Road sign detection for visually impaired people using OpenCV

Dushan Aluth Patabendige

A project submitted in partial fulfilment of the requirements for the Glyndŵr University award of B.Sc. (Hons) in Applied Computing undertaken within the Department of Computing, Institute for Arts, Science and Technology, Glyndŵr University, Wrexham

Acknowledgement

First of all, I would like to express my gratitude and thankful to the academic and supportive staff of Londontec and academic and support staff of Glyndŵr University, Wrexham for giving us valuable degree programme and providing us a good guidance continuously.

Special thank must go to my project supervisor Mr Rukmal for his assistance and guidance. I must thank all of the contributors of computer-vision methodologies for their great innovations and all of the maintainers of frameworks, libraries and research paper archives for their great job to make sure everyone got access for these resources.

Abstract

According to the world health organization there were around 285 million visually impaired people worldwide by 2010. Because of the impairment they are facing with many challenges in their day to day life. Especially when considering the safety of visually impaired pedestrians, they are in a high risk of accidents. This project presents a research to identify a solution to to ensure the safety of visually impaired pedestrians.

This research is based on usage of computer vision and object recognition methodologies to help visually impaired people to recognize road signs which are important for pedestrians. We compared two of major digital image processing techniques to identify the most suitable image processing technique. These are Haar cascade approach and template matching approach. To implement the Haar cascade approach, we trained a Haar classifier using collected images.

To ensure research outcomes we implemented two prototypes for each approach. For this, used the Python programming language and the OpenCV real time computer vision library. Other than software utilities, we used a webcam and an audio output device to implement prototypes.

Table of Contents

List of Figures

Glossary

- RAM – Random access memory
- CPU – Central processing unit
- DNA - Deoxyribonucleic Acid
- Framework - A set of class library which provide functionalities.

Introduction

Today the world is moving forward very fast and everything is getting automated and connected. Day by day we are getting new seamless features and facilities on every digital device which improves the quality of our life. But still there are so many areas which didn't get much attention of scientists and investors. Digital support for visually impaired people is one of the important areas which require much attention of inventors. While we are experiencing massive improvements of technologies in our day to day life, still visually impaired people are using the white cane to detect obstacles of their way, which was introduced in 1921.

As humans, every one of us have the same right to live in a safer environment. But when someone has some disability or a weakness which makes his life unsafe or harder, it is our social responsibility to help them by applying what we learnt from the college or the school. The main idea of this project was to do a research on usage of computer vision and object recognition methodologies to help visually impaired people to identify road signs which are important for pedestrians and ensure their safety when they travel.

Literature review

The white cane, which is using by visually impaired people to detect obstacles of their way has been introduced in 1921 by James Biggs. It took almost 10 years to presence in society as a tool and a symbol of the visually impaired people. Till now this is the mostly available tool within the community of visually impaired people to help them with the safety. As a tool this is useful when they need to detect obstacles, but still they require some assistance when they want to identify signages. To ensure the safety of drivers and pedestrians, governments use road signs to give signals about special things about the road setup or to give some emergency messages. The main advice given to drivers and pedestrians is to follow road signs when they drive or walk to ensure their safety.

But for visually impaired people this doesn't work, because they are having capability issues of detecting and identifying objects by them self. The main reason for this is, all the signs are visual signals, but not in other formats, like audio signal. Due to this reason, visually impaired people are in high risk of accidents, because always the safe of them is on the hand of someone else, for example: may be a driver or some other pedestrian. Therefore, we aimed to find a solution for this to ensure the safety of visually impaired pedestrians and at the same time allow them to recognize important road signs by themselves regardless the disability. Our main target was to use methodologies from computer science in order to give an accurate and a robust solution. So, we mainly focused on analysis of computer-science based approaches to implement this solution.

