the process of object detection and tracking, feature detection, feature extraction as well as matching. It also provides a way for automating the calibration workflows for various cameras such as single, stereo, and fisheye cameras. Considering for the support of 3D visions, it has visual and point cloud Simultaneous Localization and mapping, stereo vision, extracting the structure from the motion and point cloud processing. We can use multiprocessor and GPU's for accelerating the algorithms. It can also support code generation for C/C++ to integrate with the existing code, prototyping the desktop and embedded vision system deployment. Let's take a quick look to various Computer Vision Toolbox utility functions.

Table 2. Computer Vision Toolbox Methods

Functions	**Description**
detectSURFFeatures	Detect SURF features and return SURFPoints object.
extractFeatures	Extract interests point descriptors
matchFeatures	Find matching features.
SURFPoints	Object for storing SURF interest points

2.2.3. Wavelet Toolbox

Wavelet Toolbox includes apps and functions for signal and image analysis and synthesis. You can detect anomalies, change points, and transients, as well as denoise and compress data. Wavelet and other multiscale techniques can be used to analyze data at various time and frequency resolutions, as well as to decompose signals and images into their constituent parts. Wavelet techniques can be used to reduce dimensionality and extract distinguishing features from signals and images to train the machine and deep learning models [25].

With the Wavelet Toolbox, you can interactively remove signals, perform wavelet and multi-resolution analysis, and generate MATLAB code. The toolbox includes algorithms for discrete and continuous wavelet analysis, wavelet packet analysis, multi-resolution analysis, wavelet scattering, and other multi-scale analysis.

So, Wavelet Toolbox helps in wavelet transformation of an image for better analysis, but what exactly is wavelet transform. Wavelet transformation is breaking up a signal or image into shifted and scale version. It is a combination of two processes.

1. Smoothing: The process of smoothening involves subjecting an image or input signal to either a high pass frequency domain filter which reduces low frequency components and strengthens the high frequency components, or a low pass frequency domain filter which

reduces high frequency components and strengthens low frequency components, or a combination of both.

2. Sampling: When a signal or image goes through process of smoothening, redundant information is obtained. According to Nyquist criteria, even if the sample size is reduced by half, the information content remains the same.

In Wavelet transform, we pass the image through combination of these high pass and low-pass filter and reduce the sample size by two at each step to obtain an approximation version of image, horizontal, vertical, and diagonal edges. These concepts are used to create image fusion rules which helps in designing the Image fusion algorithm. Let's take a quick look to various Wavelet Toolbox utility functions that we would be using later for implementation of existing and proposed methods in Chapter 4 and Chapter 5.

Table 3. Wavelet Toolbox Methods

Functions	**Description**
dwt2	Used for carrying out single wavelet decomposition.
idwt2	Used for carrying out single level reconstruction.
wdenoise2	Wavelet image denoising.
wdencmp	Wavelet denoising and compression.

2.2.4. Registration Estimator App

Registration Estimator Application [2] is a MATLAB application which when combined with Image Processing Toolbox and Computer Vision Toolbox supports various kinds of Image Registration – Feature-Based Registration, Intensity-Based Registration and Non-rigid Registration. This application allows the detection of features like sharp corners, blobs, regions of uniform intensity from a reference or fixed image, and then registers the moving image with respect to these detected features. Similarly for intensity-based registration, the application

registers images with both similar and different brightness and contrast. Following are some registration techniques offered by Registration Estimator App.

Table 4. Registration Techniques Offered by Registration Estimator App

Techniques	**Description**
SURF	Used for detection of blobs.
FAST	Used for detection of sharp corners features.
MSER	Used for detection of uniform intensity.
Monomodal	Used for registration of images with similar brightness and contrast.
Multimodal	Used for registration of images with different brightness and contrast.

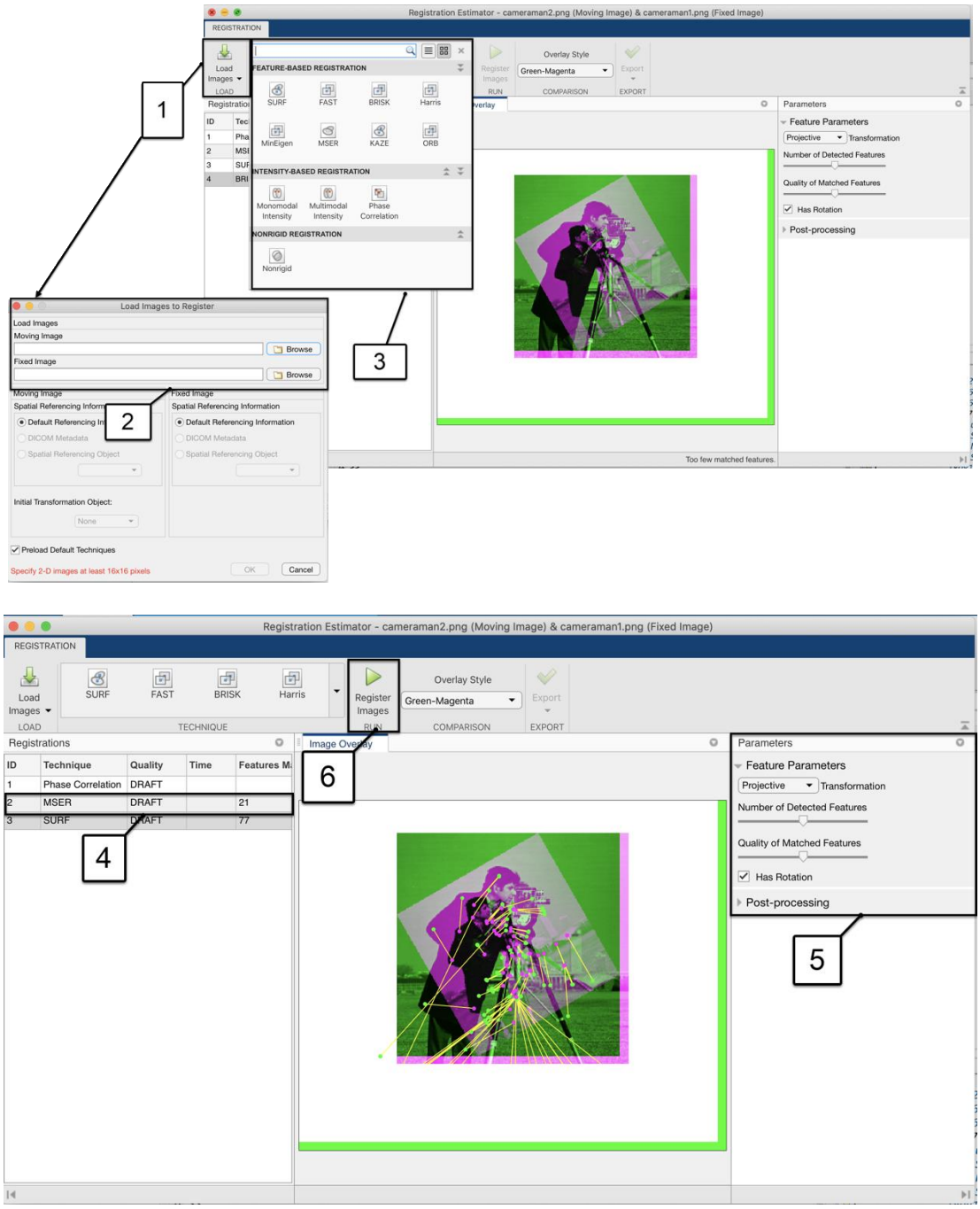

Figure 6. Registration Estimator App

The above picture demonstrates the Registration Estimator Application interface. The load image icon in the corner [Figure 6][1] allows the user to import a moving image i.e., the one that needs to be registered, and a fixed image that would serve as a reference into the workspace of the application [Figure 6][2]. Both the images would be first converted to grayscale and then displayed

in different colors in central panel of the application – Image Overlay. The default color settings for the fixed and moving image is green and purple which could be changed as well. You can select from a variety of techniques offered by the application to perform image registration from the Techniques panel right to the load images icon. Once you have selected the different techniques, you can track the matched features between the moving image and fixed image from each of those techniques in the left panel of the application [Figure 6][4]. The application allows to modify parameters like type of transformation, quality, and number of matched features in the Parameters panel [Figure 6][5]. The yellow lines depict the matched features between the moving image and fixed image. Once you are satisfied with the techniques and found more than 1 matched feature, you can register the moving image and export it from the workspace.

Most of the medical researchers have used Registration Estimator App to detect features and contrasting information from different images taken over an interval of time. It helps in multi-modal imaging - determining the exact location of tumors or masses from MRI-CT, MRI-PET scan taken from different angles.

2.2.5. Control Point Registration

This technique involves transformation of a moving image using parameters determined by point mapping between fixed and moving image. As the name suggest in this technique, first we select minimum of four distinguishable points that identify the same features of the two images, and then geometric mapping and transformation is done using the locations of these points. Since these selected points controls the registration process, therefore the technique is named as Control Point Registration. Unlike Registration Estimator App, this technique allows the registration of both grayscale and colored images.

The following image is taken from Mathworks[25]. It is a brief graphic illustration of the process involved in Control Point Registration.

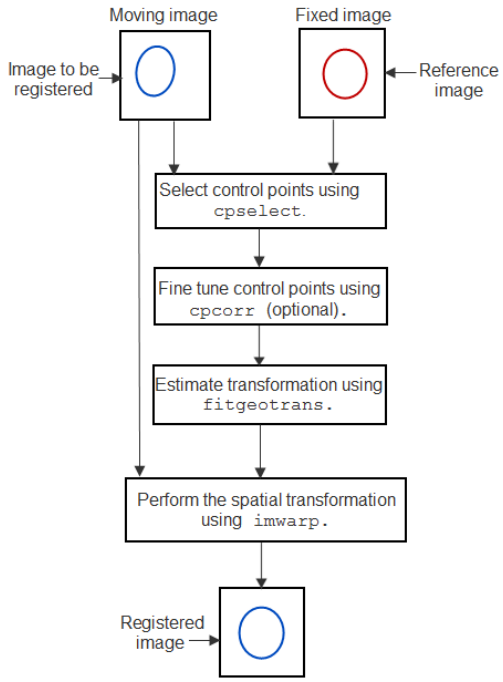

Figure 7. Control Point Registration Process

cpselect(): cpselect function starts the Control Point Selection Tool that allows the user to select control points, move them, delete them. Few of the key points to remember here is that a minimum of four control points is needed for image registration and these points should be at a considerable distant from each other. Once these control points have been selected, they can be exported to the workspace.

cpcorr(): cpcorr function allows further tuning of the selected control points in a moving image also known as moving control points, with respect to fixed control points using the normalized cross-correlation between the two images. This function allows modifying the location of the moving control points up to 4 pixels and returns the adjusted moving control points.

fitgeotrans(): fitgeotrans function returns a transform object (tform) inferred from the selected moving and fixed control points which enables forward and inverse transformation. The

transform object also depends on the selected type of transformation such as affine, similarity, projective, etc.

iwarp(): imwarp functions finally transforms the moving image using the tform object returned by fitgeotran. The function applies 2-D transformation stored in tform to all 2-D planes of the moving image and returns the transformed registered image

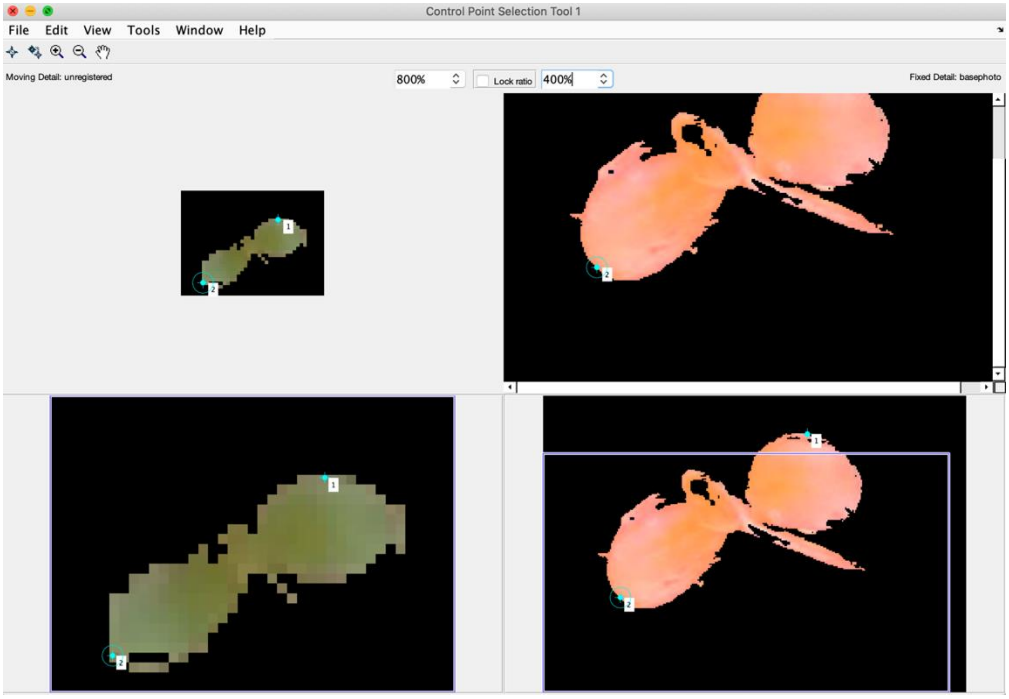

Figure 8. Control Point Selector Tool

The above picture demonstrates the Control Point Selector Tool. It is divided into 4 panels, the top two panels display the moving and the fixed image, and the lower panels displays a zoomed in version of the same image. The user selects the control points in both the images and once selection process is completed, simply closing Control Point Selector Tool exports the selected points to the workspace.

