

PROJECT HANDBOOK

**Project: Wound Analysis based on Image processing**

Course: Information Engineering and Computer Science

Semester: 2

Teacher: Prof. Dr. Timo Kahl

Project leader: Arun Kumar Holla Bommanabillu Raghavendra (24349)

Project members: Amir Abdullah Md Faruk (24350)

Anith Shaji (22914)

Jefferson .o. Akhigbe (21603)

Qurratul-Ain Abid (21037)

Mayuri Jain (24499)

**Index**

1 Document History 3

2 Project Charter 5

3 Project Aims 5

4 Project Organisation 6

5 Work Breakdown Structure (WSP) 8

6 WSP Specification/ Description 10

7 Project Milestones 11

8 Project Schedule Plan 12

9 Project Costs & Time Tracking 13

10 Project Risk 13

11 Miscellaneous............................................................................................................................... 14

12 Wound analysis algorithms............................................................................................................ 17

# Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Change History** | | | |
| **Version-Number** | **Date** | **Changes** | **Author** |
| 0.1 | 04/11/17 | Handbook initial version | Arun Holla |
|  |  |  |  |
|  |  |  |  |

# Project Charter

|  |  |
| --- | --- |
| **Project Charter** | |
| **Project Aim** (Output):  Introducing a system that implements:   * The concept of AI and image comparison algorithm. * Focusing on wound detection and healing using a web platform. * Automated system for the detection and diagnosis of wounds in early stages. * Generate Medical prescription with QR code.   Enhancement of existing ‘E-Welfare ‘project, by addition of above mentioned features. | **Non-Aims / Non-Content**:   * Scope can be expanded to a variety of disease treatment. * In a distant future, this project can be implemented on a standalone machine Ex : A robot, which would eliminate the need for human intervention. |
| **Project Achievements** (Outcome):  Early detection and diagnosis of wound and treatment to the patient with limited assistance from a medical professional. | |
| **Project Client**: | **Project Leader**: |
| **Project Members**:   * Arun Kumar Holla Bommanabillu   Raghavendra   * Amir Abdullah Md Faruk * Anith Shaji * Jefferson .o. Akhigbe * Qurratul-Ain Abid * Mayuri Jain | **Other Stakeholders**:   * Patients. * Medical Personals. |
| **Major Activities / Project Phases**:   * Website development. * Hardware Integration. * Integration and Deployment on azure platform. | **Milestones**:   * First month * Second month * Third month   (More detailed information is in page 7) |
| **Project Start**: 21 October,2017 | **Project End**: 29 January,2018 |
| **Project Costs / - Efforts**:   * Research on Comparison algorthm. * Website coding and design . * Image processing. * Simulation tools. * Data storage. | **Project Risks**:   * Tight timeframe: for data collection and research. * Project team: we have complete confidence in the project team but also realize that the individuals on the team have other demands on their time that might, at times, take priority. * Techniacal Risk : Security of collected data,algorithm implementation for image comparison. * Cost : Cost of hardware devices. |
| ……………………………………………..  <Date>, Signature Project Leader | ……………………………………………..  <Date>, Signature Project Supervisor (Teacher) |

# 

# 3 Project Aims

|  |  |  |
| --- | --- | --- |
| **Plan of Project Aims** | | |
| **Type of Aim** | **Project Aims** | **Adapted Aims (incl.** **<Date>)** |
| **Project Aim (Output):** | To introduce a system that implements the concept of AI and image comparison algorithm, focusing on wound detection and healing using a web platform, Automated system for the detection and diagnosis of wounds in early stages. Also to generate medical prescription with scan code. |  |
| **Sub-Goals:** | **Software Phase:**  Coding, designing and data storage management.  **Hardware Phase:**  Image scanner and camera (smart phone). |  |
| **Non-Aims / Non-Content:** | * Scope can be expanded to a variety of disease treatment. * In a distant future, this project can be implemented on a standalone machine Ex: A robot, which would eliminate the need for human intervention. |  |

# Project Organisation

|  |  |  |
| --- | --- | --- |
| **Project Organisation** | | |
| **Role** | **Role Description** | **Name** |
| Project Client | * Treating patients and prescribing medicines | * Doctors and Pharmacist |
| Project Leader | * Hardware Interfacing and Algorithm implementation. | * Arun Kumar Holla Bommanabillu Raghavendra |
| Project Member | * Web development * Web development * Database * Processing Algorithm * Database | * Amir Abdullah Md Faruk, * Anith Shaji , * Jefferson .o. Akhigbe, * Qurratul-Ain Abid * Mayuri Jain |