By 2010, there were around 285 million of visually impaired people around the globe. But being visually impaired does not mean they are totally detached from the surroundings. Still they have the sensitivity of touch, audio, smell and all the other ways except the visual sensitivity. So we wanted to find a way to allow them to recognize road signs in other way of sensitivity. Sending audio signals is a more natural way to pass information to the user. Since we can use the natural speaking language of the user to give audio signals it is more comfortable to the user and it will feel the user this solution is more user friendly. But still we have to replace the visual sensitivity of the user by a computer-vision based methodology, because all road signs are giving visual information only. Therefore, we paid our attention on two of major image processing techniques to detect road signs. These are Haar cascade approach and template matching approach. So first we can detect the road sign using any of these approaches and then we can give an audio signal to the user to inform him about the road sign.

There are many researches already done to improve the accuracy of computer-vision based object recognition. We planned to research on these and use the existing knowledge of this domain and combine it to get this solution done. The final product of this research may help visually impaired people to travel more safer and reliably when they travel by their own.

Methodology

We consider the safety of people as the most critical thing in every part of our daily routine regardless the task, job or situation. The road safety is one of the most common and critical part of everyone's life since we are always travelling. To ensure the road safety, there are road signs which are important for both drivers and pedestrians. This is the common way of transmitting road safety instructions to travelers including drivers and pedestrians in every part of the world. And all these signs are visual signs. But 285 million of people out of the world population are visually impaired by 2010 according to the world health organization. Being safe is a fundamental requirement for every one of us and we have to ensure that visually impaired people also safe when they travel by their own. For sure, they will not drive unless they got cars with a 100% autopilot feature, which never requires user input in any case including an emergency situation. But, they are travelling by their own, by buses and by foot. Especially when they travel by foot we have to ensure they can perfectly identify road signs which are important for pedestrians, for example: Pedestrian crossing sign, bus stop sign. So, we decided to find a robust way to detect road signs on behalf of them.

In order to design the solution, we gathered already existing knowledge about object recognition based on computer vision approach. And also, we gathered information about most common road signs which are important for pedestrians and we selected few of mostly engaging signs out these to implement the prototype. To prompt about the detected sign to the user, we considered few approaches, for example: prompt a voice message using the natural language, prompt a beep sound in a specific pattern related to each of road signs or make a vibration on the device in a specific pattern related to each of road signs. And finally, we decided to target the audience which do not have hearing issues but with visual impairment and decide to prompt

a voice message using the speaking natural language because our main goal is to implement a solution to detect the road sign using a computer-vision based approach.

Findings

Object recognition

There are two common stages in computer-vision based object recognition. These are detection and recognition. At the detection stage, the system identifies most matching area of the image frame. The detection can be based on the shape of the object, colour of the object, texture pattern of the object.

After detecting objects, these are going through a recognition process to recognize objects which are exactly wanted to capture by system. To recognize most matching items, the system can match with predefined templates, or can use some classifiers or neural networks. There are many approaches which can for object recognition. For example,

Haar-like feature

In 2001, Paul Viola and Michael Jones introduce with their Viola–Jones object detection framework. This is the first object detection framework which provides competitive object detection rates in real time. This is a machine learning based approach for object detection which we should train a classifier using a lot of images. For that we need two types of images as, positive images and negative images. For example, in our scenario we need positive images which contains selected road signs, and negative images which do not have road signs which we are about to detect.

This approach uses the concept which is called cascade of classifier. Which means we group image features into different stages of the classifier and apply them one by one instead applying all features at once. For example, if our application is looking for the road sign of pedestrian crossing on a given image frame, it will try to apply from stage 0 to above. If some image frame got enough features to go over all stages, that feature area will consider as the detected object area, and if the it could not find expected features after some stage that image frame will discard and no object will be detected.