2.2.6. Automated Feature Detection and Matching

As the name suggest, this technique automatically determines the geometric transformation between the moving image and fixed image. This technique is generally used when the two images

are different with respect to rotation and scale. This technique requires both Computer Vision and Image Processing Toolbox.

The subsequent flow chart describes the process involved in automated feature detection and matching technique [1]. Firstly, we load the moving image and fixed image into the workspace and detect the features in each of the image using detectSURFFeatures function. Using these detected features, we extract feature descriptor using extractFeatures function. Feature Descriptors are encoded series of number that differentiates features from each other. The next step is to match the extracted features and retrieve the location of points of the corresponding matched features. The retrieved points help in creating a geometric transformation using estimateGeometricTransform2D function which is used to register the moving or distorted image. All the above-mentioned functions are provided by Computer Vision and Image Processing Toolbox. Another key point to remember here is that the two input images should be grayscale in nature.

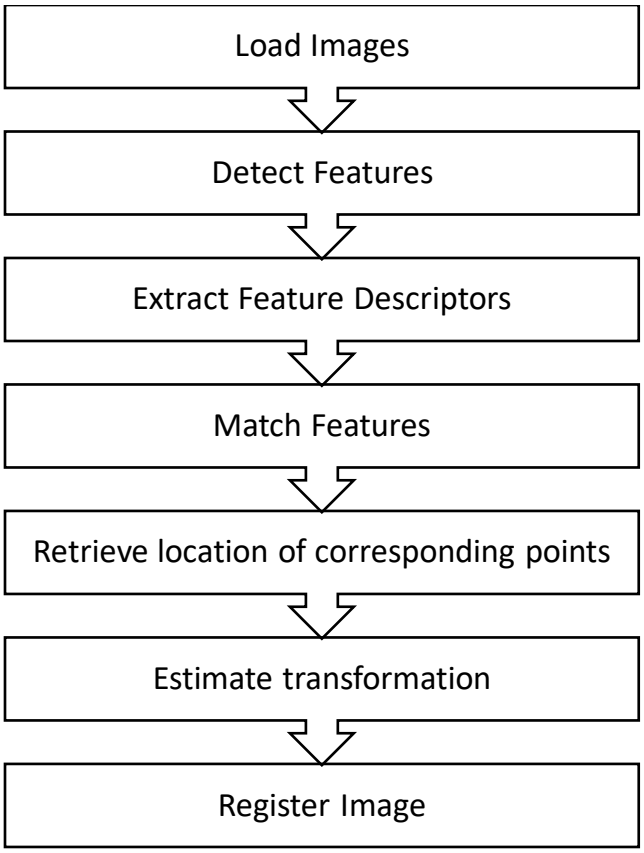

Figure 9. Automated Feature Detection and Matching Process

2.2.7. Average Method

The concept of Image Fusion could be considered as simply taking an average of the images available. This method is the most basic technique that highlights both positive and negative details of the input images. In this technique we simply take the average of intensities of the moving image and fixed image at each pixel to construct our fused image.

Fused Image = (Moving Image+Fixed Image)/2

As you can see, this is the basic image fusion algorithm which could be used as a sub step in other complex algorithms. This method does not help us in our research as we want the maximum information content from our input images. This algorithm has been used in our proposed method to formulate fusion rules.

2.2.8. DCT Based Laplacian Pyramid Fusion Technique

The DCT Based Laplacian Pyramid Fusion Technique [5][8] is a very similar technique to wavelet transformation which we discussed above in this section. To have a better understanding of Laplacian pyramid we need to learn about Gaussian Pyramid first. The construction of Gaussian pyramid is basically reducing an original image of size (2^m x 2^n) into a new blurred and less detailed image of size (2^{m-1} x 2^{n-1}) using a gaussian filter. This process is repeated until the size of the image becomes (1 x 1). During this process, at each level apart from the new blurred and less detailed image, we get another image which stores all the details of the initial image. All the detailed images received at each level forms the Laplacian image pyramid. In the below image you can see g_0 is our original image and g_1, g_2, and so on are our transformed Gaussian images at each level, l_0, l_1 are our Laplacian images.

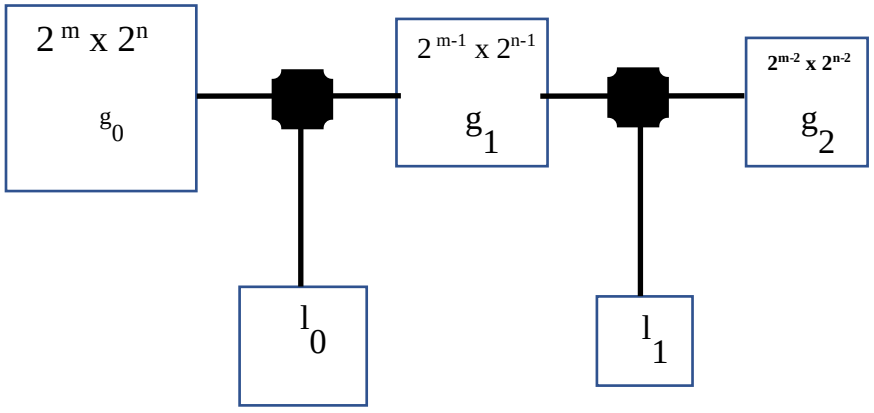

Figure 10. Construction of Gaussian and Laplacian Pyramid

We apply our fusion rules to Gaussian images obtained at each level and to these Laplacian images obtained from moving and fixed image. The modified Laplacian pyramid is used again to reconstruct our new transformed gaussian image into the final fuse image. The process of decreasing the size of image is called reduction and process of re-constructing the image is called expansion.

2.2.9. Fuzzy Logic Fusion

Fuzzy logic [8][14] is generally used to solve problems whose solution may lie between completely true and completely false. It is a black box that is used for mapping input space to an output space. For example, if we provide information of how service provided at a restaurant to the black box, it returns us how much tip should be given. This black box can contain fuzzy system, equation, neural networks etc. to map the input to output. MATLAB provides the ability of solving such problems with the help of Fuzzy Logic Toolbox and fuzzy inference system which can be used a black box. Fuzzy logic toolbox provides functions that can be used to design fuzzy inference systems. MATLAB also provides a Fuzzy Logic Designer App that allows you to develop and test fuzzy inference system. For image fusion process, we define the fusion rules in fuzzy inference system, which maps the input value according to the fusion rules and returns the output value. This output value is used to reconstruct the fused image.

Following are few functions provided by Fuzzy Toolbox that would be used later in Chapter 4 and Chapter 5.

Table 5. Fuzzy Logic Toolbox Methods

Functions	**Description**
readfis	Load fuzzy inference system from a file.
evalfis	Evaluates the fuzzy system and maps the input to output accordingly.

CHAPTER 3. IMAGE REGISTRATION AND FUSION APPLICATION

3.1. Design Overview

There are various types of application design paradigms that helps in understanding its functionality, data involved and logic of any project. They not only help us to envision the final project but also helps in identifying areas that might pose potential challenges in other stages of SDLC. In this section, we will mainly focus on how IRFA was designed, and try to understand the functionality of the application using Use Case Diagram and Activity Diagrams. Following is the basic use case diagram of the application that describes how exactly application fuses two colored images.

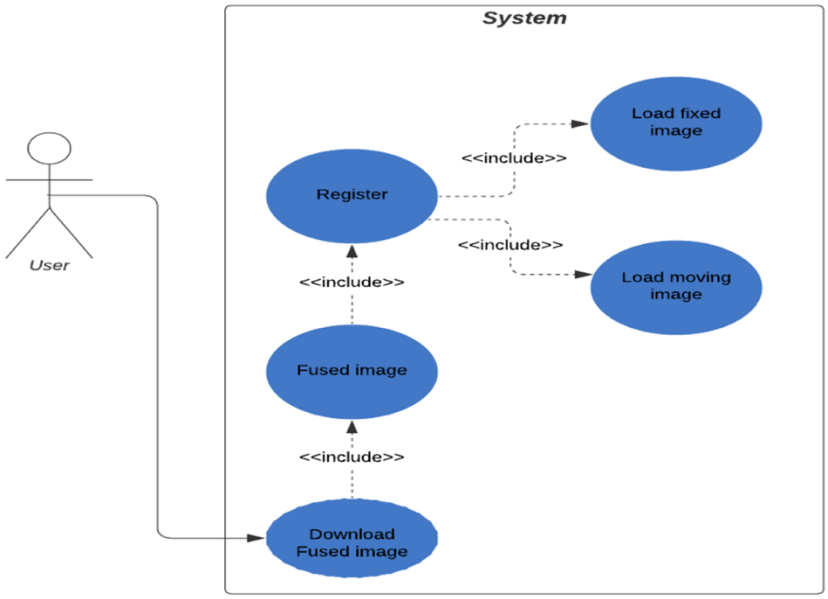

Figure 11. Use Case Diagram

The IRFA is mainly used to execute image registration and image fusion. Using this app, you can:

1. Obtain the registered image and fused image of a grayscale image
2. Obtain the registered image and fused image of a colored image

3. Obtain the fused images of a batch of grayscale/colored images

The IRFA app mainly consists of 3 tabs which will be discussed in detail in this section.

3.1.1. Single Processing – Grayscale

This tab is used to acquire the registered image and fused image of a single grayscale image. The user is expected to upload a moving image (grayscale) and a fixed image (grayscale). This tab is designed in such a way that if a color image is uploaded, it automatically converts it to gray scale.

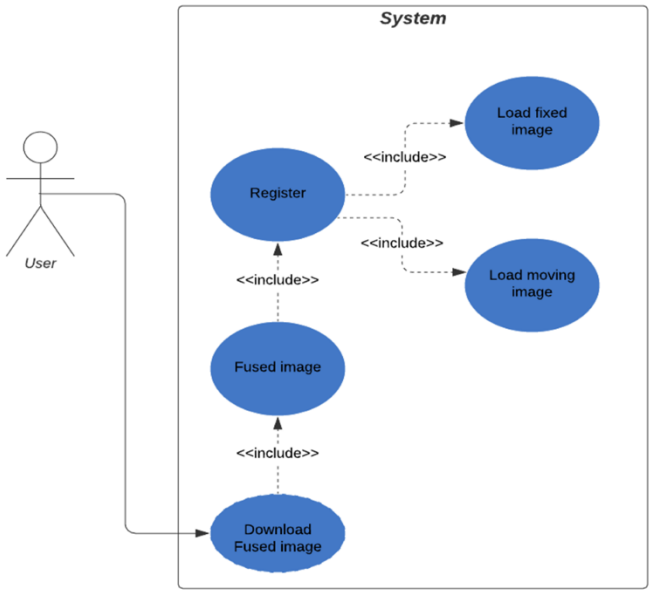

Figure 12. IRFA- Single Processing Grayscale Tab

First task is to do image registration. Once the user clicks "register image" button, the image will be registered. The imregister function uses an optimizer and metric created by imregconfig. The configurations for optimizer have default values set. If the user wants specific values for optimizer, they can be set in the Parameters component in the UI. The image registration usually takes up to 10 seconds for small-size images. The next step is image fusion. Once the user clicks "Fused Image" button, fused image will be generated. The program uses single

level discrete 2-D wavelet transform (dwt2) for image fusion. This image can be downloaded by clicking “download fused image” button. It will be downloaded to the local workspace.