Mayuri Jain

Arun Kumar Holla BommanabilluRaghavendra

Jefferson .o. Akhigbe

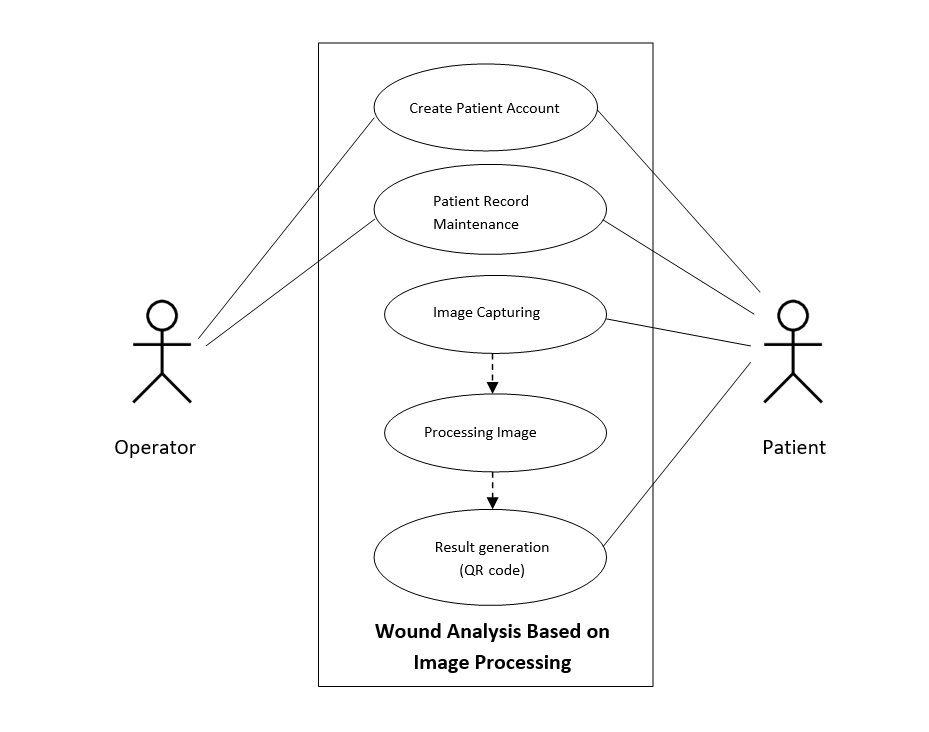
Qurratul-Ain Abid

Anith Shaji

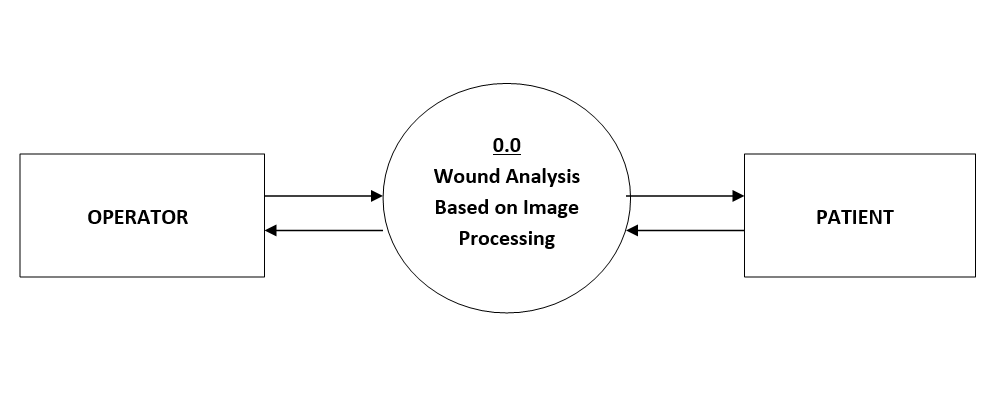
Amir Abdullah MdFaruk

Patients, Medical personnel

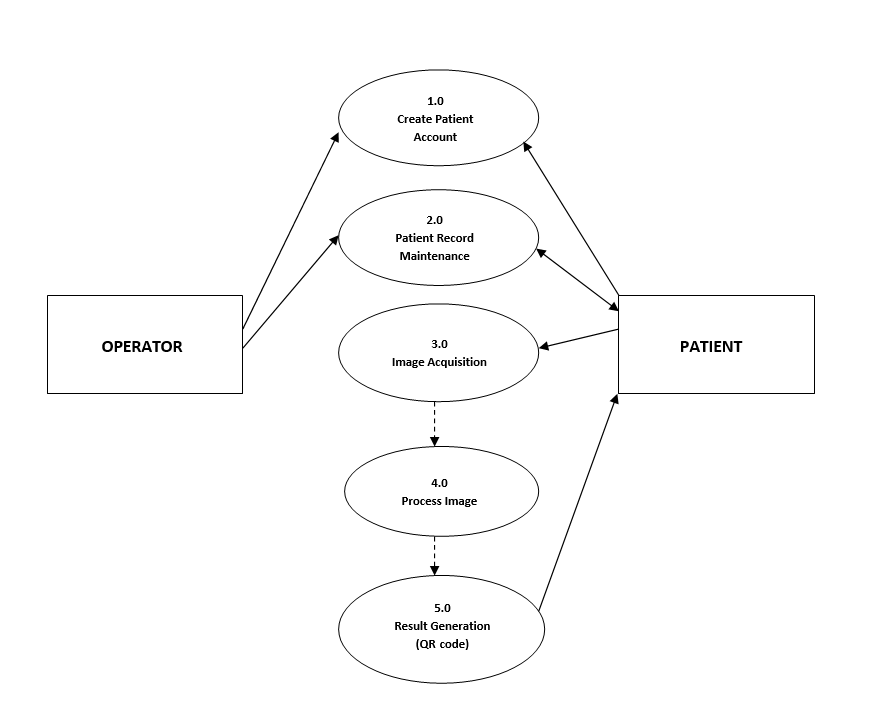
# . Work Breakdown Structure



*Fig: Use Case Diagram*



*Fig: Level 0 DFD*



*Fig: Level 1 DFD*

1. **WSP Specification/ Description**

Describe the most important work packages of your WSP.

