Software Engineering Project – COS 301

Ambitious Designs

Software User Manual

Project Team: Dian Veldsman – u12081095

Project Manager: Stephen Swanepoel – u11032091

Project Owner: Willem Burger

Abstract

This document is the Software User Manual for Smart Image Identifier. The Software User Manual instructs how to install and use the Smart Image Identifier software.

This project is part of the Software Engineering course COS 301 at the University of Pretoria.

Contents

Overview

The software implements human detection within images as well as videos. Images are retrieved from the server and processed by Smart Image Identifier which returns a response and outputs an example of the detection which gets saved into your projects source folder titled “output”.

The PeopleDetect.java file has been Doxygen commented for your convenience, detailing every method and giving a brief description of its purpose.

Tutorial

1. Installing the Software

1.1. Install Eclipse

The easiest way to build the dispatcher from source is using Eclipse Neon. Eclipse Neon is a development environment and is open-source software. The latest version of this software can be obtained at https://www.eclipse.org/downloads/. After the download is completed, install the software using the instructions provided by the installer.

1.2. Importing the project

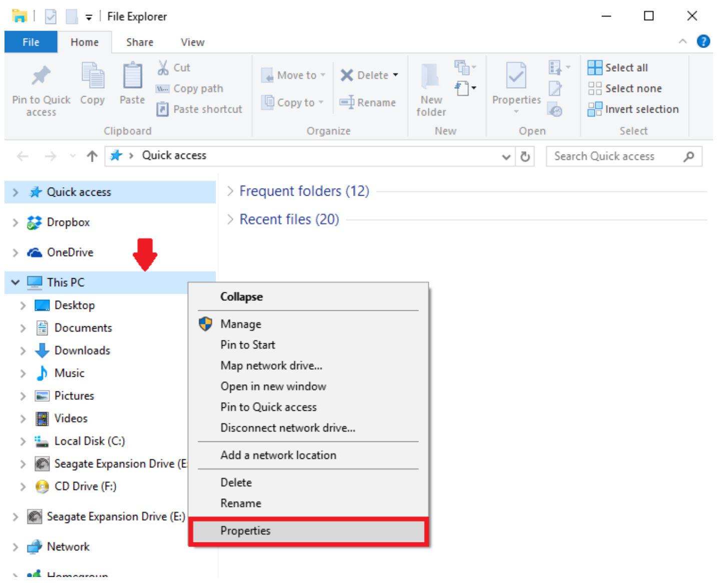
In Eclipse, click *File* > *Import* > *General* > finally, select *Existing Projects into Workspace* and click *Next*.
Select root directory > click *browse* > locate the given project folder > finally, select *Finish*.

1.3. Setting system environment variables

Setting of environment variables is necessary for your system to detect the OpenCV framework which was used for developing Smart Image Identifier.

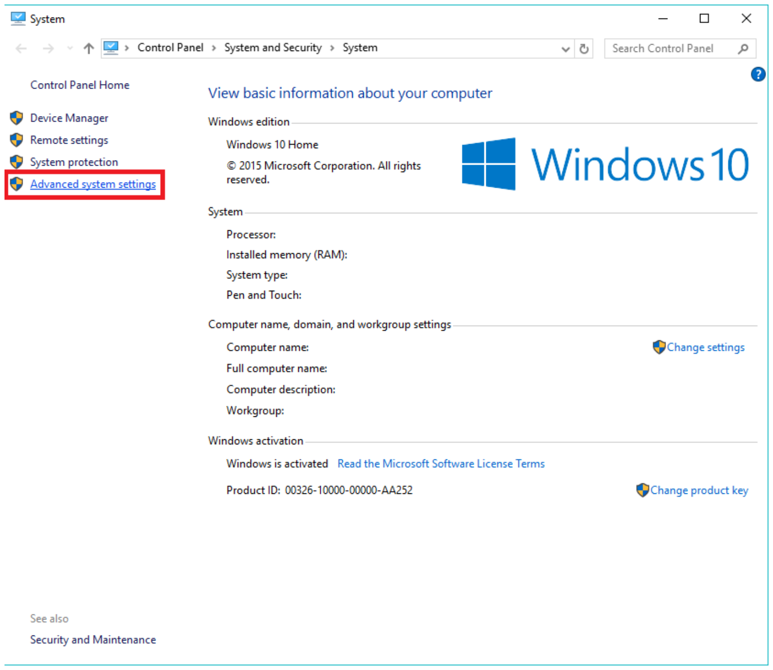
Let's get started.

1.3.1. *Step 1*



Open *File Explorer* > right click on *This PC* (Yours may be named otherwise) > click *Properties*

1.3.2. *Step 2*



View basic information about your computer

Windows edition

Windows 10 Home

© 2015 Microsoft Corporation. All rights reserved.