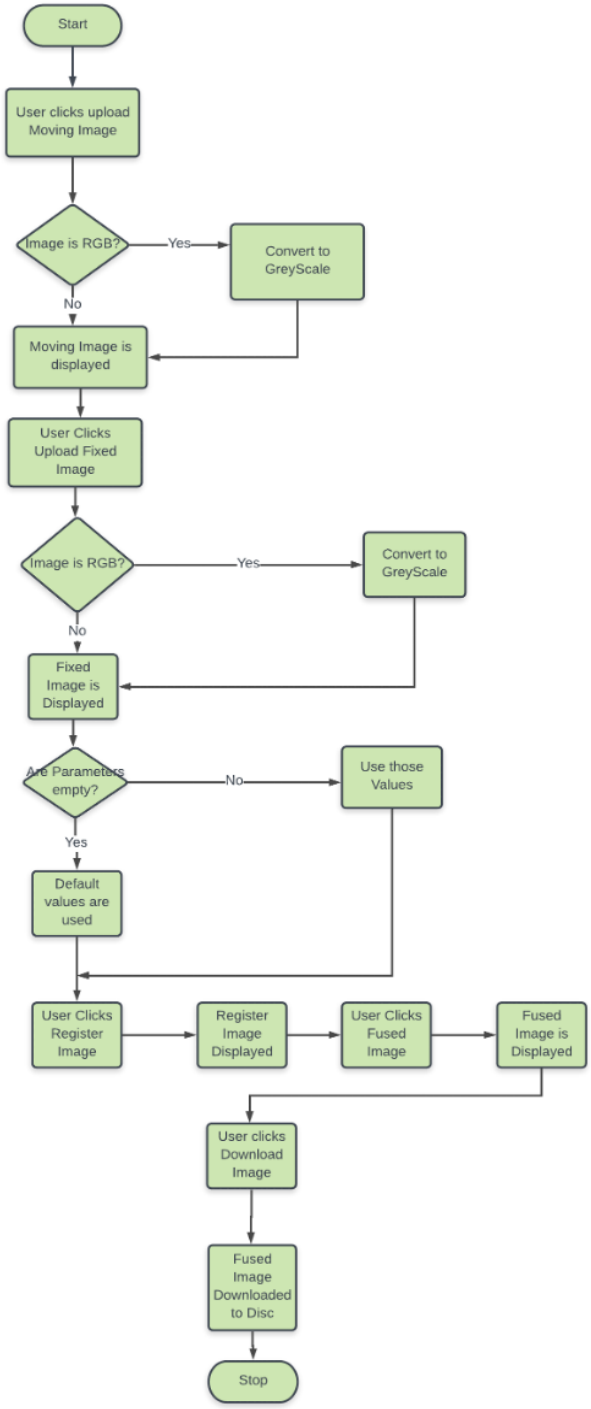

Figure 13. Activity Diagram - Single Processing Grayscale

3.1.2. Single Processing – Colored

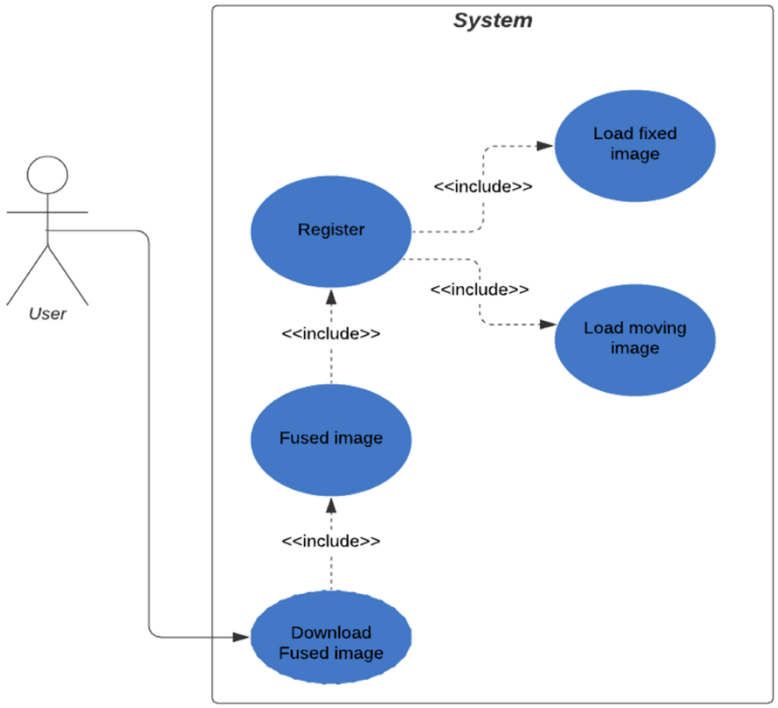

Figure 14. IRFA- Single Processing Colored Tab

This tab works very similar to the first one. It is used for getting the registered and fused image of a single-colored image. The user is expected to upload a moving image (colored) and a fixed image (colored). This tab is designed in such a way that if a gray scale image is uploaded, it gives an error. Image registration and image fusion are done after the user clicks “register image” button and “fused image” button respectively, which can be downloaded using “download fused image” button.

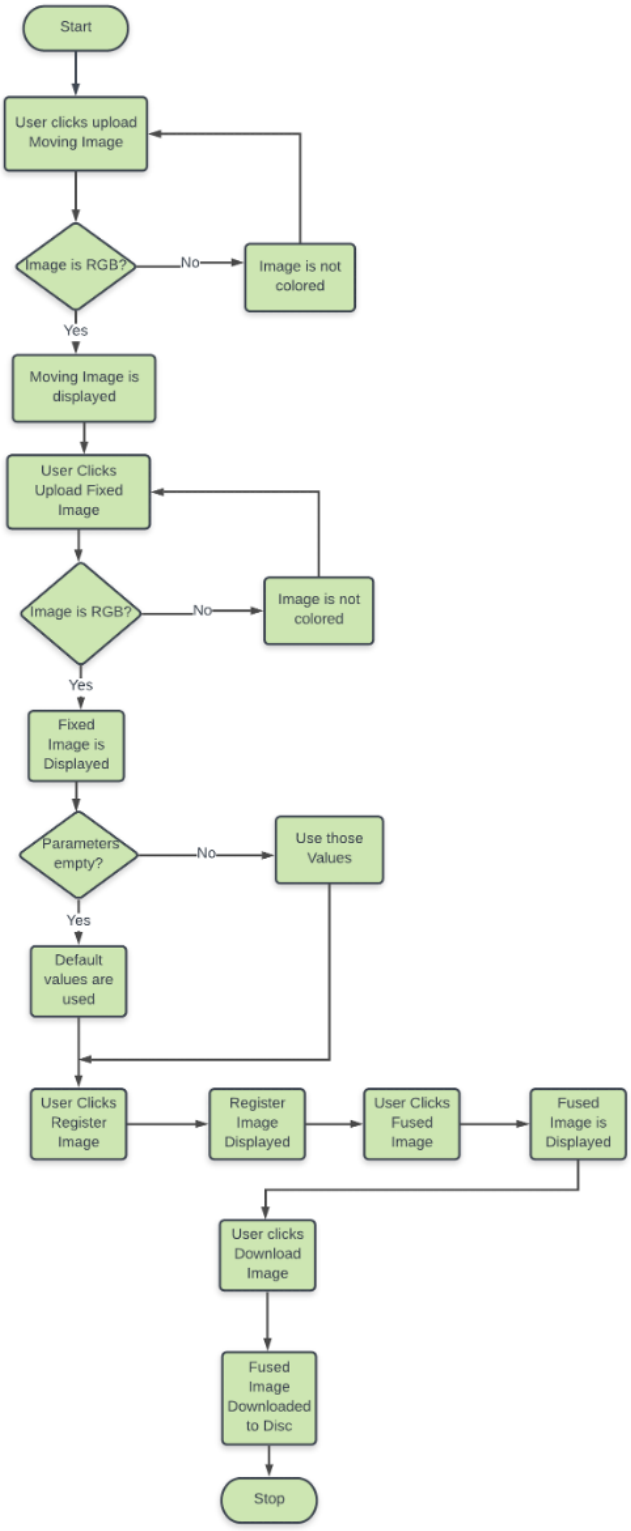

Figure 15. Activity Diagram - Single Processing Grayscale

3.1.3. Batch Processing

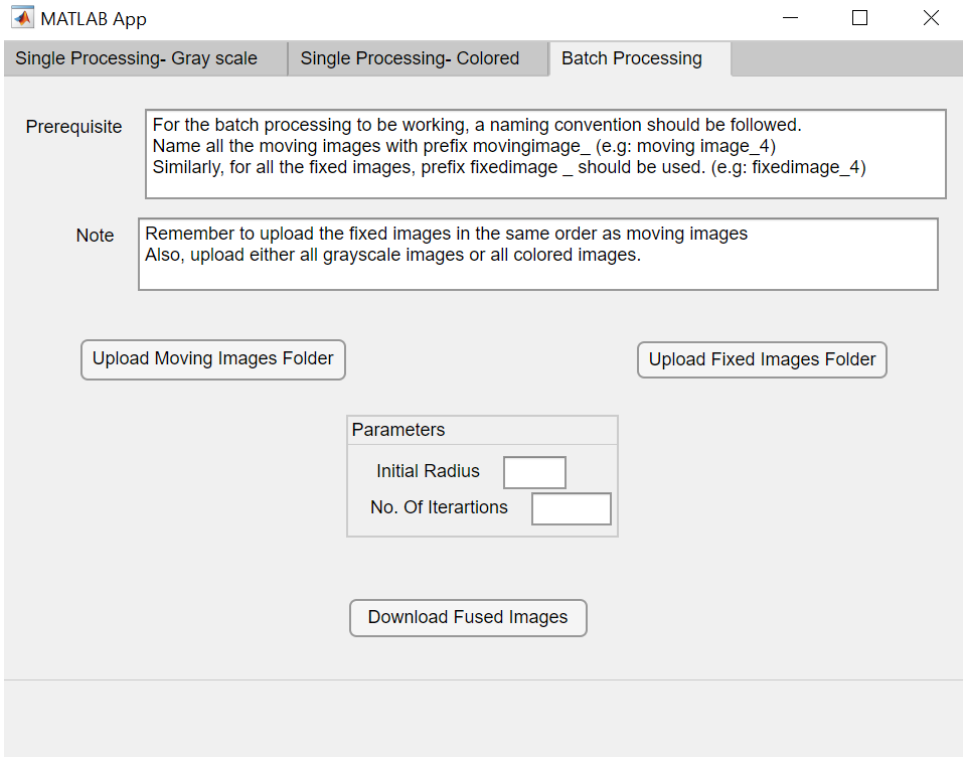

Figure 16. IRFA- Batch Processing Tab

This tab is used for image registration and image fusion of large set of grayscale/colored images. The user uses upload moving images folder and upload fixed images folder buttons to input the location of respective images folder for the application. For the batch processing to be working, a naming convention should be followed. Name all the moving images with prefix movingimage_ (e.g: moving image_4). Similarly, for all the fixed images, prefix fixedimage _ should be used. (e.g: fixedimage_4). The user is expected to upload the fixed images in the same order as moving images i.e., both the folders containing images are expected to be ordered and equal in number. Also, uploaded images should be either grayscale images only or colored images only.

The application has an upper threshold of 500 images i.e., a maximum of 500 images can be registered and fused with IRFA. The performance can vary depending on no. Of images. The

user can download all the fused images using the download fused images button. The user can use single processing tabs to determine the suitable parameters (initial radius and no. of parameters) by trial and error. Once these parameters are established, batch processing of a set of images can be performed.

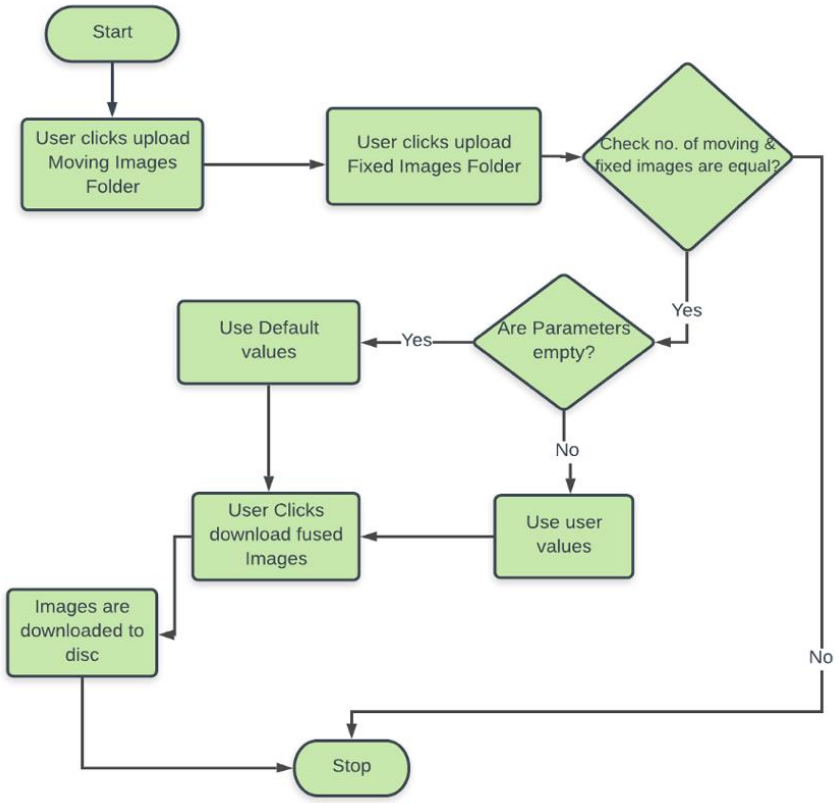

Figure 17. Activity Diagram – Batch Processing Grayscale

3.2. Logic

The logic of the application could be divided into various sub modules such as uploading images, performing image registration, performing image fusion, downloading images etc. All the three tabs of IRFA extends the same logic modules for uploading and downloading images. The uploading and downloading of images are pretty much handled by the in-built functions provided by the image processing toolbox, such as imread and imwrite which we have already discussed in

previous chapters. Performing Image Registration and Image Fusion on uploaded images can be identified as the main logic of IRFA.

3.2.1. Image Registration

IRFA uses automatic intensity-based image registration technique to register moving images with respect to fix images. This technique is mainly used for registering grayscale images, but we have extended this technique to perform image registration of colored images automatically.