|  |  |  |
| --- | --- | --- |
|  |  | **WP Specification** |
| **Work Package:** 1.0 Research | |  |
| WP Content / Results: |  | Research on image processing algorithm, hardware, types of |
|  |  | wound, design and development tools. |
|  |  |  |
| Responsible Person: |  | All group members |
|  |  |  |
| Progress since last status |  | Research on Image processing algorithms : Anith, Qurratul-Ain and Arun. |
| report: |  | Research on Types of Wound: Mayuri and Jefferson. |
|  |  | Research on Hardware: Amir. |
|  |  |  |
| Open issues: |  | <Description> |
|  |  |  |

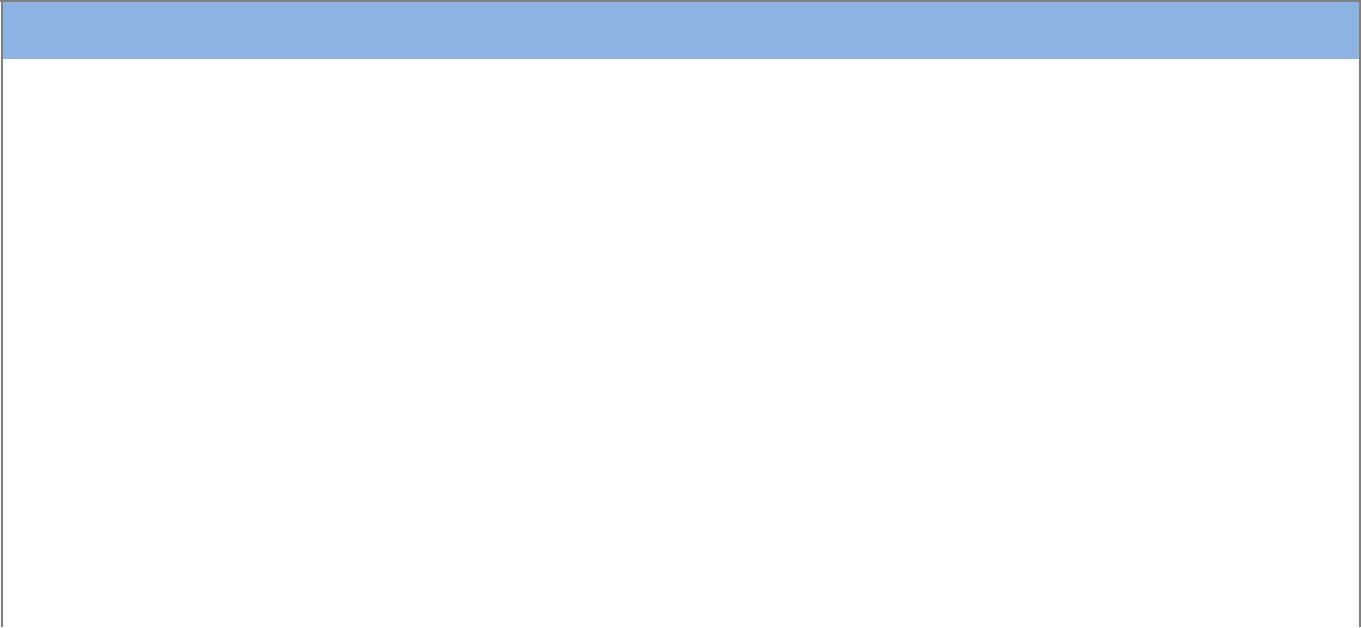
|  |  |
| --- | --- |
|  | **WP Specification** |
| **Work Package:** 2.0 User Interface Design and Development | |
| WP Content / Results: | User Interface Design and Coding |
|  |  |
| Responsible Person: | Anith and Amir |
|  |  |
| Progress since last status | <Description> |
| report: |  |
|  |  |
| Open issues: | <Description> |
|  |  |

|  |  |
| --- | --- |
|  | **WP Specification** |
| **Work Package:** 3.0 Image Processing and Hardware Integration | |
| WP Content / Results: | Implementation of image processing algorithm and hardware. |
|  |  |
| Responsible Person: | Qurratul Ain and Arun |
|  |  |
| Progress since last status | <Description> |
| report: |  |
|  |  |
| Open issues: | <Description> |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | **WP Specification** |  |
| **Work Package:** 4.0 Databases | |  |
| WP Content / Results: | Database designing and development |  |
|  |  |  |
| Responsible Person: | Jefferson and Mayuri |  |
|  |  |  |
| Progress since last status | <Description> |  |
| report: |  |  |
|  |  |  |
| Open issues: | <Description> |  |
|  |  |  |

|  |  |
| --- | --- |
|  | **WP Specification** |
| **Work Package:** 5.0 Testing and Deployment | |
| WP Content / Results: | Testing and Final Deployment |
| Responsible Person: | All group members |
|  |  |
| Progress since last status | <Description> |
| report: |  |
|  |  |
| Open issues: | <Description> |
|  |  |