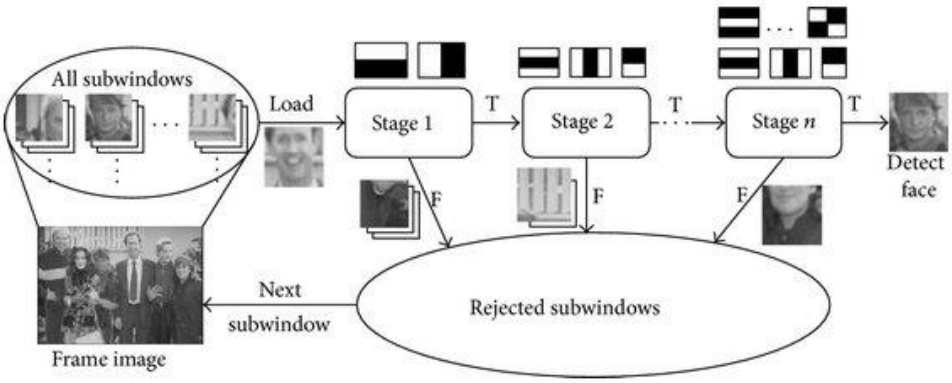

Figure 1 - Haar-cascade structure: Source: www.researchgate.net

Haar-like feature is an accurate approach comparatively, but there are many challenges in this approach also. There are mainly three challenges identified in use of Haar-like feature. The first challenge is the extent of its efficiency in the detection of objects. Due to the non-invariant nature of the normal Haar-like features, classifiers trained with this method are often unable to finding rotated objects. It is possible to use rotated positive examples during training, but such an approach may cause to inaccurate classifiers. Because of that various methods have attempted to solve this problem by introducing inclined features, by generic angles, in the learning boosting stage.

The second challenge of Haar-Like features is how to present them practically. For normal features, their presentation is straightforward and easy to achieve practically. Contrariwise, the presentation of rotated features is a big challenge because the presentation of an inclined rectangle, in an image, at an angle different to 0° degrees, may cause a distortion of its sides, which makes the determination of integral image very hard.

The third challenge is how to calculate the integral image of a rotated feature by any angle. The normal integral image is very easy to be calculated, that is done by summing the pixels values above and to the left of the given pixel. But for rotated Haar-like feature, their computation is practically very hard; this is due to the distortion of their sides caused by their rotation. So, the determination of the pixels forming these sides will be very difficult, and this will lose the Integral Image its simplicity and its quickness for which is defined by Viola & Jones.

The training of the cascade is a time consuming and a very sensitive process which requires much attention. The first step was to gather required images. This was the most sensitive part of training the cascade. Because when we select images, we have to make sure we are gathering images which has appropriate features in order to make an efficient cascade. In the middle of the research we used a set of face images as negatives and backgrounds for the positives, but it caused to train an inefficient classifier because, that face image data set contained many images from different angles of the same face. So that was a wrong attempt for us during the cascade training. So, we have to careful when collecting images and after we collect valid set of images, we have to create samples based on them.

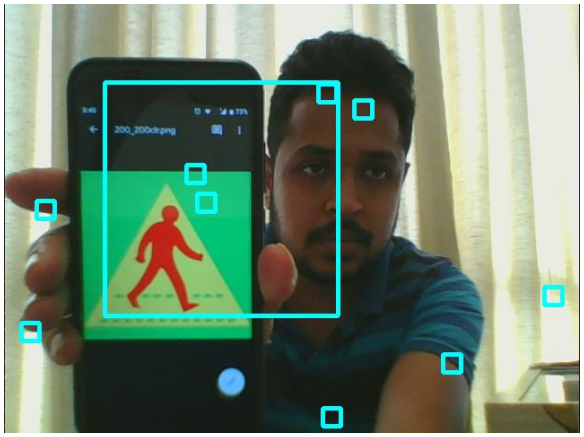

Figure 2: In-efficient classifier detects unwanted areas.

Because it is so hard to get thousands of real positive images, unless we spend dedicated time on that. So, since we are using the OpenCV framework, it allows us to have a single positive image and any number of negative images and then run a command to generate positive images by combining negative images as backgrounds and the positive image as the foreground. We generated 3216 of positive images and used following command to generate them.

```
opencv_createsamples -img 200_200clr.png -bg bg.txt -info info/info.lst -pngoutput info -maxxangle 0.5 -maxyangle 0.5 -maxzangle 0.5 -w 20 -h 20 -num 3216
```

Options used in above command are as follows.