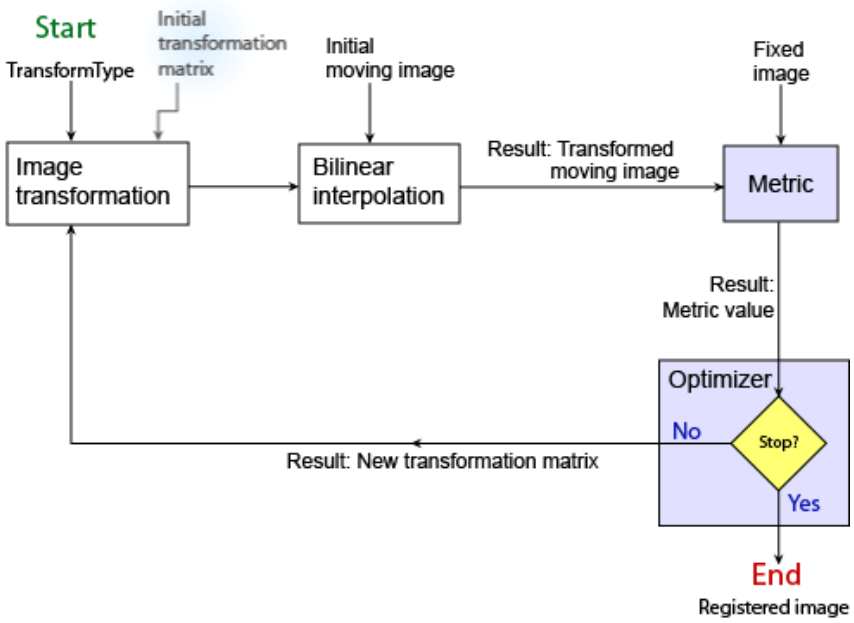

Figure 18. Automatic Image Registration.

This image has been taken from Mathworks, it clearly demonstrates the process involved in automatic intensity-based image registration. As you can see in the figure, it is an iterative process carried out until best results are achieved. Automatic Intensity-based registration has following requirements.

1. Input Images: Load the moving and fixed image into the workspace.
2. Transformation: Select a transformation type such as affine or projective, or similarity.

3. Transformation Matrix: This is an internally determined transformation metric provided in the starting of the process.
4. Optimizer: Optimizer is an object that defines methodology for maximizing and minimizing the metric. The methodology is determined by modifying its properties. The properties of Optimizer object and their default values are
    (a) Growth Factor: 1.050000e+00
    (b) Epsilon: 1.500000e-06
    (c) Initial Radius: 6.250000e-03
    (d) Maximum Iterations: 100

In this process, we upload our fixed and moving image, transform the moving image using the initial transformation metric and selected transformation type. Once the moving image has been transformed, a new similarity metric is created after comparison between transformed image and fixed image. Now optimizer looks for the end point of the process, i.e., if maximum number of iterations have been achieved then process terminates otherwise optimizer updates the transformation matrix using the new created similarity metric and its properties, the process continues until end point is reached.

This is the logic that we have been using in our Single Processing – grayscale tab of IRFA. Let's try to understand how we have extended this logic for colored images, which is being used in Single Processing – colored tab of IRFA.

As we have discussed earlier in section 1.4.2, an RGB image comprises of three 2-D planes R, G, and B, each representing their own pixel intensity. We will split both moving and fixed image into three channels R-Channel, G-Channel, and B-Channel respectively. Each channel is considered as a grayscale image. Now we would register R-Channel of moving image with respect

to R-Channel of fixed image and repeat this process for other two channels. After registering all three channels of moving image, we have three registered 2-D images which can be concatenated together to form a single RGB image, this image is our final registered image.

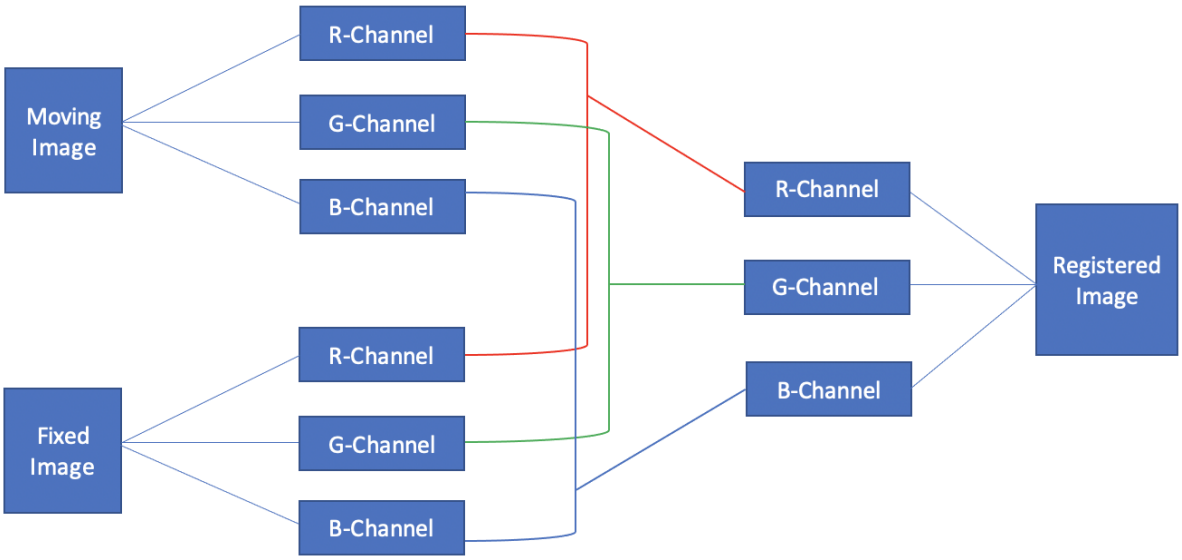

Figure 19. IRFA – Image Registration of Colored Image

3.2.2. Image Fusion

We have used a single – level discrete 2-D wavelet-based image fusion technique to fuse our fixed image with the registered image obtained in the previous method. In Chapter 2, we discussed about wavelet transformation. In the below flow chart, you can clearly see the registered image and the fixed image is again split into three channels as wavelet transformation can be performed only on 2-D images. Wavelet transformation is then applied to each of these individual channels to create small wavelets which is represented in form of a coefficient matrix in MATLAB. A wavelet can be represented as [cA, cH, cV, cD] where cA is an approximate coefficient matrix, and cH, cV and cD are detailed coefficient matrix for horizontal, vertical, and diagonal respectively. Once we have obtained our coefficient matrix then we apply basic averaging and

maximizing operations to these coefficient matrices which can also be considered as fusion rules to obtain a set of transformed coefficient matrix. Finally, we apply inverse wavelet decomposition on these transform matrices to obtain three channels which is then concatenated together to form our final fused image.

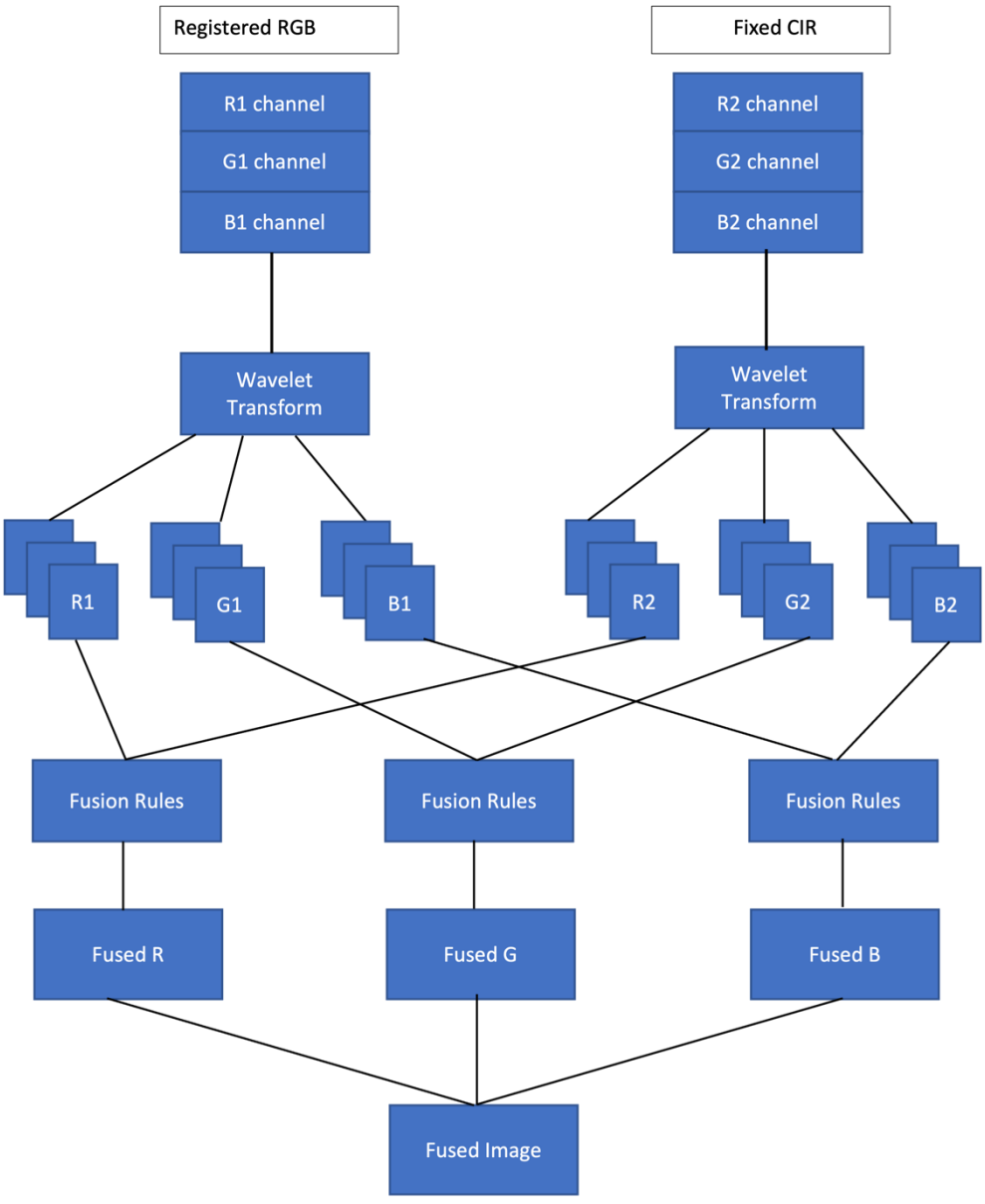

Figure 20. IRFA – Image Fusion of Colored Image

CHAPTER 4. IMPLEMENTATION OF EXISTING METHODS

4.1. Overview

As stated in the section 1.5, our two main challenges were to come up with an automated approach for image registration and fusion, and the second challenge was to be able to protect the color information, i.e., our proposed method should support colored images as well. This section demonstrates how existing techniques failed to fulfill our two main requirements, arising the need for our proposed method. In this section, we would be revisiting the existing techniques described in Chapter 2 for image registration and image fusion individually and try implementing those techniques with our input data sets.

4.2. Input Data Set

The input data set used in this research was collected by Department of Agriculture and Biosystem engineering, NDSU. Two sets of cameras were used to capture images of wheat plants which were grown in NDSU greenhouses. The first camera was used to capture colored image, and the second multi-band camera was used to capture Color-Infrared (CIR) images of the same wheat plants. Both colored and CIR Images are RGB in nature. Once both the images were collected, a pair data set which included a color image, and a corresponding CIR image of the same plant was created. These pair data set was cropped using an automatic webtool to obtain Region of Interest (ROI). All the images were then subjected to noise removal and image segmentation process. The wheat pants mainly had two kinds of disease – Tanspot and Rust. The complete image data set was then divided into four batches according to severity of the disease.

1. Light Tanspot Disease
2. Severe Tanspot Disease
3. Light Rust Disease

4. Severe Rust Disease

Each of these batches included approximately 250 pair data set, thus our total data set for this research was thousand color images and thousand CIR images. We need an image registration and fusion method that supports thousand colored and CIR image. The data set looks something like this. The RGB image is our moving image, and the high-resolution CIR image is considered our fixed image.