1. **Project Milestones**

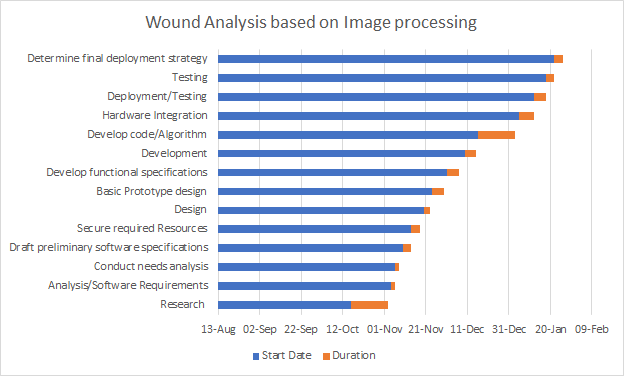


**Milestones**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **WSP-Code** | **Milestone-** |  | **START-DATE** |  | **END-DATE** |  |
|  | **Name** |  |  |  |
|  |  |  |  |  |  |  |
| 1.0 | Initial Research | Research | 16 | October | 4 November | |  |
|  | |  |  | |  |  |  |
| 2.0 Analysis/Software | | Software Requirement | 4 November | | 14 | November |  |
| Requirements | | Specification |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| 3.0 | Design | UI Design and Code | 20 | November | 15 | December |  |
|  |  |  |  | |  |  |  |
| 4.0 | Development | Image processing and | 5 December | | 10 | January |  |
| (Phase 1) | | hardware integration |  |  |  |  |  |
|  |  |  |  | |  | |  |
| 4.0 | Development | Databases | 5 December | | 5 January | |  |
| (Phase 2) | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| 5.0 | Deployment | Deployment and | 12 | January | 24 | January |  |
|  |  | Testing |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

1. **Project Schedule Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.I No.** | **Start Date** | **Duration** | **Task Names as in WBS** |
| T\_1.0 | 16-Oct | 18 | Research |
| T\_2.0 | 04-Nov | 2 | Analysis/Software Requirements |
| T\_2.1 | 06-Nov | 2 | Conduct needs analysis |
| T\_2.2 | 10-Nov | 4 | Draft preliminary software specifications |
| T\_2.3 | 14-Nov | 4 | Secure required Resources |
| T\_3.0 | 20-Nov | 3 | Design |
| T\_3.1 | 24-Nov | 6 | Basic Prototype Design |
| T\_3.2 | 01-Dec | 6 | Develop functional specifications |
| T\_4.0 | 10-Dec | 5 | Development |
| T\_4.1 | 16-Dec | 18 | Develop code/Algorithm |
| T\_4.2 | 05-Jan | 7 | Hardware Integration |
| T\_5.0 | 12-Jan | 6 | Deployment/Testing |
| T\_5.1 | 18-Jan | 4 | Testing |
| T\_5.2 | 22-Jan | 4 | Determine final deployment strategy |



*Fig: Gantt Chart*

1. **Project Costs and Time Tracking**

|  |  |  |
| --- | --- | --- |
| **Work Package / Milestone** | **Type of Costs** | **Amount/Effort** |
| **Name** | **Ex: Personal, material** | **Ex: 5 hours** |
| Barcode scanner | material | 15 hours |
| Camera | material | 5 hours |
| Clouding system | material | 15 hours |

**10.Project Risks**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Event risk** | **Impact** | **measures** |
| **NAME AND DISCRIPTION** | **in %** | **Type of object** | **Wbs code name** |
| Integration of hardware and software | 30% | Barcode scanner |  |
| Software specifications | 30% | R coding / Python |  |
| Camera of choice | 10% | Digital camera/ smart phone camera |  |
| Integration of database to cloud | 30% | Azure/ docker cloud |  |

**11) Miscellaneous :**

**1. Research Topics Detail**

|  |  |  |
| --- | --- | --- |
| 1. | Anith | Research on different types of Image comparison Algorithms |
| 2. | Qurratul Ain | Research on different types of Image comparison Algorithms |
| 3. | Arun | Research on different types of Image comparison Algorithms |
| 4. | Jeff | Searching types of Wounds |
| 5. | Mayuri | Searching types of Wounds |
| 6. | Amir | Research on hardware device for Images |

**2. Task Distribution List**

1. Architecture design.
2. Layering the system.
3. Front end design.
4. Back end design.
5. Algorithm integration.
6. Code testing in R, Python, C#, MAT lab.
7. Wound Image collection for training the system.
8. Hardware integration.
9. Entity search and finalisation.
10. Database SQL server set up and implementation(primary publishing)
11. IBM cloud set up and implementation (secondary publishing)
12. Testing.

|  |  |  |
| --- | --- | --- |
| # | Tasks | Assigned to |
| 1 | Architecture design. | Anith |
| 2 | Layering the system. | Anith |
| 3 | Front end design. | Qurratul Ain |
| 4 | Back end design. | Amir |
| 5 | Algorithm integration methods. | Arun, Qurratul Ain |
| 6 | Image comparison Code testing in R. | Anith |
| 7 | Image comparison Code testing in Python | Arun |
| 8 | Image comparison Code testing in C# | Amir |
| 9 | Image comparison Code testing in MAT lab | Mayuri, Qurratul Ain |
| 10 | Wound Image collection for training the system. | Jeff,Mayuri |
| 11 | Entity search and finalisation. | Qurratul Ain,Arun |
| 12 | Hardware integration. | Amir,Arun |
| 13 | Database sql server set up and implementation(primary publishing) | Anith,Amir |
| 14 | IBM cloud set up and implementation (secondary publishing) | Jeff,Arun |
| 15 | Testing . | Mayuri, Jeff |

**3. Flow chart (Function flow)**

1. **PATIENT :**



1. **OPERATOR :**

Start

Send email with rejection reason

Send email with account details

End

Verify Patient details

View Patient Details

Delete Patient Record

Requested Patient List

Enrolled Patient List

Website Tab 2

Website Tab 1

Operator Login

**Wound analysis algorithms.**

**Research for implementation in c#**

* Algorithm shortlisted and used:

### K Nearest Neighbours – Classification

### K nearest neighbours: stores all available cases and classifies new cases based on a similarity measure.

### SURF Algorithm

### Three main parts: interest point detection, local description and matching.

### In the sample program:

### Clustering algorithm and Speeded up Robust Features (SURF) is used to perform image classification.

### A set of images are initially loaded and put it into different containers.

* Use SURF, so in order to use a standard clustering algorithm that is based on Euclidean distances.
* A good algorithm for clustering code word is the Binary Split variant of the K-Means algorithm.
* Training based on the set of images available or pre loaded into the system
* Use it to extract representations for each of the images in both training and testing sets.
* Creates the Support Vector Machines that will identify images based on their Bag-of-Visual-Words feature vector representation.
* Automatically estimates a good starting point for Gaussian's sigma parameter using initialization heuristics.
* Method automatically estimates a good starting point for the complexity parameter (C) of the SVM learning algorithm.
* Classifies images into one of the possible classes using the Support Vector Machines learned in the previous steps.

All the methods used have inbuilt files or algorithm used in Visual studio in the form of classes. But still has some error facing for the integration into our system.

**Research implementation in MATLAB**

* **Algorithm shortlisted and used:**

First part of image comparison includes detecting edges of the image for this canny edge detection algorithm is used.

### Canny Edge Detection Algorithm

The Canny algorithm is flexible to different environments. Its parameters allow it to be tailored to detection of edges of differing characteristics depending on the specific requirements.

### 

The Process of Canny edge detection algorithm can be broken down to 4 different steps:

**1.) Smooth image with a Gaussian**

Since edge detection results are easily affected image noise, so Gaussian filter is applied to smooth the image in order to remove noise to prevent false detection.

**2). Find the intensity gradients of the image**

An edge in an image may points in different directions, so the Canny algorithm uses filters to detect horizontal, vertical and diagonal edges.

**3). Applying non-maximum suppression**

Thin edges by applying non-maxima suppression to the gradient magnitude to get rid of spurious response to edge detection

**4). Hysteresis threshold**

Next is to filter out edge pixel with weak and strong gradient value. If an edge pixel’s gradient value is higher than the high threshold value, it is marked as a strong edge pixel. If an edge pixel’s gradient value is smaller than the high threshold value and larger than the low threshold value, it is marked as a weak edge pixel. Weak edges are discarded in the end.

**Wound edge detection using JavaScript**

Using this JavaScript algorithm(Figure 1) we can detect edge of any wound.This code is compatible with Visual studio.

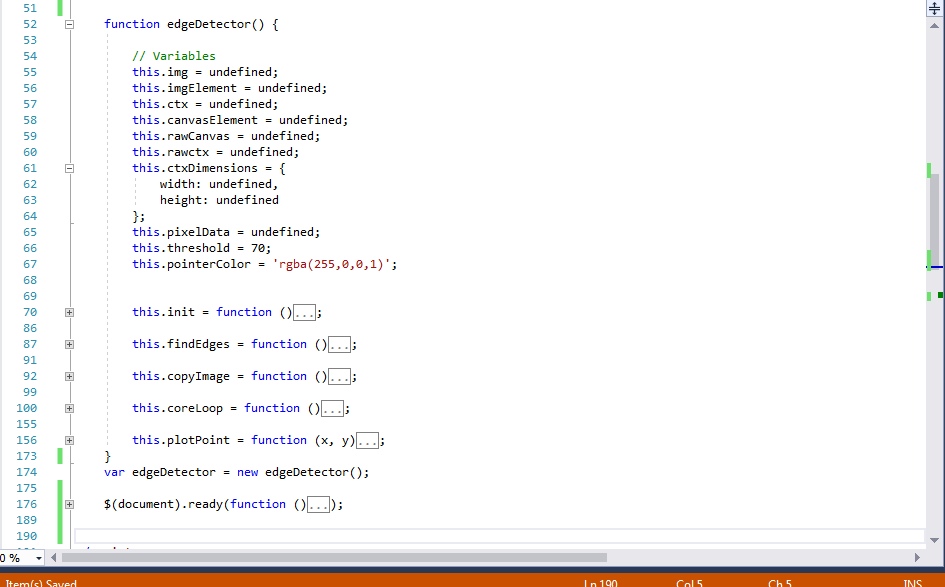


Figure 1: JavaScript Algorithm for wound detection

Detection edge is one of the important parts of our research project. Therefore we can use this algorithm for our web application where user will upload their wound image. After uploading their image we can show the user the exact edge of the wound (Figure 2).