System

Processor:

Installed memory (RAM):

System type:

Pen and Touch:

Computer name, domain, and workgroup settings

Computer name:

Full computer name:

Computer description:

Workgroup:

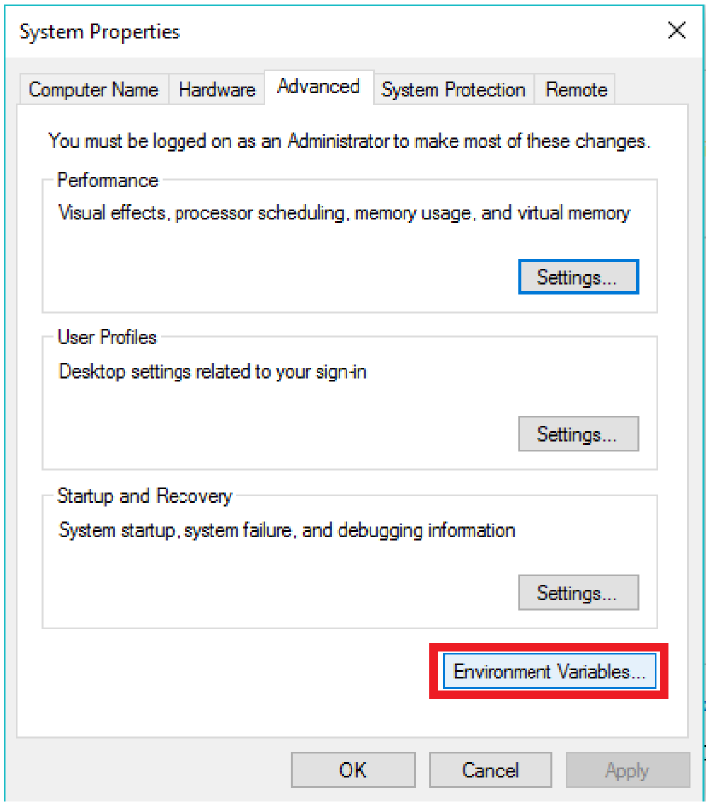
Windows activation

Windows is activated Read the Microsoft Software License Terms

Product ID: 00326-10000-00000-AA252

On the left hand panel select *Advanced System Settings*

1.3.3. *Step 3*



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...

User Profiles
Desktop settings related to your sign-in
Settings...

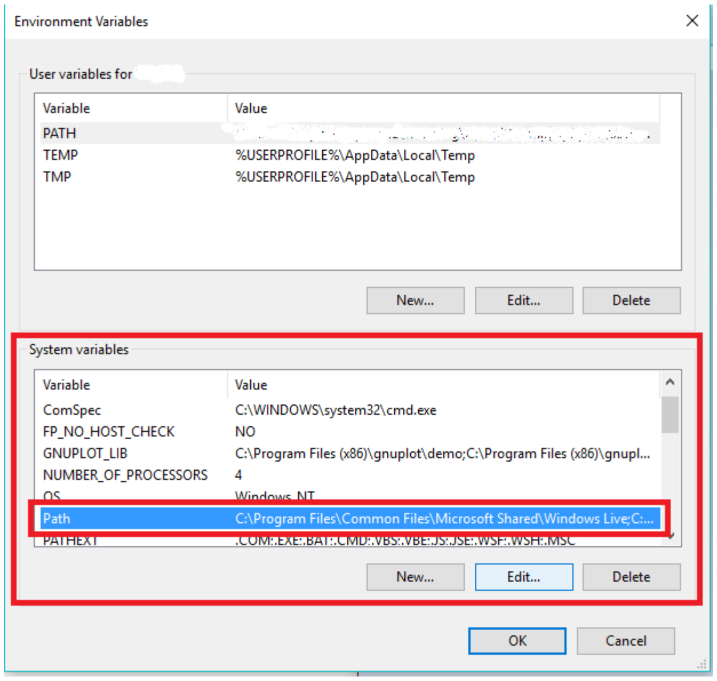
Startup and Recovery
System startup, system failure, and debugging information
Settings...

Environment Variables...