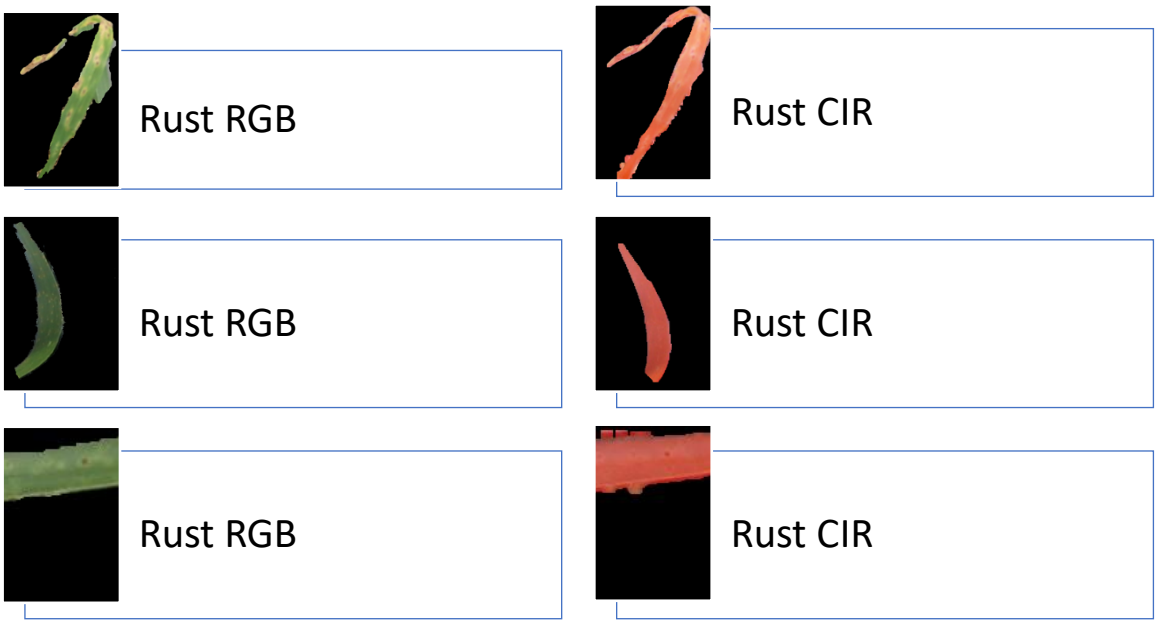

Figure 21. Input Pair Data Set

4.3. Image Registration Methods

4.3.1. Registration Estimator App

First, we load our colored moving image and fixed image into the Registration Estimator Application. As we can see in Figure ……, a warning dialogue box shows up saying that the RGB images are converted into grayscale. Registration Estimator App does not support colored images. Therefore, we cannot protect the color information of our input data set, thus failing our one of the

challenges. We can try to semi-automate this process of registration, but due to lack of protecting the color information, we still cannot use Registration Estimator App for our research.

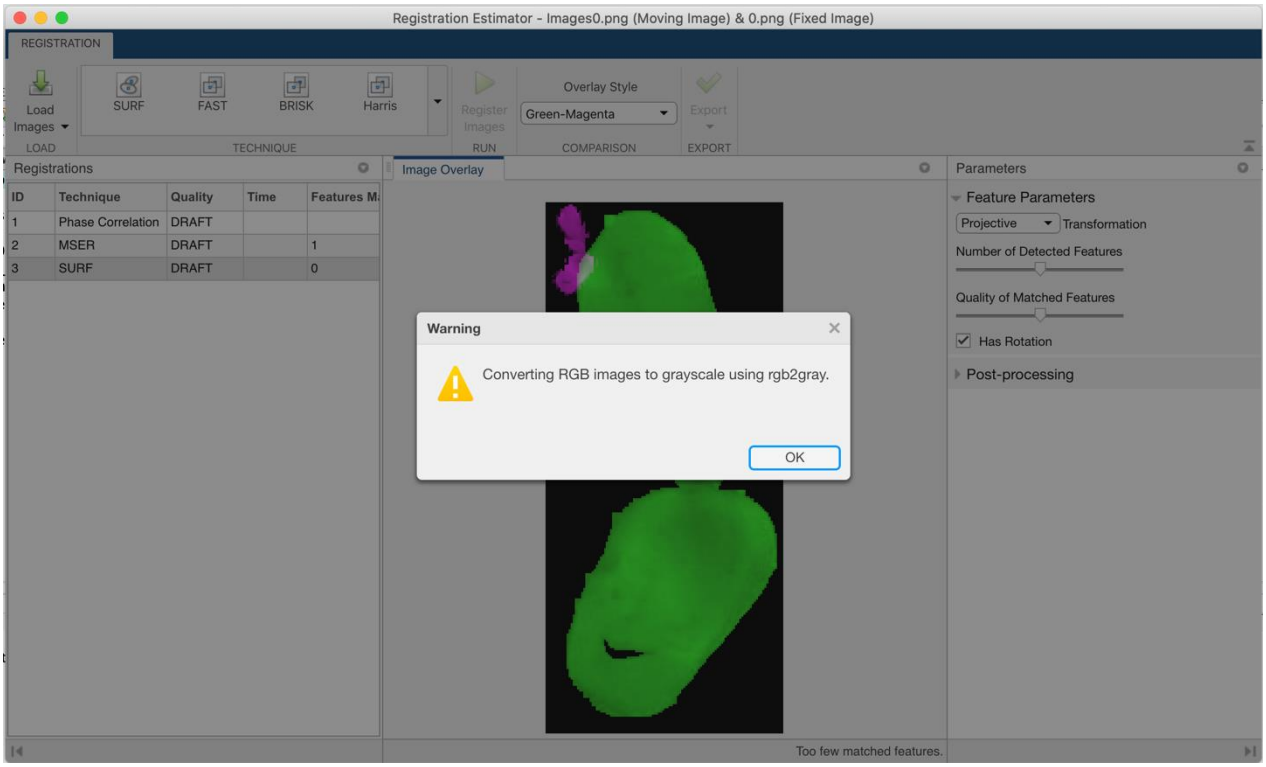

Figure 22. Implementation of Registration Estimator App with our input data

4.3.2. Control Point Registration

In Control Point Registration, we read moving image and fixed image into the workspace. For reading the images into workspace we use imread function provided by Image Processing Toolbox. We then resize the moving image to the size of fixed image. We have use subplot function to compare the moving, fixed and registered image.

```
%read images
sourcepic = imread('000002.png');
sourcepic_resized = imresize(sourcepic, [284,126]);
subplot(221);
imshow(sourcepic_resized);
title('FIXED IMAGE');

unregistered = imread('000002_marked.png');
unregistered_resized = imresize(unregistered, [284,126]);
subplot(222);
imshow(unregistered_resized);
title('MOVING IMAGE');
```

Code Snippet 1. Implementation of Control Point Registration

We have use subplot and imshow function to divide our display grid into four subplots, the first subplot displays the fixed image, the second one displays the moving image that needs to be registered and third subplot displays the registered image which would be displayed once the image was registered. The sub-plotting images helps to compare each of these images.

The next step would be to select the control points in moving and fixed image using the control point selector window and store the location of selected points in each of the images into moving points and fixed points arrays. In figure 23, we can see how we have selected control points in our input moving and fixed images.

```
%select control point pairs
[moving_points, fixed_points]=cpselect(unregistered_resized, sourcepic_resized,'wait', true);
```

Code Snippet 2. Control Points

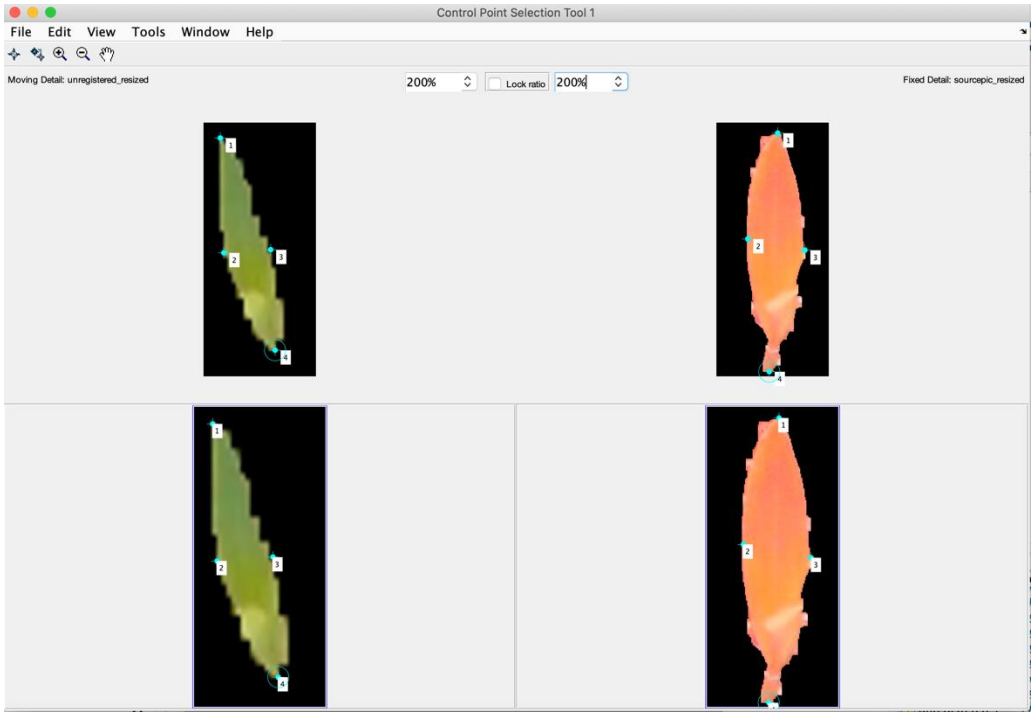

Figure 23. Implementation of Control Point Registration with Our Input Data

Finally, we create a tform affine object from the selected control points and affine transformation. This tform object is used to transform our moving image into the final registered image, which is then displayed in the third subplot of display grid. We then use the imwrite function to save our registered image.

```
%infer geometric transformation
tform = fitgeotrans(moving_points, fixed_points, 'affine');
%Rfixed = imref2d(size(sourcepic_resized));
registered = imwarp(unregistered_resized, tform);
registered_resized = imresize(registered, [284,126]);
subplot(223);
imshow(registered_resized);
title('FINAL REGISTERED IMAGE');
%subplot(224);
%imshowpair(sourcepic_resized,registered_resized,'blend')
imwrite(registered, 'FinalRegistered.png');
```

Code Snippet 3. Implementation of Control Point Registration

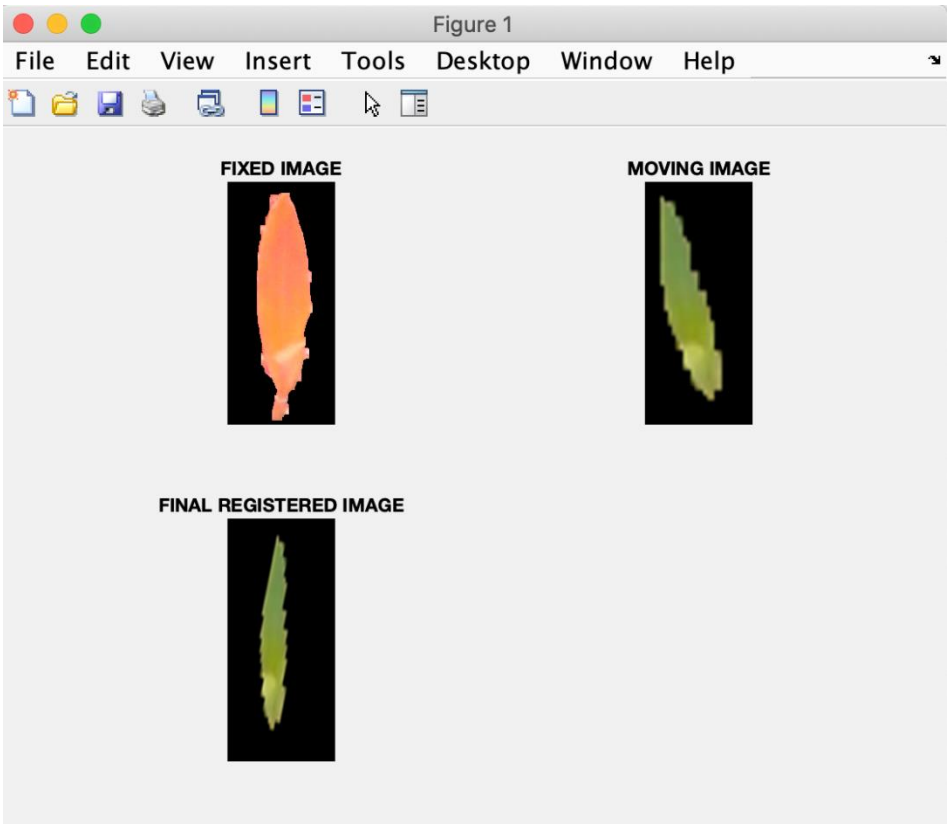

Figure 24. Control Point Registration Result with Our Input Data

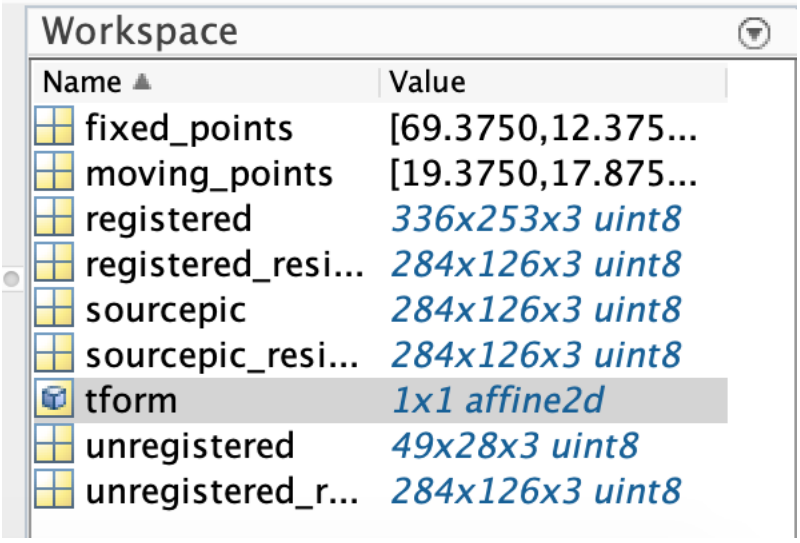

Figure 25. Workspace Data

Figure 25 shows a snapshot of all our data loaded in the workspace. The size of all the image loaded into workspace, fixed points, moving points and tform object.