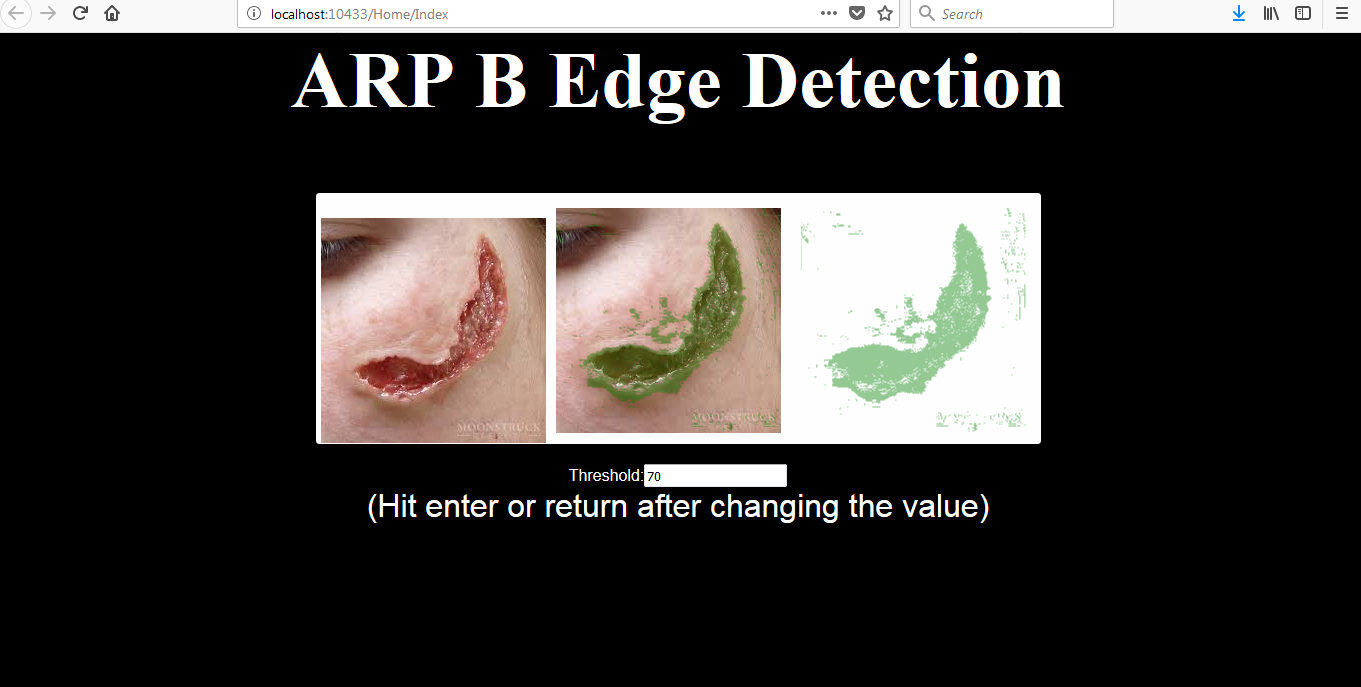


Figure2: Detect wound edge of the user uploaded image(first one is user uploaded image)

However user can change threshold to make the detection more perfect. There is an input field (Figure 3) where user can change threshold value and press enter to view the change. When the user is done with the threshold value, he can click on a submit button and the system will store uploaded the image with only edge.

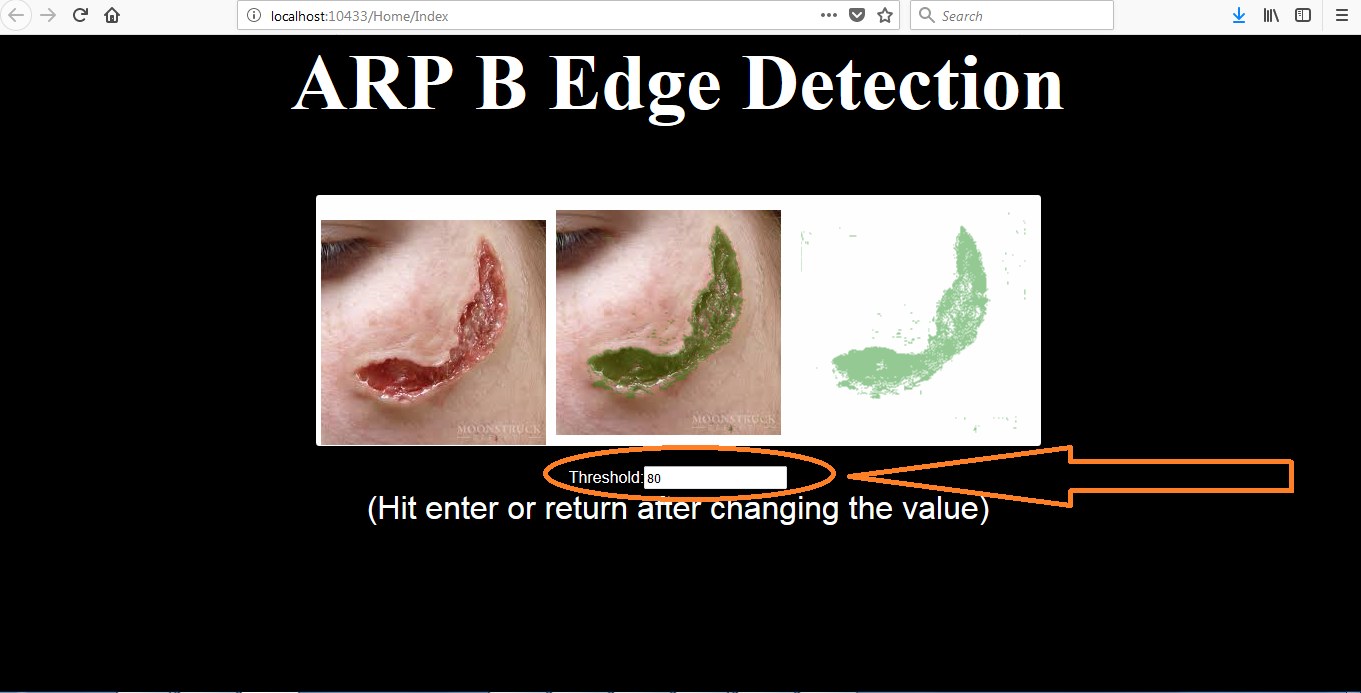


Figure 3: User can change threshold to make the wound edge detection perfect

Image comparision and Edge detection in Python

1. The first step the user runs the algorithm, then the camera is initiated for snapping the picture, it takes into consideration the distance and focuses the camera lenses

2. Image taken and they are compared to 10 or more images stored in the database and the closest image is stored in the database

3. The output is processed and the image with the highest matching pattern and also with current edge image pattern is produced and displayed.