OK | Cancel | Apply

Click *Environment Variables* located at the bottom of the window

1.3.4. *Step 4*



Environment Variables

User variables for [redacted]

Variable	Value
PATH	[redacted]
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

New... | Edit... | Delete

System variables

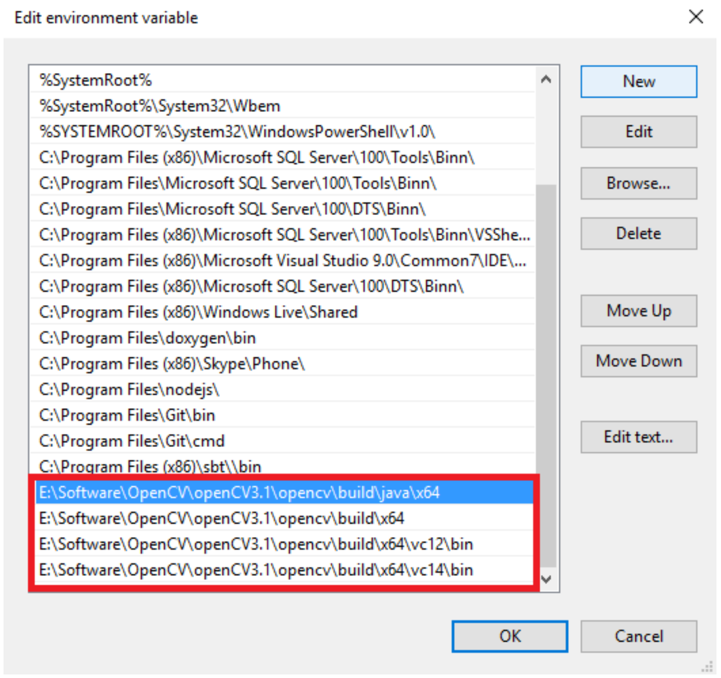
Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_CHECK	NO
GNUPLOT_LIB	C:\Program Files (x86)\gnuplot\demo;C:\Program Files (x86)\gnupl...
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files\Common Files\Microsoft Shared\Windows Live;C:...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

New... | Edit... | Delete

OK | Cancel

Select *Path* > click *Edit*

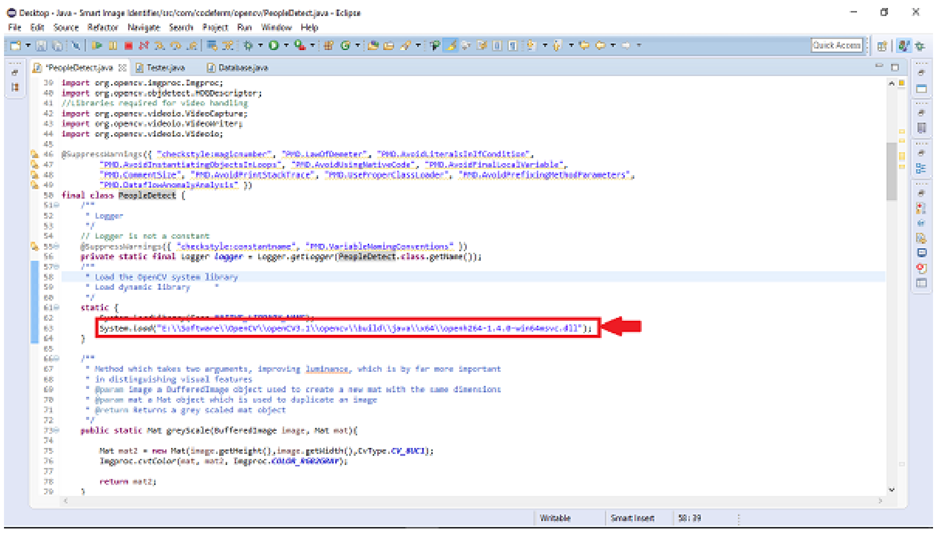
1.3.5. *Step 5*


Edit environment variable

```
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
C:\Program Files (x86)\Microsoft SQL Server\100\Tools\Binn\
C:\Program Files\Microsoft SQL Server\100\Tools\Binn\
C:\Program Files\Microsoft SQL Server\100\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\100\Tools\Binn\VSShe...
C:\Program Files (x86)\Microsoft Visual Studio 9.0\Common7\IDE\...
C:\Program Files (x86)\Microsoft SQL Server\100\DTS\Binn\
C:\Program Files (x86)\Windows Live\Shared
C:\Program Files\doxygen\bin
C:\Program Files (x86)\Skype\Phone\
C:\Program Files\nodejs\
C:\Program Files\Git\bin
C:\Program Files\Git\cmd
C:\Program Files (x86)\sbt\\bin
E:\Software\OpenCV\openCV3.1\opencv\build\java\x64
E:\Software\OpenCV\openCV3.1\opencv\build\x64
E:\Software\OpenCV\openCV3.1\opencv\build\x64\vc12\bin
E:\Software\OpenCV\openCV3.1\opencv\build\x64\vc14\bin
```

Select *New* > enter the paths to the above folders which may be located in the given OpenCV files > restart computer to take effect.