The control point registration was successful for colored images, but this whole process could not be automated as for each image, we would have to manually select control points to form a geometric transform object. Therefore, this requires more human labor and time to register our large input data set, thus not suitable for our research.

4.4. Image Fusion Methods

4.4.1. DCT Based Laplacian Pyramid Fusion

In this section we will cover implementation of DCT Based Laplacian Pyramid Fusion. This technique includes reduction and expansion of images using dct2 function and idct2. The function dct2 returns the two-dimension discrete cosine transform coefficient matrix of an image.

One of the key points to remember in this fusion technique is that our given moving image and fixed image should be of size 2^m x 2^n, which does not work with our input data set, as our input data set might be or might not be of that size format and reducing it to that size format would not give best results. We have still implemented this technique, to see if any work around could be achieved.

```
for i=1:k
    IM = reduce2d(IM1);
    Id1 = IM1 - expand2d(IM);
    IM1 = IM;
    IM = reduce2d(IM2);
    Id2 = IM2 - expand2d(IM);
    IM2 = IM;
    dl = abs(Id1)-abs(Id2)>=0;
    Idf{i} = dl.*Id1+(~dl).*Id2;
end
imf = 0.5*(IM1+IM2);

for i=k:-1:1
    imf = Idf{i} + expand2d(imf);
end
```

```
%=========================================
function[Ie] = expand2d(I)
mn = size(I)*2;
disp(mn);
IDCT = dct2(I);
Ie = round(idct2(IDCT,[mn(1) mn(2)]));

%=========================================
function[Id] = reduce2d(I)
mn = size(I)/2;
disp(mn);
IDCT = dct2(I);
Id = round(idct2(IDCT(1:mn(1),1:mn(2))));
```

Code Snippet 4. Implementation of Laplacian Transform Fusion

In the above code snippet, you can see we are performing reduction on each image for a fixed number of levels ‘k’, which in our case k = 2. IM1 and IM2 are our initial images that needs to be fused, which are reduced to gaussian images, and the difference between the original and gaussian image is stored in laplacian image. A maximum of the two laplacian image is selected at each level, which is used to reconstruct our new transformed image imf (which is simply an average of the two original images) to achieve a fused image.

The reduce and expand function apply cosine transform and inverse cosine transform to the images using dct2 and idct2 functions. The result is displayed in the next figure, the first two images are our registered image and fixed image, and the third image is our fused image. This technique could be extended for color images, but it would not have given us best results due to size indifference format.

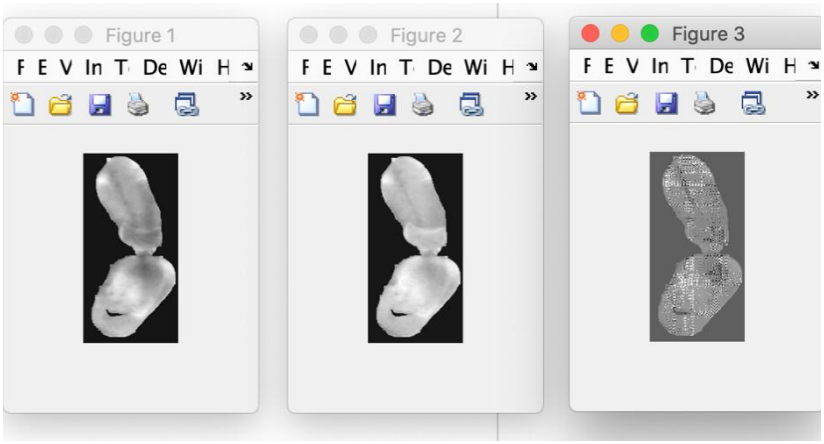

Figure 26. Laplacian Transform Fusion Result with Our Input Data

4.4.2. Fuzzy Logic Image Fusion

As we discussed in Chapter 2, fuzzy logic image fusion is a technique to map an input image to an output image using a fuzzy inference system. Different fusion rules can be used to fuse image according to the requirement of the problem statement.

```
im=evalfis([M1(:) M2(:)],f);
% Converting the column into matrix
k=l;
for i=l:1:zl
for j=1:1:s1
img(j,i)=im(k);
k=k+1;
end;
end;
```

Figure 27. Fuzzy Logic Fusion Rules

For example, in fruit detection process [8], the membership functions in fuzzy inference systems have been used to interpret information from visible and thermal images. If the thermal is warm and visible is red, then fused image is a fruit, if thermal is average, the fused in non-fruit, and if visible is not red then, fused is not a fruit.

For our research, we try to identify the common area between the two images. M1(:) simply converts the 2-D matrix of image M1 into a single column matrix. We use pixel intensities from these two-column matrix to map the input fuzzy inference system that contains both of our images into output column matrix. Finally, we reconstruct image from the fused column matrix.

This approach does not work for us, as fuzzy logic cannot be extended to color images, as this technique does not give accurate solution to our input data set. We can use this technique to find whether wheat has any disease but cannot be used for accurate results.

CHAPTER 5. IRFA DEVELOPMENT

5.1. Development Overview

IRFA is a standalone application that has been developed on MATLAB using App Designer. Like other standalone applications, IRFA doesn't require any internet connections or services installed, but it does require MATLAB and its add-on toolboxes such as Image Processing Toolbox, Computer Vision Toolbox, etc. IRFA design has been explained in Chapter 3. In this Chapter, we would be focusing on how IRFA was developed, this could be broken down mainly into 4 main functions uploading images, downloading images, registering images, and fusing images. These main functions have been modified for each of the tabs of our application.

5.2. Logic

IRFA Development could be broken down mainly into four logic modules - uploading images, downloading images, registering images, and fusing images. These main logic modules have been modified for each of the tabs of our application. For example, the logic for uploading image has been used for uploading moving image and uploading fixed image which is then extended for uploading grayscale and colored images.

5.2.1. Uploading Images

According to our use case diagram, the first ability of the application is to upload a fixed image and a moving image. We have been using same Upload Image Module for uploading both the images. This method has been extended to first two tabs of the application, Single Processing – Grayscale tab and Single Processing – Color tab.

```
global b;
[filename, pathname] = uigetfile('*.*', 'Pick a moving Image');
filename=strcat(pathname,filename);
b=imread(filename);
imshow(b,'Parent',app.UIAxes);
```

Code Snippet 5. IRFA – Uploading Image

The above code snippet is a callback function attached to upload moving image button. Similar callback function is developed for upload fixed image button. We have used uigetfile method, which opens a file selection dialogue box and allows the user to select the image file. This method returns the filename and its directory, which is then used to load the image into workspace of the application, and finally the image is displayed on their respective axes. The images are stored in variable which is declared globally as it will be used in other callback functions as well.

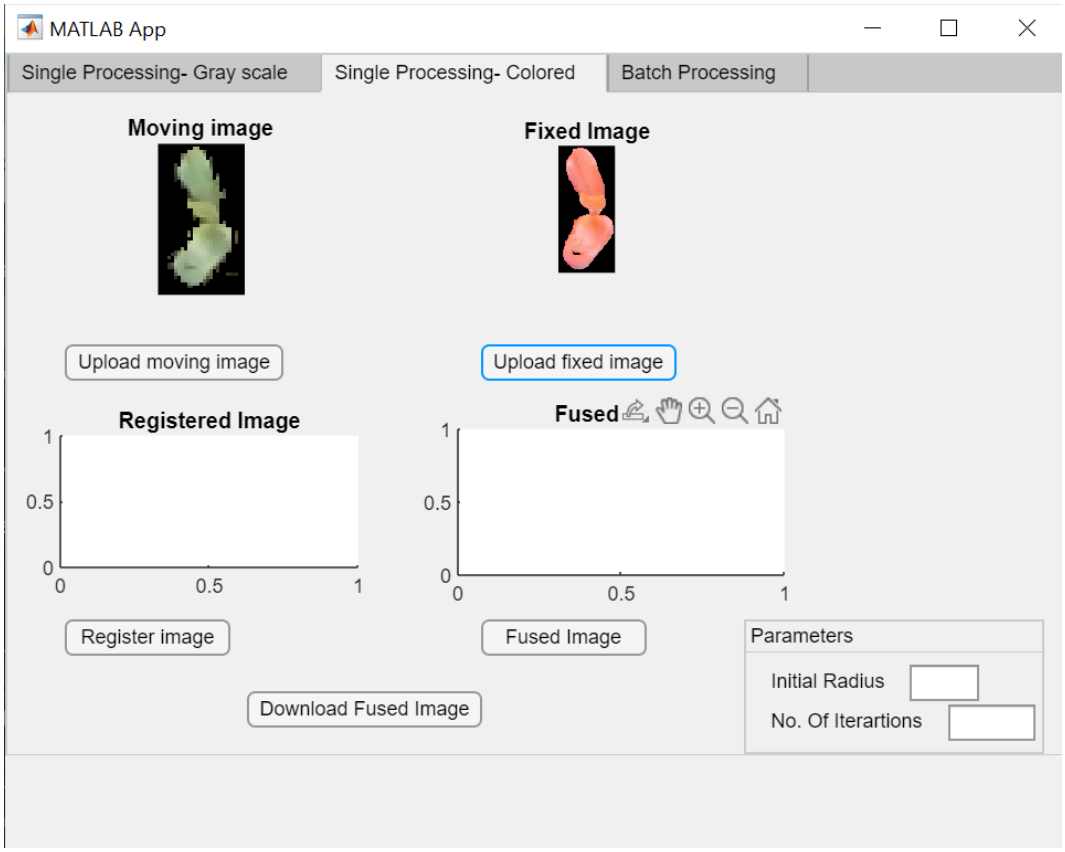

Figure 28. IRFA- Upload Image Result

5.2.2. Registering Images

Our application allows image registration of both grayscale and color images in different tabs. As we discussed in Chapter 3, we have used automatic intensity-based registration for registering our input image data set.

```
            global a;
            sourcepic_resized = a;
             [rowsA, colsA, numberOfChannelsA]=size(sourcepic_resized);
             [R1,G1,B1] = imsplit(sourcepic_resized);

             global b;
             unregistered = b;
             [rowsB, colsB, numberOfChannelsB]=size(unregistered);
             if rowsB~= rowsA||colsA~=colsB
        unregistered_resized = imresize(unregistered,[rowsA colsA]);
             end
             [R2,G2,B2] = imsplit(unregistered_resized);
             [optimizer, metric] = imregconfig('multimodal');
    optimizer.InitialRadius = optimizer.InitialRadius/3;
    optimizer.MaximumIterations = 500;
    global RGB;
    R = imregister(R2,R1,'affine',optimizer,metric);
    G = imregister(G2,R1,'affine',optimizer,metric);
    B = imregister(B2,R1,'affine',optimizer,metric);
    RGB = cat(3,R,G,B);
     imshow(RGB,'Parent',app.UIAxes5);
```

Code Snippet 6. IRFA –Register Image

In the above code snippet, ‘a’ and ‘b’ are our moving and fixed images. We have used imresize function to resize our moving image to the size of fixed image. We have then split both moving and fixed images into R, G and B Channels using imsplit function. An initial transformation matrix is created using imreconfig function which is then used to register the each of the channels of moving image with respective fixed image channels. The optimizer keeps updating transformation matrix using a similarity metric created by similarities of transformed image and fixed image at each iteration, until maximum iterations have reached. These maximum iterations are set default to 500 if no external parameters are provided. Now we have three registered channels R, G and B which is concatenated together to form a registered image. This

registered image would be used in the process of image fusion, therefore is decarded globally as well. The registered image is then displayed on its respective axes using imshow function. The same process is done for all the images in the selected folder in our Batch Processing tab.