Option	Description
-img	Source object image.
-bg	Background description file; contains a list of images which are used as a background for randomly distorted versions of the object.
-info	Description file of marked up images collection.
-pngoutput	With this option switched on opencv_createsamples tool generates a collection of PNG samples and a number of associated annotation files, instead of a single vec file.
-maxxangle	Max X rotation angle. Maximum rotation angle must be given in radians.
-maxyangle	Max Y rotation angle. Maximum rotation angle must be given in radians.
-maxzangle	Max Z rotation angle. Maximum rotation angle must be given in radians.
-w	Width (in pixels) of the output samples.
-h	Height (in pixels) of the output samples.

-num	Number of positive samples to generate.

Following is the output of above command.

```
Info file name: info/info.lst
Img file name: 200_200clr.png
Vec file name: (NULL)
BG  file name: bg.txt
Num: 3216
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 0.5
Max y angle: 0.5
Max z angle: 0.5
Show samples: FALSE
Width: 20
Height: 20
Max Scale: -1
Create test samples from single image applying distortions...
libpng warning: iCCP: known incorrect sRGB profile
```

The next step is to generate the vector file which contains all positive samples. We used following command to generate that.

```
opencv_createsamples -info info/info.lst -num 3216 -w 20 -h 20
-maxxangle 0.5 -maxyangle 0.5 -maxzangle 0.5 -vec positives.vec
```

Options used for this command are as follows.

Option	Description
-info	Description file of marked up images collection.
-num	Number of positive samples to generate.
-w	Width (in pixels) of the output samples.
-h	Height (in pixels) of the output samples.
-maxxangle	Max X rotation angle. Maximum rotation angle must be given in radians.
-maxyangle	Max Y rotation angle. Maximum rotation angle must be given in radians.
-maxzangle	Max Z rotation angle. Maximum rotation angle must be given in radians.
-vec	Name of the output file containing the positive samples for training.

Following is the output of above command.

```
Info file name: info/info.lst
Img file name: (NULL)
Vec file name: positives.vec
BG  file name: (NULL)
Num: 3216
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 0.5
Max y angle: 0.5
Max z angle: 0.5
Show samples: FALSE
Width: 20
Height: 20
Max Scale: -1
Create training samples from images collection...
Done. Created 3216 samples
```

After generating the vector file, we can proceed with cascade training. The OpenCV traincascade utility is an improvement over its predecessor in several aspects, one of them being that traincascade allows the training process to be multithreaded, which reduces the time it takes to finish the training of the classifier. This multithreaded approach is only applied during the pre-calculation step however, so the overall time to train is still quite significant, resulting in hours, days and weeks of training time. We used 2500 positive images and 1250 negative

images to train the cascade. As we found during the research the number of positive images which we should use here can get multiplying the total amount of positive images by 0.8. And also, it is recommended to maintain 2:1 ratio between positive count and negative count in order to get an accurately trained cascade. following is the command to train the cascade.

```
opencv_traincascade -data data -vec positives.vec -bg bg.txt -
numPos 2500 -numNeg 1250 -numStages 20 -w 20 -h 20 -mode ALL
```

Options used in above command are as follows.

Options	Description
-data	Where the trained classifier should be stored.
-vec	vec-file with positive samples (created by opencv_createsamples utility).
-bg	Background description file.
-numPos	Number of positive samples used in training for every classifier stage.
-numNeg	Number of negative samples used in training for every classifier stage.
-numStages	Number of cascade stages to be trained.
-w	Width of training samples (in pixels). Must have exactly the same values as used during training samples creation (opencv_createsamples utility).
-h	Height of training samples (in pixels). Must have exactly the same values as used during training samples creation (opencv_createsamples utility).
-mode	Selects the type of Haar features set used in training. BASIC use only upright features, while ALL uses the full set of upright and 45 degree rotated feature set.

Check Appendix 1 for the cascade file which we got finally at the end of the training cascade process. This cascade has been trained up to 07 stages and it was accurate enough to identify the selected road sign.