1. initialize the camera

import numpy as np

import cv2

from matplotlib import pyplot as plt

from PIL import Image

def take\_and\_save\_picture(im\_save):

'''Take a picture and save it

Args:

im\_save: filepath where the image should be stored

'''

camera\_port = 0

ramp\_frames = 30

cap = cv2.VideoCapture(camera\_port)

def get\_image():

retval, im = cap.read()

return im

for i in xrange(ramp\_frames):

temp = get\_image()

print("Taking image...")

# Take the actual image we want to keep

camera\_capture = get\_image()

#im\_save\_tmp = im\_save + '.jpg'

im\_save\_tmp = im\_save

# A nice feature of the imwrite method is that it will automatically choose the

# correct format based on the file extension you provide. Convenient!

cv2.imwrite(im\_save\_tmp, camera\_capture)

# You'll want to release the camera, otherwise you won't be able to create a new

# capture object until your script exits

2. comparing in a for loop:

def get\_images\_to\_compare():

images\_to\_compare = []

while True:

comp\_img = raw\_input("Path of image to compare to: ")

if len(comp\_img) <= 1:

# break if someone just hits enter

break

images\_to\_compare.append(comp\_img)

return images\_to\_compare

def main():

#capture\_img = "/Users/Me/home1.png"

capture\_img = input('enter path of the file from database')

#img\_to\_compare = "/Users/Me/Documents/python programs/compare/img2.jpg"

take\_and\_save\_picture(capture\_img)

#### you have some odd var names here, basic gist, add a for loop

for comp\_image in get\_images\_to\_compare():

diff = compute\_edges\_diff(im1, im2)

print "Difference (percentage):", diff

if diff > 0.5:

print im1

else:

print im2

3. Displays output of images taken

def main(folder\_path\_to\_search, files\_to\_compare\_to, source\_image\_path):

#capture\_img = "/Users/Me/home1.png"

capture\_img = input('enter path of the file from database')

#img\_to\_compare = "/Users/Me/Documents/python programs/compare/img2.jpg"

take\_and\_save\_picture(capture\_img)

images\_to\_compare = [ os.path.join(folder\_path\_to\_search,file\_path) for file\_path in os.listdir(folder\_path\_to\_search) if file\_path.endswith(files\_to\_compare\_to) ]

for comp\_image in get\_images\_to\_compare():

diff = compute\_edges\_diff(source\_image\_path, comp\_image)

print "Difference (percentage):", diff, "(", source\_image\_path, ":", comp\_image, ")"

if \_\_name\_\_ == '\_\_main\_\_':

folder\_path\_to\_search = raw\_input("Enter folder path to search")

files\_to\_compare\_to = raw\_input("enter file extention to glob ex: '.jpg'")

source\_image\_path = raw\_input("enter full file path of source image")

main(folder\_path\_to\_search, files\_to\_compare\_to, source\_image\_path)

**Abc Image analysis using edge detection using MATLAB**

There are number of edge detection operators available, each designed to be sensitive to certain type of edges. The quality of edge detection can be measured from several criteria objectively. Some criteria are proposed in terms of mathematical measurement, some of them are based on application and implementation requirements .In all five cases a quantitative evaluation of performance requires use of images where the true edges are known.

* **Good detection**: There should be minimum number of false edges. Usually, edges are detected after a threshold operation. The high threshold will lead to less false edges, but it also reduces the number of true edges detected.
* **Noise sensitivity**: The algorithm can detect edges in certain acceptable noise environments.   Good localization: The edge location must be reported as close as possible to the correct possible position, i.e. edge localization accuracy.
* **Orientation Sensitivity**: The operator not only detects edge magnitude, but it also detects edge orientation correctly. Orientation can be used in post processing to connect edge segments , reject noise and suppress non-maximum edge magnitude
* **Speed and efficiency**: The algorithm should be fast enough to be usable in an image processing system. An algorithm that allows recursive implementation or separately processing can greatly improve efficiency.

**Techniques of Edge Detection:**

**Robert operator:**

It is the gradient operator. The simple 2\*2 Roberts operators were one of the earliest methods employed to detect edges. The Roberts operator is implemented using two convolution masks/kernels, each designed to respond maximally to edges running at ±45º to the pixel grid, which return the image x-derivative and y derivative, Gx and Gy respectively. This method is not preferred for today’s technology as it is highly sensitive to noise and not compatible with all the elements.

**Sobel Operator:**

The Sobel operator performs a 2-D spatial gradient measurement on an image. It uses a pair of 3×3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (Gx and Gy). Errors due to effects of noise are reduced by local averaging within the neighborhood of the mask.

**The Prewitt filter:**

The Prewitt filter is very similar to Sobel operator. The 3x3 total convolution masks is used to detect gradient in the X, Y directions .Prewitt filter is a fast method for edge detection The Prewitt/Sobel kernels are generally preferred to the Roberts approach because the gradient is not shifted by half a pixel in both directions and extension to larger sizes (for filter neighbor hoods greater than 3\*3) is not readily possible with the Roberts operators. The key difference between the Sobel and Prewitt operators is that the Sobel kernel implements differentiation in one direction and (approximate) Gaussian averaging in the other. . It is only suitable for well-contrasted noiseless images. The advantage of this is that it smoothes the edge region, reducing the likelihood that noisy or isolated pixels will dominate the filter response.