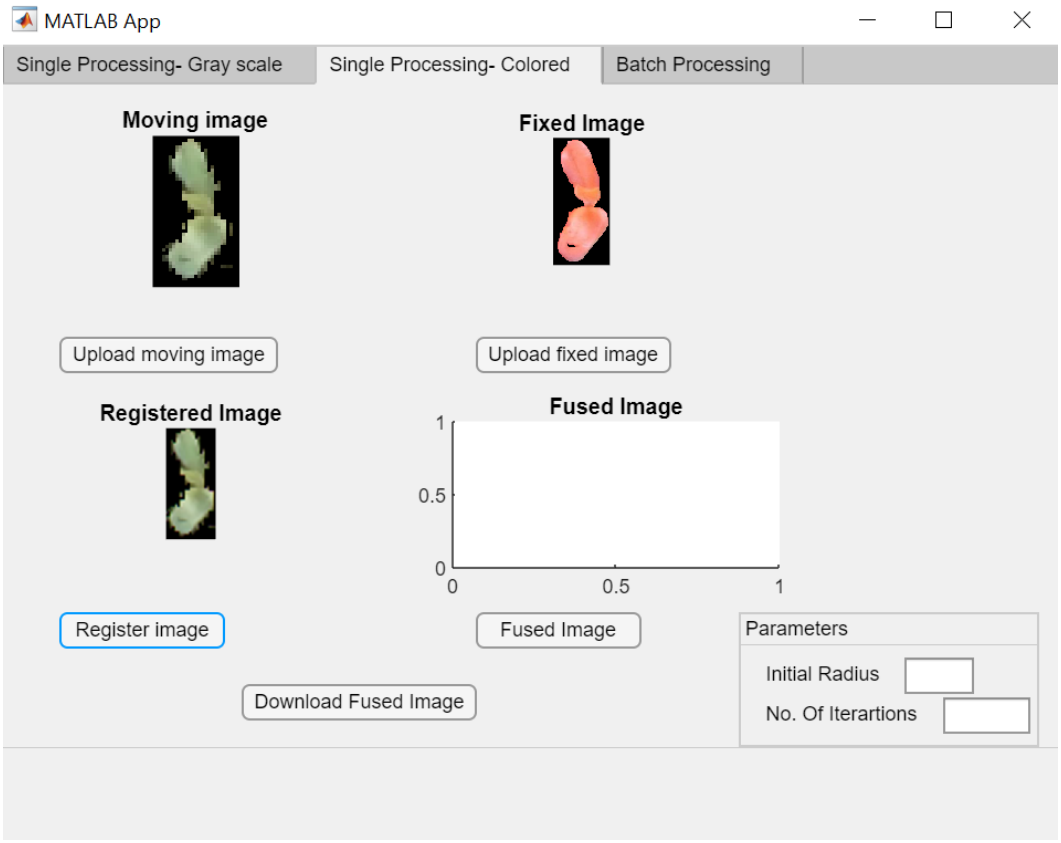

Figure 29. IRFA- Register Image Result

5.2.3. Fusing Images

We have been using single – level discrete 2-D wavelet-based Fusion technique for fusing images in our application. This technique has been explained in Chapter 3, we would be looking at the implementation of this technique.

The first step is again to split the registered image and fixed image into three channels and apply wavelet transformation to each of the channels. The images could be split into channels either by using imsplit function or by dividing the matrices along each channel, (:,:,1) is simply used to obtain the grayscale image in each 2-D plane. We have use dwt2 function which computes

single – level 2-D wavelet decomposition of the grayscale images and returns coefficient matrices for each axis.

In Code Snippet 7, [Ra, Rb1, Rc1, Rd1] can be explained as

1. Ra1: Approximation coefficient matrix of red Channel of fixed image.
2. Ra2: Coefficient Matrix in Horizontal axes for red Channel of fixed image.
3. Ra3: Coefficient Matrix in Vertical axes for red Channel of fixed image.
4. Ra4: Coefficient Matrix in Diagonal axes for red Channel of fixed image.

```
            global a;
            image1 = a;
            global RGB;
            image2 = RGB;
            R1 = image1(:,:,1); % Red channel
            G1 = image1(:,:,2); % Green channel
            B1 = image1(:,:,3); % Blue channel

            R2 = image2(:,:,1); % Red channel
            G2 = image2(:,:,2); % Green channel
            B2 = image2(:,:,3); % Blue channel

            [Ra1,Rb1,Rc1,Rd1]=dwt2(R1,'db2');
            [Ga1,Gb1,Gc1,Gd1]=dwt2(G1,'db2');
            [Ba1,Bb1,Bc1,Bd1]=dwt2(B1,'db2');
            [Ra2,Rb2,Rc2,Rd2]=dwt2(R2,'db2');
            [Ga2,Gb2,Gc2,Gd2]=dwt2(G2,'db2');
            [Ba2,Bb2,Bc2,Bd2]=dwt2(B2,'db2');

            [k1, k2] = size(Ra1);

             for i=1:k1
        for j=1:k2
            Ra3(i,j)=(Ra1(i,j)+Ra2(i,j))/2;
         end
end
```

Code Snippet 7. IRFA –Fuse Image Logic

The next step is to determine our fusion rules. We want to protect our color information of the images; therefore, we have selected the maximum coefficient at each pixel in all three axes

from respective channels of both the image, and for approximation coefficient matrix we have simply selected the average of the matrices of each channel

```
for i=1:k1
     for j=1:k2
        Gb3(i,j)=max(Gb1(i,j),Gb2(i,j));
        Gc3(i,j)=max(Gc1(i,j),Gc2(i,j));
        Gd3(i,j)=max(Gd1(i,j),Gd2(i,j));
     end
end
for i=1:k1
    for j=1:k2
        Ba3(i,j)=(Ba1(i,j)+Ba2(i,j))/2;
    end
end
for i=1:k1
    for j=1:k2
        Bb3(i,j)=max(Bb1(i,j),Bb2(i,j));
        Bc3(i,j)=max(Bc1(i,j),Bc2(i,j));
        Bd3(i,j)=max(Bd1(i,j),Bd2(i,j));
    end
end

CR=idwt2(Ra3,Rb3,Rc3,Rd3,'db2');
CG=idwt2(Ga3,Gb3,Gc3,Gd3,'db2');
CB=idwt2(Ba3,Bb3,Bc3,Bd3,'db2');

global FI;
FI = cat(3,CR,CG,CB);
FI=uint8(FI);
imshow(FI,'Parent',app.UIAxes6);
```

Code Snippet 8. IRFA –Fusion Rules

You can clearly see in the above code snippet, that we have applied our fusion rule for each respective channel of fixed image and registered image over the size of the image, i.e. for each pixel of the two images. Once we have obtained a transformed coefficient matrix for each of the channels. We have applied inverse discrete wavelet transformation to obtain our three channels: CR, CG, CB, which have been concatenated together to form the final fused image which gets displayed in its respective axes.

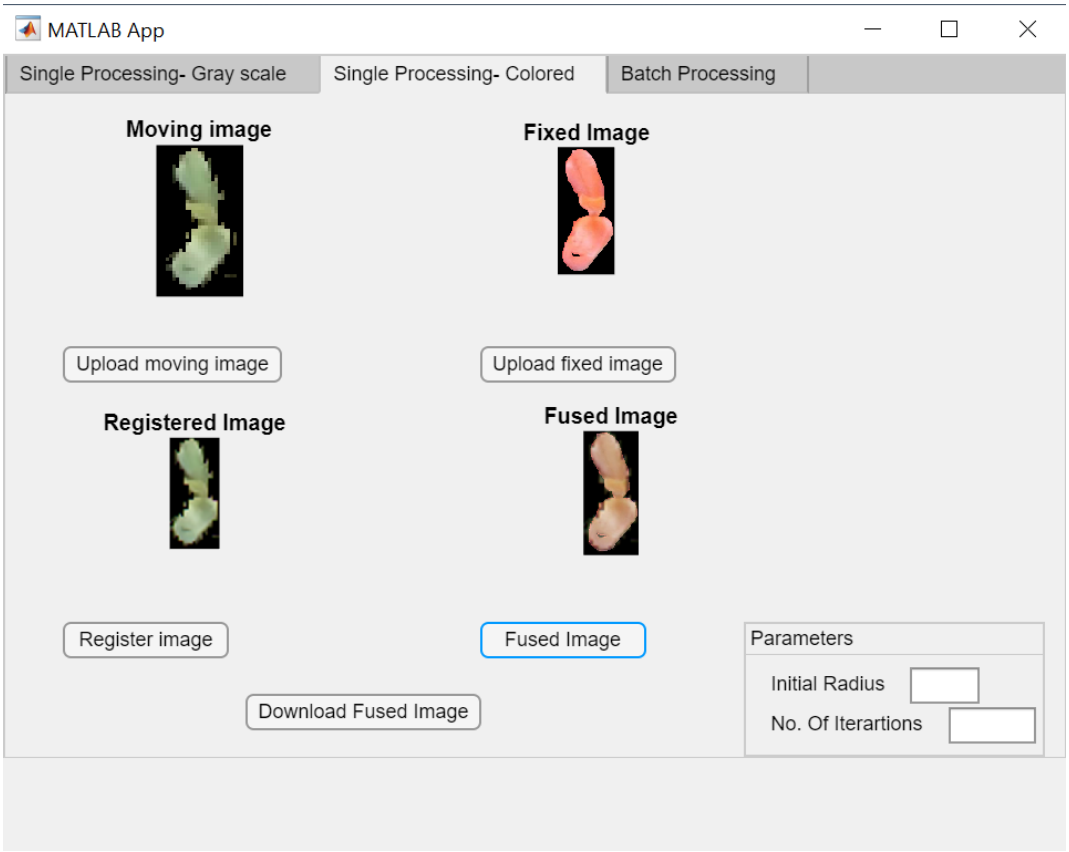

Figure 30. IRFA- Fuse Image Result

5.2.4. Downloading Images

The downloading image module is being used in our first two tabs. The logic behind this tab is to simply download the resultant fused image to the user's system. We have used imwrite function for this purpose, it downloads the image into the MATLAB's folder with user-provided filename.

```
        % Button pushed function: DownloadFusedImageButton,
        % DownloadFusedImageButton_3
        function DownloadFusedImageButtonPushed(app, event)
            global FI;
            imwrite(FI,'FusedImage.png');
        end
    end
```

Code Snippet 9. IRFA –Download Image Logic

In our Batch Processing Tab, the functionality provided by Register Image Button, Fuse Image Button and Download Image button has been combined into a single Fuse Image Button. Batch Processing Tab has been developed with a mind set to perform image fusion for a large data set, displaying all the four sets of images on the application is not viable in this case, therefore Fuse Image button directly saves the resultant fused image to the user's system.

CHAPTER 6. EVALUATION

6.1. Results

In Chapter 4 and Chapter 5, we implemented various existing and proposed image registration and fusion techniques using our input data set. Here, we compared the results obtained from each of the existing methods and the proposed method.

Table 6. Techniques Comparison

Methods	**Nature of Images**	**Automatic**	**Comments**
Control Point Registration	Supports both grayscale and colored images	No	Manually select control points in each input images
Registration Estimator Application	Supports only grayscale	Can be automated	It can be automated but does not guarantee good results for all the images of the input set. Whereas manual registration allows to choose from various available techniques thus giving better results.
Automatic Intensity Based Registration (Proposed)	Supports both grayscale and colored images.	Yes	Proposed method is automatic, thus can be used for a large data set.
DCT Based Laplacian Pyramid Fusion	Supports grayscale images, but can be extended to colored images as well	Yes	This can be only applied to images having a size of 2^m*2^n. Resizing our input images does not yield best results.
Fuzzy Logic Fusion	Supports grayscale. Also supports colored images to some extent	Yes	This method cannot be used for our input data set.
Single level Discrete Wavelength Transform Fusion (Proposed)	Supports both grayscale and colored images.	Yes	Proposed method is automatic, thus can be used for a large data set.

One of the most important stage of a SDLC, is the testing stage, where not only each building blocks of the application is tested but the whole integrated application is also tested to see if the complete application is working cohesively as a single unit. In this section we would be testing the performance of our application – whether our application was able to resolve the problems posed by existing methods.

6.2. Comparison between DCT Based Laplacian Pyramid Fusion and Single Level Discrete Wavelength Transform Fusion

Both the techniques, Discrete Cosine Transform Based Laplacian Pyramid technique and the single level Discrete Wavelength Transform Fusion could be extended to work for color images as well. The DCT Based Laplacian Pyramid technique proved to be not suitable for our input data set due the size restriction of the images. This technique involves sub sampling of the images at each stage, hence at each stage even if we approximate the size of our fixed image and registered image, we will incur a significant loss of information by the end of the process. Whereas, in single level Discrete Wavelength Transform Fusion technique, we have sub sampled the image only once, hence the information loss is negligeable.

Let's take a quick gander at the fused images obtained from both the techniques and try to understand how exactly information was lost. The following are the fixed image and the registered image used as input images in both the techniques.

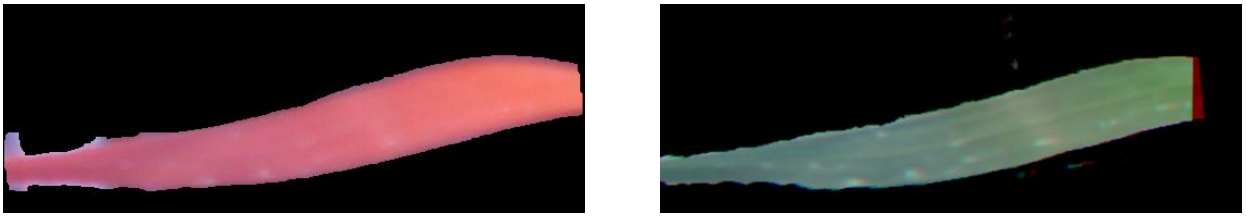

Figure 31. Input Images for Image Fusion

Figure 32. Comparison of Fused Images

In Figure 32, the left image is a result of DCT Based Laplacian Pyramid Fusion, and the right image is the result of Single level Discrete Wavelength Transform Fusion. As you can see, in the first technique we forcefully had to resize our image to the size requirement of the technique, which means we had to add extra pixels to our input image, as a result the intensity information stored at each pixel is not accurate anymore. Whereas in the proposed technique, a maximum of a single pixel would be added to the fused image, which does not make a significant difference to the intensity information between the input image and fused image. The Laplacian Pyramid fusion techniques works better than the Wavelet Fusion Technique for the images that are strictly following the size requirements and have similar hue and saturation between the fixed and registered images, as Laplacian technique involves converting RGB space into PCA (Principal Component Analysis) [6][10]. Generally, the RGB space has correlation between their R channel, G channel and B channel, which sometimes does not give desirable fusion result. Whereas in PCA space, if the hue and saturation is similar, then the RGB image is transformed into luminance and chrominance components [6]. This is highly suitable for the images taken by same camera, thus keeping similar hue and saturation. As our input data set was captured by two different cameras, we cannot convert RGB space into PCA space.