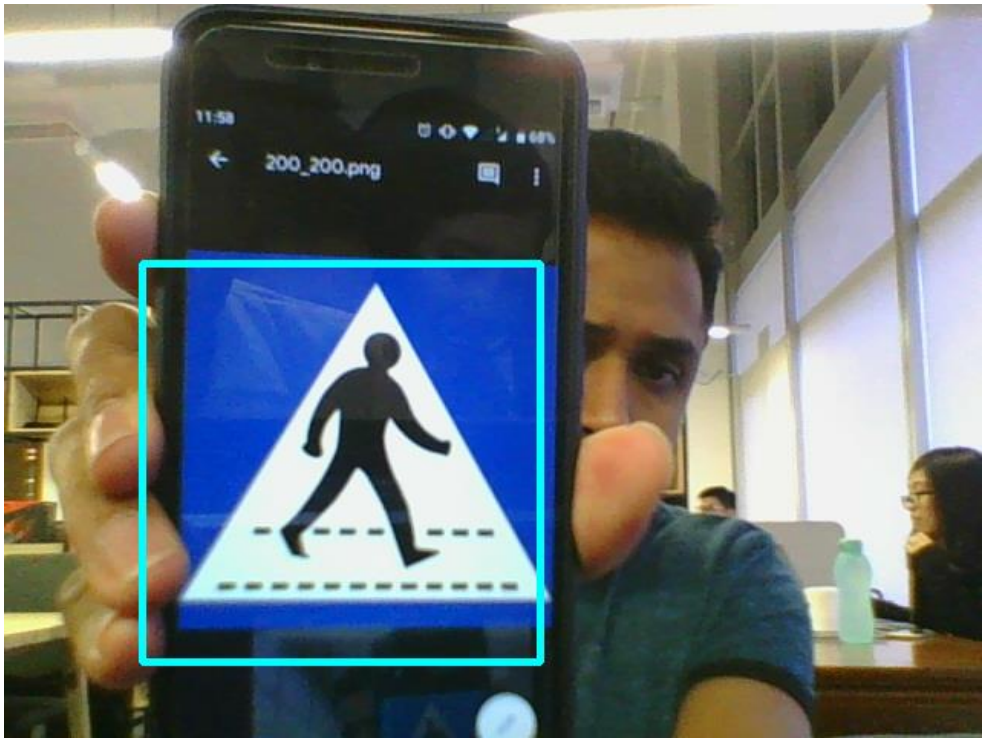

Figure 3: Well-trained classifier detected exact region with the road sign.

Template matching

This is a high-level machine vision technique which allows to recognize a part of an image which match with the given pattern. There are two approaches in template matching, as feature based approach and area-based approach. In feature-based approach mainly considered about features like points, curves or a surface model and area-based approach matching images under different conditions like illumination and color contrast. If we need to get an accurate result with template matching we have to have template images of the same object in different angles and different sizes.