In the following code, The Prewitt filter method is implemented in which an image is being taken from a camera and edge detection is applied onto it. The name of the image is “abc.jpg”

function varargout = abc(varargin)

% Begin initialization code

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @abc\_OpeningFcn, ...

'gui\_OutputFcn', @abc\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

% End initialization code

% --- Executes just before abc is made visible.

function abc\_OpeningFcn(hObject, eventdata, handles, varargin)

data1=imread('abc\_pic.jpg');

axes(handles.axes2);

imshow(data1);

clear data1;

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

varargout{1} = handles.output;

% --- Executes on button press in start.

function start\_Callback(hObject, eventdata, handles)

vid=videoinput('winvideo',2,'YUY2\_640X480');

set(vid,'TriggerRepeat',Inf);

vid.returnedcolorspace='rgb';

vid.FrameGrabInterval=2;

flag=1;

while (flag==1)

data=getsnapshot(vid);

diff=im2bw(data,0.60);

diff=bwareaopen(diff,100);

sum=0;

axes(handles.axes1);

diff=imerode(diff,strel('disk',12));

diff=imerode(diff,strel('line',3,45));

bw=bwlabel(diff,8);

stats=regionprops(bw,'Area','BoundingBox','Centroid');

imshow(data);

hold on

for object=1:length(stats)

if(stats(object).Area>1300&&stats(object).Area<6500)

bb=stats(object).BoundingBox;

bc=stats(object).Centroid;

rectangle('Position',bb,'EdgeColor','Y','LineWidth',2);

x1=bc(1);

y1=bc(2);

plot(x1,y1,'o');

sum=sum+stats(object).Area;

% display(stats(object).Area);

end

end

hold off

density=sum/768;

ncars=round(density/3.2);

set(handles.text3,'String',num2str(density));

set(handles.text6,'String',num2str(ncars));

red=0; y=2; grn=0;

if(ncars>=0&&ncars<2)

red=7; grn=3;

end

if(ncars>=2&&ncars<5)

red=5; grn=4;

end

if(ncars>=5&&ncars<9)

red=3; grn=7;

end

if(ncars>9)

red=1; grn=10;

end

set(handles.text10,'BackgroundColor','Red');

while(red>0)

set(handles.edit1,'String',num2str(red));

red=red-1;

pause(1);

end

red=2;

set(handles.text10,'BackgroundColor','Yellow');

while(red>0)

set(handles.edit1,'String',num2str(red));

red=red-1;

pause(1);

end

set(handles.text10,'BackgroundColor','Green');

while(grn>0)

set(handles.edit1,'String',num2str(grn));

grn=grn-1;

pause(1);

end

stx=get(handles.text4,'String');

if strcmp(stx,'Approximate Cars')

flag=1;

else

flag=2;

end

if(flag==2)

close all;

clear all;

flag=2;

end

end

function stop\_Callback(hObject, eventdata, handles)

set(handles.text4,'String','Stopped');

disp('END');

function edit1\_Callback(hObject, eventdata, handles)

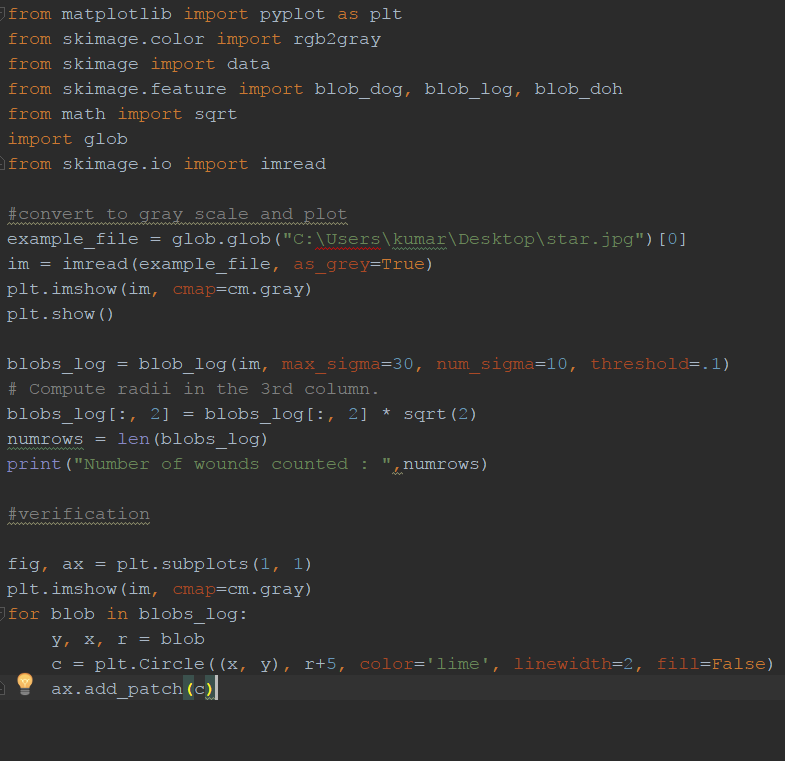
function edit1\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

Multiple wound detection using Python:



The radii of the wound can be fixed to a certain and given as input to this code

Objects will be detected based on the given radii range.