VISUAL AID FOR THE DISABLED

by Jay Kamdar

Submission date: 13-Apr-2020 11:51AM (UTC+0530)
Submission ID: 1296293789
File name: Group7_VisualAidForTheDisabled.pdf (1.76M)
Word count: 6204
Character count: 31263

VISUAL AID FOR THE DISABLED

Submitted in partial fulfillment of the requirements
of the degree of

Bachelor of Engineering

by

Jay Kamdar (60001160022)
Samruddhi Pai (60001160036)
Shaival Parikh (60001160038)

Project Guide:
Prof. Darshana Sankhe

Electronics Engineering
Dwarkadas J. Sanghvi College of Engineering
Mumbai University
2019-2020

Certificate

This is to certify that the project entitled **"Visual Aid for the Disabled"** is a bonafide work of **Jay Kamdar(60001160022), Samruddhi Pai(60001160036) and Shaival Parikh(60001160038)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor Of Engineering"** in Electronics Engineering.

Internal Guide
Prof. Darshana Sankhe

Internal Examiner

Head of Department
Prof. Prasad Joshi

External Examiner

Principal
Dr. Hari Vasudevan

Project Report Approval for B. E.

This project report entitled ***Visual Aid for the Disabled*** by Jay Kamdar, Samruddhi Pai and Shaival Parikh is approved for the degree of **Electronics Engineering**.

Examiners

1.----------

2.----------

Date:

Place: VileParle(west), Mumbai.

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Jay Kamdar (60001160022)

Samruddhi Pai (60001160036)

Shaival Parikh (60001160038)

Date:

Abstract

Blindness, or severe sight impairment, can affect people of any age. An estimate of 253 million people are visually challenged and out of those approximately 36 million people are blind. The multitude of difficulties faced by them on a day to day basis is immense. Owing to the advancement in science and technology, a lot of research has been done to find ways to improve the lifestyle for the partially-sighted and blind people. Reading and recognition devices could make smartphones, tablets and smart glasses into indispensable aids for the visually impaired.

The aim of the project is to provide assistance to visually impaired people. We aim to provide a navigation guidance for the blind and hence aim to do our bit to improve their lives. The product first receives input from a camera and processes the image and detects various objects in it. Using neural classifiers and trained data sets, we identify and classify the objects detected. We then identify which object out of the total objects is the closest to the observer in order to give the necessary feedback actions. After these processing steps the objects that are closest to the person or that might have the potential to harm them are detected and their positions are informed to the person using a speaker. This enables the person to become aware of their surroundings and thus act accordingly.

Acknowledgments

The making of project involves teamwork, vision, patience and perseverance on the part of group members. But a team can achieve a greater height only by proper guidance from faculty members and friends. Hence, we would like to express our gratitude to all those who in instrumental during the course of developing of the project.

First, we would like to thank our internal guide **Prof Darshana Sankhe** for her valuable inputs and timely guidance. She always had very practical suggestions and sound knowledge.

We would like to thank our **Laboratory-Assistants** and **Laboratory-Attendants** for the help and co-operation offered by them along every stage of the project.

Lastly, we would like to thank our head of department, **Dr. Prasad S. Joshi** for extending his support towards the completion of this project and for the timely encouragement.

Jay Kamdar (60001160022)

Samruddhi Pai (60001160036)

Shaival Parikh (60001160038)

Contents

Chapter 1

Introduction

Autonomous navigation is extremely important for people with visual impairment. While walking canes and guide dogs have been used since time immemorial, these lack the capability of covering a long range for obstacle detection. With the rapid development and advance in technology, there has been a significant progress in the use of headgears for assistance. This feat in technology has expanded and opened up new horizons for intelligent navigation capabilities. Different approaches for obstacle detection using ultrasound, laser and RADAR have already been employed. Our aim in this project is to employ the visual assistance using image processing algorithms.

In this project we first implement an object classification algorithm which helps us identify objects that can be potential threats or obstacles for the impaired. We have further used 3 approaches in order to find out which object is the closest to the observer using image processing. Instead of implementing the distance measurement using sensors we have decided to use image processing since it promises more accurate results, owing to the costs of sensors. On the hardware side, the project is implemented on a Raspberry Pi connected locally to a laptop via the Wi-Fi protocol.

Since the classification and depth perception of the image requires a lot computational processing, it has necessitated the use of a laptop for processing within a local network.

The objectives of the project can be divided as follows:

- Obstacle detection and classification
- Localization and perception of objects
- Audio output of feedback action

Chapter 2

Review of Literature

2.1 Samartha Koharwal, Samer Bani Awwad, Aparna Vyakaranam "Navigation System for Blind - Third Eye" International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-5 March, 2019

2.1.1Description:

This paper proposes an electronic guidance embedded vision system to support the visually impaired. IR sensor, solar sensor and camera are used in the system. A microcontroller processes the reflected signals from all devices in order to classify the obstacle. The system is portable so that it can be attached to a walking stick.

2.1.2 Result:

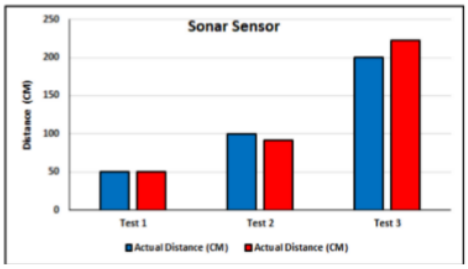


Fig 2.1.1

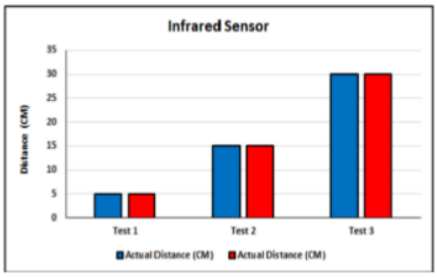


Fig 2.1.2

The above result is the output of the sensors when compared with the actual distances of the test cases in consideration. The results are quite accurate and efficient

.