There are many application areas which uses template matching. For example, in biological science and agriculture there are many applications like protein estimation in plant biochemistry, DNA analysis and colony counting. To demonstrate the template matching approach, we used OpenCV library and implemented a sample as follows.

```
import cv2
import numpy as np

cap = cv2.VideoCapture(0)
template    =    cv2.imread("cascade-train/dataset/150-bus-stop-
cropped/003.jpg", cv2.IMREAD_GRAYSCALE)
w, h = template.shape[::-1]

while (True):
   # Capture frame-by-frame
   ret, frame = cap.read()

   # Our operations on the frame come here
   gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

   res = cv2.matchTemplate(gray, template, cv2.TM_CCOEFF_NORMED)
   loc = np.where(res >= 0.6)

    for pt in zip(*loc[::-1]):
        cv2.rectangle(frame, pt, (pt[0] + w, pt[1] + h), (0,
255, 0), 3)

   # Display the resulting frame
   cv2.imshow('frame', frame)
   if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

The machine which we executed all of above had following specifications.

RAM - 15.6 GiB
CPU - Intel Core i7-6500U CPU @ 2.50GHz × 4

It is possible to train the same cascade even with less memory and CPU, but the time consumption may vary.

Analysis of Findings

In the point of view of user, the road sign recognition should be accurate as much as possible regardless the angle, lightning and vision quality. For example, sometimes the sign boards might be too tall comparatively to the user, or may be angled or turned. In some cases, there will not be enough lightning the get an accurate in detail picture, especially in evening or nights. And also, there might be weather conditions which may cause to blur the vision or maybe get some distraction.

Another point is our solution should be accurate enough to recognize the object with minimum possible delay after captured by the input device. Since the user is always moving the speed of recognition is really important. The Haar-like feature approach was more futuristic because we can train the cascade as much as we want to make it accurate, but it takes time and it costs. But the template matching approach was pretty straight forward and at the same time it was not that accurate comparatively to the Haar-like feature approach.

Evaluation of Results

When considering the research outcomes of Haar-like feature and template matching features, the implementation of Haar-like feature is time consuming while template matching is pretty straight forward. Because in Haar-like feature, we have to gather images and train the cascade and it is time consuming, but in template matching we just need source images to match. In terms of the accuracy, the Haar-like feature gives a better result than template matching. Because the Haar-like feature is a machine learning based approach. If we consider the response time of both approaches, again the Haar-like feature gives a best result. Because in Haar-like feature, it does not analyze all the features of the image at once, but stage by stage.

But in template matching approach, it may need much processing power in order to process a large image. The adaptability is really important because, world is changing in every moment. So, it is good if our selected approach can adapt to changes, for example, if the selected road sign is going to be different or some other change on it. If we use Haar-like feature, still we can train the cascate further or train a new cascade without changing the implementation. But in template matching approach we may have to do some change in our implementation because it

is not a machine learning based approach so we cannot teach it further. Following table contains a summary of evaluation.

Description	Haar-like feature	Template matching
Implementation	Takes time	Easy
Accuracy	High	Low
Response time	Low	High
Adaptability	High	Low

Discussion

In order to help with road sign recognition to visually impaired people, we wanted to find out the best approach out of computer-based object recognition approaches. Because our solution should be more reliable and accurate, since they are going to depend on our solution to ensure their safety. And our solution is going decide what it see on behalf of the user. So, after many findings we got confirmed the Haar-like feature is the best approach to implement this solution. Because it is a machine learning based approach and it is more accurate.

But to train a production level Haar-like cascade need a lot of hardware resources and time. Because it is a heavy process which trains the cascade based on our input. And also, the validity of input is really important to have a well-trained accurate Haar-like cascade. Because it is solely based on what we give, and it learns only from our input. We had to spent a lot of time on fine tuning the cascade output because, every image we give as an input matters on what we get from cascade.

In future we hope there will be much more improved hardwares to achieve such tasks within a minimum period of time. The image processing domain still requires a lot of researches on both software and hardware utilities. If we can improve these, we may get a lot of benefits from these methodologies to make our life easier and safer.

Critical evaluation of project

The main part of this project is to evaluate different digital image processing approaches and find the most suitable one to implement the proposed solution. Since our goal is to support visually impaired people to ensure their safety, there are many critical areas which we should pay our attention. The first one is the accuracy of decisions which are taking by our solution. This is really important because they are going to depend on our input based on the surrounding and decide how they should move next.

The other critical part of the solution is, since the final product mostly going to be a wearable, our software solution should be fit within a limited hardware specification. Which means if we consume a lot of processing power or memory, the hardware device will be bigger and it will add an additional cost on the hardware. So, it is important to find the best approach which requires comparatively minimum hardware power while giving accurate results.

And also, the adaptability is so important for such a solution. Because requirements of user are always increasing and the world is changing so rapidly. So, the solution must support to enhance or upgrade in future easily. To make it more adaptable our selected approach is more important, because that is the core of this solution.

Conclusion

Safety is the main concern for every one of us in every aspect. Road safety is one of the mostly engaging area and still there are high level of records for road accidents. There are different types of people with different disabilities, so it is important to ensure their safe regardless the disability. The main goal of this project was to find out the best approach to implement a solution to aid visually impaired people to recognize road signs which are important for pedestrians. So, after many findings we decided to implement the solution using Haar-like feature.

During the analysis we found different characteristics of different image processing approaches. Still we can enhance this solution in future by improving in many ways. For example, we can build this to learn about new road signs automatically by fetching information from various sources like, websites, real time captured images and image databases. And also, we can enhance this to support visually impaired people with identifying almost everything of their surrounding as the look around.

During this project we learnt so many things related to computer vision technologies. For example, available algorithms, issues of them, libraries and how to implement a computer-vision based solution.

References

[1] Auranuch Lorsakul1, Jackrit Suthakorn. *Traffic Sign Recognition Using Neural Network on OpenCV: Toward Intelligent Vehicle/Driver Assistance System*.

[2] Shuihua Wang, Chucai Yi, YingLi Tian. *Detecting and Recognizing Signage for Blind Persons to Access Unfamiliar Environments.*

[3] Prachi Gawande. (April, 2017). *Traffic Sign Detection and Recognition Using Open CV.*

[4] Ms. Sharmila, Ms. Tamizh Selvi, Dr. Sonia. (February, 2018). *REAL TIME FACE DETECTION AND ROAD SIGN DETECTION FOR BLIND PEOPLE USING OPEN CV.*

[5] Jamal S. Zraqou, Wissam M. Alkhadour and Mohammad Z. Siam. (February, 2017). *Real-Time Objects Recognition Approach for Assisting Blind People.*

[6] Pranita S.Swami, Prof. (Dr.). Pravin Futane. (April, 2018). *TRAFFIC LIGHT DETECTION SYSTEM FOR VISUALLY IMPAIRED PERSON WITH VOICE SYSTEM.*

Appendices

Appendix 1. Trained cascade file output.

