References

[1] Ciamberlini C., Francini F., Longobardi G., Sansoni P., Tiribilli, B., (1996), "Defect detection in textured materials by optical filtering with structured detectors and self adaptable masks", Opt. Eng. 35(3), pp.838-844.

[2] Cohen, F.S., Fan, Z.G., Arrai, S., (1991). , "Automated inspection of textile fabrics using textural models", 13(8), pp. 803-808.

[3] Dastoor, P.H., Radhakrishnaiah, P., Srinivasan, K., Jayaraman, S., 2004, "A Knowledge-based framework for analyzing defects in apparel manufacturing", 5(7), pp.14-20.

[4] Escofet J., Navarro R., Millán M.S., Pladellorens J., (1996), "Detection of local defects in textile webs using Gabor filters", in "Vision Systems: New Image ProcessingTechniques" Ph. Réfrégier, ed. Proceedings SPIE vol. 2785, pp.163-170.

[5] Escofet J., Navarro R., Millán M.S., Pladellorens J., (1998), "Detection of local defects in textile webs using Gabor filters", Opt. Eng. 37(8) pp.2297-2307.

[6] Hemdan, A., Abou-Taleb, Eng., Aya Tallah, M., Sallam, "On-line Fabric Defect Detection And Full Control in Circular Knitting Machine", pp.130-141.

[7] Ioannis G. Mariolis., Evangelos S. Dermatas., (2009) "Illumination invariant seam line detection in real Textiles", International Journal of Clothing Science and Technology, Vol. 21 Iss: 5, pp.286 – 299.

[8] Jasper W.J., Garnier S.J., Potlapalli H.,(1996), "Texture characterization and defect detection using adaptive wavelets", Opt. Eng. 35(11), pp. 3140-3149.

[9] Jasper W.J., Potlapalli H., (1995), "Image analysis of mispicks in woven fabric", Text. Res.J. 65(1), pp.683-692

[10] Kaicheng Yin., Weidong Yu., 2008, "Image Processing for the Use of Textile Production Detection System," cisp, vol. 3, pp.349-352.

[11] Knell, A.L., (1975), Automatic Fabric Inspection, Textile Institute and Industry, January.

[12] Mustafa Al-Eidarous, (1998) "Locating defects on shirt collars using image processing", International Journal of Clothing Science and Technology, Vol. 10 Iss: 5, pp.365 – 378.

[13] Prof. Hemdan, A., Aya Tallah, M., "On-line fabric defect detection and full control in circular knitting machine", pp.13-25.

[14] Ralló M., Millán M. S., Escofet J., (2003), "Wavelet based techniques for textile inspection", Opt. Eng. 26(2), pp.838-844

[15] Schreiner, A., 2006, "Image processing techniques for face recognition", 71-86.

[16] Shao D.X., Wang H.,Zhu L., Li J.L., 2008," Research on a New Fabric Defect Identification Method", pp. 814-817.

[17] Shen Y. J. and Zhu G. Y., "Detection of fabric defects based on artificial neural network," Silk, vol. 6, pp. 38-41, 2007.

[18] Tamnun E Mursalin1, 2 Fajrana Zebin Eishita , 3Ahamed Ridwanul Islam ., (2008), "Fabric defect inspection system using neural network and microcontroller", pp.560-569

[19] Yuen, C.W.M., Wong, W.K., Qian, S.Q., Chan, L.K., Fung, E.H.K., (2008) , "A hybrid model using genetic algorithm and neural network for classifying Textile defects" , 36(2009), pp.2037-2047.

[20] Zhang Y. X., Bresee R. R., "Fabric detection and classification using image analysis," Textile Res. J., vol. 65, pp. 1-9, 1995.

[21] http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

Appendix A - Use Case Diagram for Recognition of Textile Defects

Figure 9.1 shows the use case diagram for recognition of textile defects .It describes how the process of defects classification in terms of user is involving.