6.2.1. Fusion Performance

Let us try to compare Laplacian Pyramid Fusion Technique using PCA space and Wavelet Transform Fusion Technique using RGB space with few performance parameters [10]. Image

Processing Toolbox provides a series of functions like mse(), psnr() and std2() to help calculate following performance parameters.

1. Root mean square Error: In layman terms, RMSE simply is a measure for accounting the intensity change at each pixel between the referenced image and resultant fused image during the entire image fusion process. The lesser the RMSE value, the better the fusion method. Ideally, RMSE will be zero if the referenced image and the fused images are very alike. It is basically taking the average difference of values at each pixel in each of the images.

$$RMSE = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(I_r(i,j) - I_f(i,j)\right)^2}$$

2. Peak Signal to Noise Ratio: As the name suggest, it is measure of defining how much noise was added to the final image during the process of image fusion. It is the ratio of maximum possible intensity of an image and the maximum possible noise encountered. The higher the PSNR, the better the image fusion process.

$$PSNR = 20\log_{10}\left(\frac{L^2}{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(I_r(i,j) - I_f(i,j)\right)^2}\right)$$

where L in the number of gray levels in the image

3. Standard deviation: Standard deviation for an image measures the contrast of the image. The higher the standard deviation, the more scattered the intensity across the image, i.e., higher the contrast. Whether SD should be higher or lower depends on the requirement of the problem. In our scenario, since we are fusing RGB and CIR image, the higher the SD, better the color information of the image.

Let’s take a quick look at the results of each of these fusion performance parameters for both the techniques. The below results are the average of fifty image pair data set fused using both the techniques. As you can see for Wavelet Transform fusion technique, performance parameter has a better result as compared to the Laplacian pyramid technique. The proposed method has higher RMSE and PSNR values. The Standard deviation almost remains same for both the methods; hence both the methods equally good at protecting the color information.

Table 7. Fusion Performance Comparison

Parameters	**Laplacian Pyramid (PCA space)**	**Wavelet Transform (RGB space)**
RMSE	11.6142	10.5981
PSNR	19.3893	20.7788
SD	58.8153	58.9453

6.3. Fusing Time

In this section we would be testing the performance of application based on the time it takes to finally fuse the images. The Fusing time would be different for grayscale and color images. The color images are first divided into three channels and then each channel of the two images is registered. The three channels of the registered image are again fused with the three channels of our fixed image, and finally concatenated together to form a final fused image. This complete process takes more time almost double as compared to registering and fusing a grayscale image, as it does not involve splitting the images.

In the graph below you can see the comparison between time taken for fusing colored images and fusing the same images when converted to grayscale.

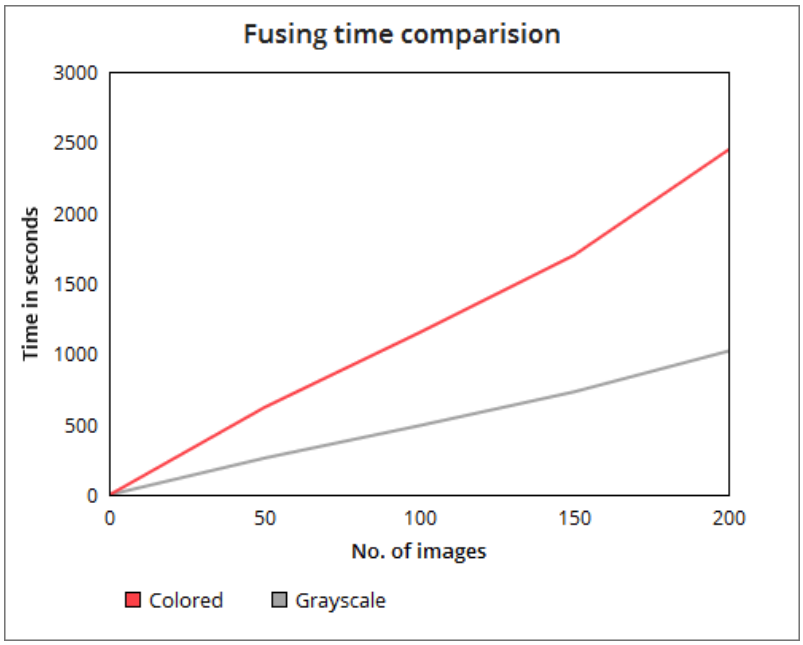

Figure 33. Fusing Time Comparison

6.4. Testing

6.4.1. Logic against Limitation

This can be called a type of acceptance testing, which is done to make sure that all the requirements discussed in Chapter 1 were met or not. The two main requirements were:

1. Protect Color Information
2. Supports Automatic Image Registration and Fusion of Large Data Set

As compared to existing methods which lack in fulfilling one or both requirements, our application does meet both the requirements. This testing was conducted when development of application was completed.

6.4.2. Unit Testing

This testing was done when each feature of the application was developed. It is conducted by testing each function-module independent of the system. This type of testing allows

identification and fixing of bugs at early stages of development. Some of the test case scenarios were:

1. Uploading a color image in Single Processing – Grayscale tab: In this case the application simply converts the color image into grayscale and displays the image with a warning – converting images into grayscale.
2. Uploading a grayscale image in Single Processing – Color tab: In this case nothing would be uploaded, an error would pop-up saying Image is not colored.

CHAPTER 7. CONCLUSION

7.1. Conclusion

Artificial Intelligence and Image Processing goes hand in hand. There have been significant advances in image processing techniques used for training and testing data models. As we have mentioned before, current image processing techniques revolves around grayscale images. Majority of AI models have been developed to work with grayscale images as the color of the images didn't play significant role when it comes to process of detection. Lately, there has been change in requirements, especially in agriculture sector where color of the image plays a vital role.

The purpose of this research was to come up with a method for image registration and fusion that supports a large data set of both grayscale and colored images. The protected color information in our case was used to determine disease in wheat plants, which would be hard to identify if the images were converted to grayscale. The result data set of fused images would be later used to train and test AI models built for detection of wheat disease.

7.2. Future Work

In any application, there is always a scope of further improvement. Since image processing has been recognized more than ever now, and probably the graphs for its applications would go up in future. Currently our application has pre-requisite that the images uploaded should already be denoised and perfectly segmented, so that the ROI is already identified. Our application only performs image registration and fusion and provides the ability to download the fused images. This app could be extended to include these pre-requisite processes as well. The user wouldn't have to worry about handling these processes separately.

The other possible extension to this application could be to offer more variety of image registration and fusion techniques to user to choose from. A technique that gives best result to our

input data set might not give best result to others, depending on requirements. With the variety of techniques to choose from, the user might be able to compare the result from each technique and decide accordingly.

7.3. Extension to Related Work

As we mentioned before, this application was particularly developed to significantly help in agriculture sector. The fused images received as output from IRFA can be used to train data models. In our case, the data model helps in early detection of crop diseases, which potentially would help in improving quality and quantity of harvest. This proposed methodology can also help in identifying severity of infected fruits produce such as apples and pear and differentiating them with the good produce.

The output fused images obtained from IRFA could help in other fields as well such as medical industry. We can extend our proposed image registration and image fusion method to wound healing assessment. A comparison between a non-injured skin and an injured skin with different level of injury severity would significantly help medical industry in better understanding and analyzing the wounds and prescribing accordingly. This technique could also be extended for detection of lesions in human digestive tracts, differentiating bowel lesions from congestive gastropathy, etc. It could also be extended for bone fracture detection and monitoring it’s healing status.

REFERENCES

[1] X. Jin, Q. Jiang, S. Yao, D. Zhou, R. Nie, J. Hai, K. He, "A survey of infrared and visual image fusion methods" in Infrared Physics and Technology, vol. 85, pp. 478-501. Sept. 2017, doi: 10.1016/j.infrared.2017.07.010

[2] D. Muthukumaran, M.Sivakumar ,"Medical Image Registration: A Matlab Based Approach," in International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), vol. 2, pp. 235-239, Feb. 2017, doi: 10.1109/IIJSRCSEIT.2017.39.

[3] M. Pradeep, "Implementation of image fusion algorithm using MATLAB (LAPLACIAN PYRAMID)," in International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), pp. 165-168, Mar. 2013 doi: 10.1109/iMac4s.2013.6526401.

[4] S. Li, X. Kang and J. Hu, "Image Fusion with Guided Filtering," in IEEE Transactions on Image Processing, vol. 22, pp. 2864-2875, July 2013, doi: 10.1109/TIP.2013.2244222

[5] V.P.S. Naidu, B. Elias, "A Novel Image Fusion Technique using DCT based Laplacian Pyramid" in International Journal of Inventive Engineering and Sciences (IJIES), vol. 2, Sep. 2013 ISSN: 2319–9598

[6] U. Patil and U. Mudengudi, "Image fusion using hierarchical PCA.," in International Conference on Image Information Processing, pp. 1-6, Nov. 2011, doi: 10.1109/ICIIP.2011.6108966

[7] B. Yang and S. Li, "Multifocus Image Fusion and Restoration with Sparse Representation," in IEEE Transactions on Instrumentation and Measurement, vol. 59, pp. 884-892, April 2010, doi: 10.1109/TIM.2009.2026612

[8] D.M.Bulanon ,T.F.Burks V.Alchanatis, "Image fusion of visible and thermal images for fruit detection" in Biosystems Engineering , vol. 103, pp. 12-22,May 2009

[9] V.P.S. Naidu and J.R. Raol National Aerospace Laboratories, Bangalore-160 017, "Pixel-level Image Fusion using Wavelets and Principal Component Analysis" in Defense Science Journal, vol. 58, pp. 338-352, May 2008, doi: 10.14429/dsj.58.1653

[10] T. Wan, N. Canagarajah and A. Achim, "Compressive image fusion," in 15th IEEE International Conference on Image Processing, pp. 1308-1311, Oct.2008, doi: 10.1109/ICIP.2008.4712003.

[11] F. Sadjadi, "Comparative Image Fusion Analysis," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops, vol. 1, pp. 8, 2005, doi: 10.1109/CVPR.2005.436.

[12] Z. Wang, D. Ziou, C. Armenakis, D. Li and Q. Li, "A comparative analysis of image fusion methods," in IEEE Transactions on Geoscience and Remote Sensing, vol. 43, pp. 1391-1402, Jun. 2005, doi: 10.1109/TGRS.2005.846874

[13] X. Otazu, M. Gonzalez-Audicana, O. Fors and J. Nunez, "Introduction of sensor spectral response into image fusion methods. Application to wavelet-based methods," in IEEE Transactions on Geoscience and Remote Sensing, vol. 43, pp. 2376-2385, Oct. 2005, doi: 10.1109/TGRS.2005.856106

[14] H. Singh, J. Raj, G. Kaur and T. Meitzler, "Image fusion using fuzzy logic and applications," in IEEE International Conference on Fuzzy Systems, vol.1,pp. 337-340, Jul. 2004, doi: 10.1109/FUZZY.2004.137574.

[15] V. S. Petrovic and C. S. Xydeas, "Gradient-based multiresolution image fusion," in IEEE Transactions on Image Processing, vol. 13, pp. 228-237, Feb. 2004, doi: 10.1109/TIP.2004.823821

[16] Z. Xue and R. S. Blum, "Concealed Weapon Detection Using Color Image Fusion," in the 6th International Conference on Image Fusion, Queensland, Australi, pp. 622-627, Jul. 2003, doi: 10.1109/ICIF.2003.177504.

[17] G. Piella and H. Heijmans, "A new quality metric for image fusion," in International Conference on Image Processing, pp. III-173, Sep 2003, doi: 10.1109/ICIP.2003.1247209.

[18] D.W. Townsend, S.R. Cherry, "Combining anatomy and function: the path to true image fusion. Eur Radiol" in European radiology, vol. 11, pp. 1968–1974, 2001, doi: 10.1007/s003300101007

[19] Nunez, X. Otazu, O. Fors, A. Prades, V. Pala and R. Arbiol, "Multiresolution-based image fusion with additive wavelet decomposition," in IEEE Transactions on Geoscience and Remote Sensing, vol. 37, pp. 1204-1211, May 1999, doi: 10.1109/36.763274.

[20] A. Toet, J. Walraven, "New false color mapping for image fusion," in Optical Engineering, vol.35, pp. 650-658, Mar. 1996, doi: 10.1117/1.600657

[21] L. J. Chipman, T. M. Orr and L. N. Graham, "Wavelets and image fusion," in International Conference on Image Processing, vol.3 pp. 248-251, 1995, doi: 10.1109/ICIP.1995.537627

[22] A. Toet, " A. Hierarchical image fusion. Machine" in Machine Vision and Application. vol. 3, pp. 1–11, 1990, doi: org/10.1007/BF01211447

[23] The Online Resource for Research in Image Fusion www.imagefusion.org

[24] The Online Resource for MATLAB www.mathworks.com