```
<?xml version="1.0"?>
<opencv_storage>
<cascade>
  <stageType>BOOST</stageType>
  <featureType>HAAR</featureType>
  <height>20</height>
  <width>20</width>
  <stageParams>
    <boostType>GAB</boostType>
    <minHitRate>9.9500000476837158e-01</minHitRate>
    <maxFalseAlarm>5.0000000000000000e-01</maxFalseAlarm>
    <weightTrimRate>9.4999999999999996e-01</weightTrimRate>
    <maxDepth>1</maxDepth>
    <maxWeakCount>100</maxWeakCount></stageParams>
  <featureParams>
    <maxCatCount>0</maxCatCount>
    <featSize>1</featSize>
    <mode>ALL</mode></featureParams>
  <stageNum>9</stageNum>
  <stages>
    <!-- stage 0 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>-1.5180648863315582e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 6 6.7307338118553162e-02</internalNodes>
          <leafValues>
            -9.5716637372970581e-01 9.5110410451889038e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 8 -1.1560164391994476e-02</internalNodes>
          <leafValues>
            8.0535989999771118e-01 -9.8486816883087158e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 1 -->
```

```
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>-1.4194832742214203e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 18 -1.9156890921294689e-03</internalNodes>
          <leafValues>
            8.1864714622497559e-01 -9.7438424825668335e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 12 1.0694725811481476e-01</internalNodes>
          <leafValues>
            -6.6993618011474609e-01 8.3243590593338013e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 2 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>-2.4583482742309570e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 3 -1.4133797027170658e-02</internalNodes>
          <leafValues>
            7.5221240520477295e-01 -9.4594591856002808e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 10 5.2818845957517624e-02</internalNodes>
          <leafValues>
            -7.6280623674392700e-01 7.0011109113693237e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 3 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>-1.1986194550991058e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 17 -2.1664629457518458e-04</internalNodes>
          <leafValues>
            -9.2178773880004883e-01 7.2679507732391357e-
01</leafValues></_>
```

```
        <_>
          <internalNodes>
            0 -1 15 6.4756493084132671e-03</internalNodes>
          <leafValues>
            -6.7695057392120361e-01 8.0192577838897705e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 4 -->
    <_>
      <maxWeakCount>4</maxWeakCount>
      <stageThreshold>-2.8947681188583374e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 9 5.1011104136705399e-02</internalNodes>
          <leafValues>
            -7.5461256504058838e-01 7.7569395303726196e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 2 4.2036022059619427e-03</internalNodes>
          <leafValues>
            -9.2924195528030396e-01 6.8441039323806763e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 11 2.8728572651743889e-03</internalNodes>
          <leafValues>
            -6.8623560667037964e-01 7.5910365581512451e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 5 -4.8279548063874245e-03</internalNodes>
          <leafValues>
            5.5030679702758789e-01 -9.5275378227233887e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 5 -->
    <_>
      <maxWeakCount>2</maxWeakCount>
      <stageThreshold>-1.0691972821950912e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 1 7.0850234478712082e-03</internalNodes>
          <leafValues>
```