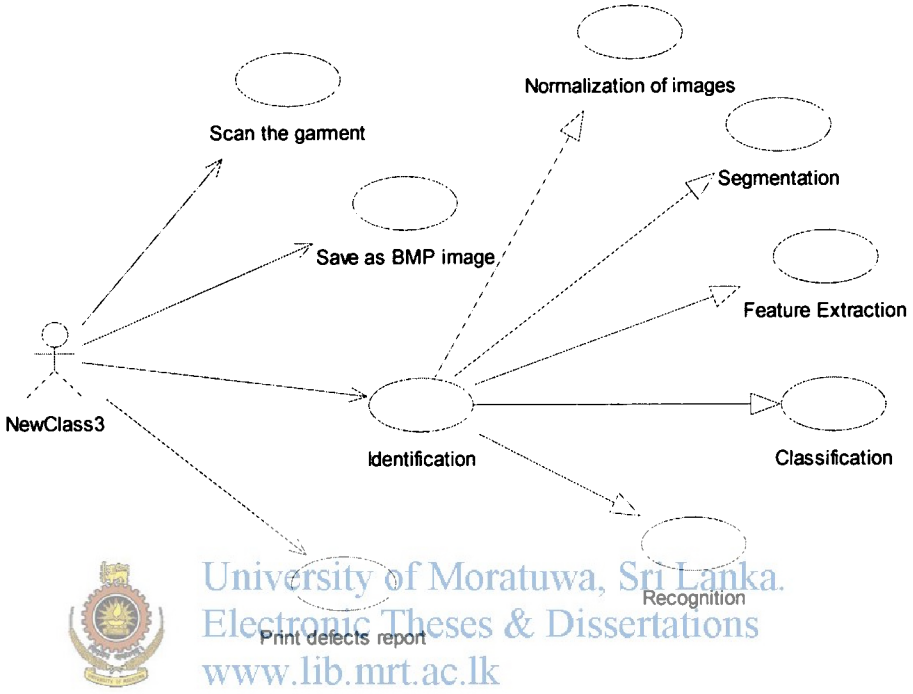

Figure A: 1 Use Case Diagram for Recognition of Textile Defects

Appendix B – Source Code of Create a Neural Network

These are the codes generated when create a neural network using Matlab nftool.

```
function net = create_fit_net(inputs,targets)
%CREATE_FIT_NET Creates and trains a fitting neural network.
%
% NET = CREATE_FIT_NET(INPUTS,TARGETS) takes these arguments:
%   INPUTS - RxQ matrix of Q R-element input samples
%   TARGETS - SxQ matrix of Q S-element associated target samples
% arranged as columns, and returns these results:
%   NET - The trained neural network
%
% For example, to solve the Simple Fit dataset problem with this function:
%
%   load simplefit_dataset
%   net = create_fit_net(simplefitInputs,simplefitTargets);
%   simplefitOutputs = sim(net,simplefitInputs);
%
% To reproduce the results you obtained in NFTOOL:
%
%   net = create_fit_net(data1',CH');

% Create Network
numHiddenNeurons = 1;  % Adjust as desired
net = newfit(inputs,targets,numHiddenNeurons);
net.divideParam.trainRatio = 70/100;  % Adjust as desired
net.divideParam.valRatio = 15/100;  % Adjust as desired
net.divideParam.testRatio = 15/100;  % Adjust as desired

% Train and Apply Network
[net,tr] = train(net,inputs,targets);
outputs = sim(net,inputs);

% Plot
plotperf(tr)
plotfit(net,inputs,targets)
plotregression(targets,outputs)
```

Appendix C – Source Code of GUI Design

```
function varargout = MainInterface(varargin)
% MAININTERFACE M-file for MainInterface.fig
%      MAININTERFACE, by itself, creates a new MAININTERFACE or raises the existing
%      singleton*.
%
%      H = MAININTERFACE returns the handle to a new MAININTERFACE or the handle to
%      the existing singleton*.
%
%      MAININTERFACE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in MAININTERFACE.M with the given input arguments.
%
%      MAININTERFACE('Property','Value',...) creates a new MAININTERFACE or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before MainInterface_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to MainInterface_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help MainInterface

% Last Modified by GUIDE v2.5 10-Feb-2011 00:11:34

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @MainInterface_OpeningFcn, ...
                   'gui_OutputFcn',  @MainInterface_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 - DO NOT EDIT


% --- Executes just before MainInterface is made visible.
function MainInterface_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to MainInterface (see VARARGIN)

% Choose default command line output for MainInterface
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes MainInterface wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = MainInterface_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in upload_button.
function upload_button_Callback(hObject, eventdata, handles)
% hObject    handle to upload_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
axes(handles.axes1)

[filename, pathname] = ...
    uigetfile({'*.bmp';'*.jpg';'*.tiff';'*.png';'*.*'},'Image Selector');
fname=strcat(pathname,filename);
global Im1;
Im1=imread(fname);
imshow(Im1);
%global name;
%name= filename;


% --- Executes on button press in capture_button.
```

```
function capture_button_Callback(hObject, eventdata, handles)
% hObject    handle to capture_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in defects_button.
function defects_button_Callback(hObject, eventdata, handles)
% hObject    handle to defects_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%x=1;  %used for testing only. Parse the x value here.
%if x==0
   %msgbox('Quality Test Passed','Quality OK','modal');
%elseif x==1
   %msgbox('Quality Test Failed','Quality Fail','modal');
%end

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in gray_button.
function gray_button_Callback(hObject, eventdata, handles)
% hObject    handle to gray_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Im1;
axes(handles.axes2)
global Im2;
Im2=rgb2gray(Im1);
imshow(Im2);
```

```
% --- Executes on button press in close_button.
function close_button_Callback(hObject, eventdata, handles)
% hObject    handle to close_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close all;
```