1.3.6. *Step 6*


Desktop - Java - Smart Image Identifier/src/com/codeferm/opencv/PeopleDetect.java - Eclipse

```java
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.HOGDescriptor;
//Libraries required for video handling
import org.opencv.videoio.VideoCapture;
import org.opencv.videoio.VideoWriter;
import org.opencv.videoio.Videoio;

@SuppressWarnings({ "checkstyle:magicnumber", "PMD.LawOfDemeter", "PMD.AvoidLiteralsInIfCondition",
        "PMD.AvoidInstantiatingObjectsInLoops", "PMD.AvoidUsingNativeCode", "PMD.AvoidFinalLocalVariable",
        "PMD.CommentSize", "PMD.AvoidPrintStackTrace", "PMD.UseProperClassLoader", "PMD.AvoidPrefixingMethodParameters",
        "PMD.DataflowAnomalyAnalysis" })
final class PeopleDetect {
    /**
     * Logger
     */
    // Logger is not a constant
    @SuppressWarnings({ "checkstyle:constantname", "PMD.VariableNamingConventions" })
    private static final Logger logger = Logger.getLogger(PeopleDetect.class.getName());
    /**
     * Load the OpenCV system library
     * Load dynamic library     *
     */
    static {
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        System.load("E:\\Software\\OpenCV\\openCV3.1\\opencv\\build\\java\\x64\\openh264-1.4.0-win64msvc.dll");
    }

    /**
     * Method which takes two arguments, improving luminance, which is by far more important
     * in distinguishing visual features
     * @param image a BufferedImage object used to create a new mat with the same dimensions
     * @param mat a Mat object which is used to duplicate an image
     * @return Returns a grey scaled mat object
     */
    public static Mat greyScale(BufferedImage image, Mat mat){

        Mat mat2 = new Mat(image.getHeight(),image.getWidth(),CvType.CV_8UC1);
        Imgproc.cvtColor(mat, mat2, Imgproc.COLOR_RGB2GRAY);

        return mat2;
    }
```

Launch the project in Eclipse > Locate the line above and replace it with the location of the .dll on your system

2. Understand your software

2.1. OpenCV

2.1.1. *What is it?*

OpenCV is the open source software which was selected and provides the framework utilized by the Smart Image Identifier software.

2.1.2. *Why was it used?*

Just one of the things many things OpenCV is capable of is that it provides libraries and functionality to assist in the detection of humans within images.

2.2. Classes

2.2.1. *PeopleDetect*

This is the class which Smart Image Identifier was implemented.
Full Doxygen commenting is provided in this class, describing functions, their parameters and a brief description on its purpose.

2.2.2. *Writer*

This class provides us with video writing capabilities, it was used to create a copy of the video in question.

2.2.3. *FourCC*

This class provides us with a "four character code" (4CC) which is used by AVI files.
It wraps a 32-bit value to be used as a 4CC inside an AVI file, so that it is guaranteed to be valid, and it incurs no overhead if used repeatedly.

2.3. Functions

2.3.1. greyscale(BufferedImage image, Mat mat)

This method receives two parameters, image and mat. The image variable is used to create a temporary mat object with the same dimensions whereas the mat object is used to transfer the data within the image after which is grey-scaled and returned.

Grey-scaling assists in image processing by removing all the colour the image holds less data therefor becomes easier to process.

2.3.2. equalization(BufferedImage image, Mat mat)

This method receives two parameters: image and mat. The image variable is used to create a temporary mat object with the same dimensions whereas the mat object is used to transfer the data within the image which is required to be grey-scaled.

This increased the contrast within the image helping objects to appear more defined in the image.

2.3.3. enlargeImage(BufferedImage image, Mat mat)

This method receives two parameters: image and mat. The image variable is used to create a temporary mat object with the same dimensions whereas the mat object is used to transfer the data within the image.

This method enlarges an image, it helps with identification of objects as the windows of pixels processed making it easier to detect.

2.3.4. generateMat(BufferedImage image)

This method receives one parameter: image. The image variable is used to create a mat object with the same dimensions which is returned to the system and stored for processing.

2.3.5. *generateImage(BufferedImage image, Mat mat, byte[] data)*

This method receives three parameters: image, mat and byte. The image variable is used for identifying sizes, the Mat object holds the data which is to be saved into an image and data which holds additional data containing the location of the boxes which are drawn onto the image after a human is detected.

This method outputs the results of the Smart Image Identifier and saves it as a JPG file which is stored in the output folder within the project.

2.3.6. *processImage(String url)*

This method receives one parameter: url. The url is the location of an image which is required to be processed.

This method is the heart of Smart Image Identifier, all the processing required for human detection is done here.