```
            -8.8340806961059570e-01 8.4819734096527100e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 16 2.1355065109673887e-04</internalNodes>
          <leafValues>
            7.7648836374282837e-01 -8.7612074613571167e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 6 -->
    <_>
      <maxWeakCount>3</maxWeakCount>
      <stageThreshold>-6.7695248126983643e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 4 1.5433211624622345e-01</internalNodes>
          <leafValues>
            -5.4894673824310303e-01 1.</leafValues></_>
        <_>
          <internalNodes>
            0 -1 13 7.0838481187820435e-03</internalNodes>
          <leafValues>
            -7.5729978084564209e-01 8.2037395238876343e-
01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 7 6.9031994789838791e-03</internalNodes>
          <leafValues>
            7.8149473667144775e-01 -9.4837975502014160e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 7 -->
    <_>
      <maxWeakCount>1</maxWeakCount>
      <stageThreshold>9.0450727939605713e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 14 -1.0341762099415064e-03</internalNodes>
          <leafValues>
            -9.8763251304626465e-01 9.0450727939605713e-
01</leafValues></_></weakClassifiers></_>
    <!-- stage 8 -->
    <_>
      <maxWeakCount>1</maxWeakCount>
```

```
      <stageThreshold>1.</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 0 4.6812725067138672e-01</internalNodes>
          <leafValues>
            1. -
1.</leafValues></_></weakClassifiers></_></stages>
  <features>
    <_>
      <rects>
        <_>
          0 1 14 16 -1.</_>
        <_>
          0 9 14 8 2.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          3 9 8 4 -1.</_>
        <_>
          3 11 8 2 2.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          4 4 12 2 -1.</_>
        <_>
          7 4 6 2 2.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          4 8 12 6 -1.</_>
        <_>
          8 8 4 6 3.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          4 8 16 12 -1.</_>
        <_>
          4 11 16 6 2.</_></rects>
      <tilted>0</tilted></_>
```

```
<_>
  <rects>
    <_>
      6 8 4 3 -1.</_>
    <_>
      8 8 2 3 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      6 8 10 12 -1.</_>
    <_>
      6 12 10 4 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      6 9 6 2 -1.</_>
    <_>
      6 9 3 2 2.</_></rects>
  <tilted>1</tilted></_>
<_>
  <rects>
    <_>
      7 5 5 9 -1.</_>
    <_>
      7 8 5 3 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      7 8 4 12 -1.</_>
    <_>
      7 12 4 4 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      7 11 5 9 -1.</_>
    <_>
      7 14 5 3 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
```

```
      <_>
        8 0 8 1 -1.</_>
      <_>
        10 2 4 1 2.</_></rects>
    <tilted>1</tilted></_>
  <_>
    <rects>
      <_>
        8 8 9 12 -1.</_>
      <_>
        8 11 9 6 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        9 8 3 6 -1.</_>
      <_>
        9 11 3 3 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        9 8 8 9 -1.</_>
      <_>
        9 11 8 3 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        12 2 8 4 -1.</_>
      <_>
        11 3 8 2 2.</_></rects>
    <tilted>1</tilted></_>
  <_>
    <rects>
      <_>
        13 5 3 6 -1.</_>
      <_>
        14 7 1 2 9.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        15 8 1 6 -1.</_>
```

```
        <_>
          15 11 1 3 2.</_></rects>
        <tilted>0</tilted></_>
      <_>
        <rects>
          <_>
            16 6 1 4 -1.</_>
          <_>
            16 6 1 2 2.</_></rects>
        <tilted>1</tilted></_></features></cascade>
</opencv_storage>
```