2.1.3 Conclusion:

This AI based system offers a simple electronic guidance embedded vision system which is configurable and efficient. The system helps blind and visually impaired people to be highly self-dependent by assisting their mobility regardless of where they are; outdoors or indoors

2.1.4Disadvantages:

The range of prototype sensors is not high. Object detection algorithm utilizes 100% CPU and thus, in future system, two Raspberry Pi are recommended to be used; one for object detection and one for all the sensors.

2.2 Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi "You Only Look Once: Unified, Real-Time Object Detection" University of Washington, Allen Institute for AI, Facebook AI Research

2.2.1 Description:

This technical paper compared different object detection algorithms with You Only Live Once(YOLO) based on accuracy, precision, speed.

2.2.2 Result:

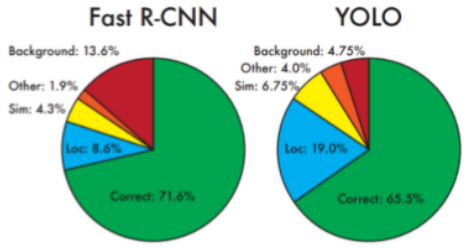


Figure 4: Error Analysis: Fast R-CNN vs. YOLO These charts show the percentage of localization and background errors in the top N detections for various categories (N = # objects in that category).

Fig 2.2.1

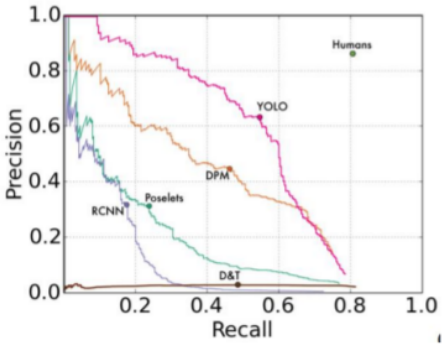


Fig 2.2.2

2.2.3 Conclusion:

YOLO is introduced as a unified model for object detection. This model can be trained directly on full images. Unlike classifier-based approaches, YOLO is trained on a loss function that directly corresponds to detection performance and the entire model is trained simultaneously. Fast YOLO is the fastest general-purpose object detector in the literature and YOLO pushes the state-of-the-art in real-time object detection. YOLO also generalizes well to new domains making it ideal for applications that rely on fast, robust object detection. Owing to the fast detection by the YOLO algorithm we have used the same in our implementation.

2.3 Finding distance from camera to object/marker using Python and OpenCV by Adrian Rosebrock on January 19, 2015 in Image Processing, Tutorials

2.3.1 Description:

This online blog tries to determine the distance of an object from the camera, using OpenCV in Python. It uses trigonometry and properties of similar triangles to calculate the distance of the object from the camera while also taking into consideration the focal length of the camera lens being used.

2.3.2 Results:

Fig 2.3.1

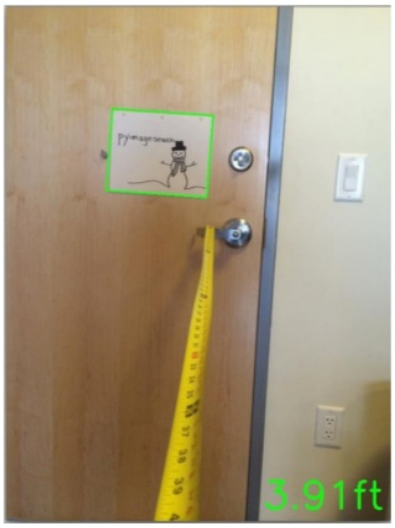


Fig 2.3.2

2.3.3 Conclusion:

Thus, upon testing the algorithm, we were able to find the distance between the camera and the fixed object. The algorithm provides results with sufficient accuracy.

2.3.4 Disadvantage:

The problem with this approach is that the height of the object from the ground should be known and the both the height of the object and the height of the camera should not change makes it less robust and prone to a lot of errors in calculating the exact distance.

Chapter 3

System Model/Architecture

The system we aim to implement is a blind and vision assistant. The functioning of the system is divided into several parts and hence a large number of algorithms from Machine Learning to computer vision and other optimization algorithms are employed. Our first aim is to implement an object detection system by using image processing. Object detection is done by employing Convolutional Neural Networks and other deep learning algorithms like YOLO. On detecting the object, the next task of the system is to differentiate between a moving and a non-moving object, living and non-living thing, etc. We use a predefined dataset called as COCO which has 80 labels from person, dog, bus to lamppost, toothbrush and cell phone. Such an extensive dataset helps us to classify and recognize the objects with greater precision and efficiency. YOLO is a state-of-the-art object detection algorithm that is incredibly fast and accurate

3.1 Convolutional Neural Networks:

In neural networks, Convolutional neural network is one of the main categories to do images recognition, images classifications. CNNs are general used to extracts features from the image. Problems like face recognition and object detection can be performed using neural networks but the number of parameters used and the processing speed required is tremendously large. These drawbacks are tackled by using convolutional neural networks.

1

CNN image classifications takes an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see h x w x d (h = Height, w = Width, d = Dimension). Eg., An image of 6 x 6 x 3 array of matrix of RGB (3 refers to RGB values) and an image of 4 x 4 x 1 array of matrix of grayscale image.

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

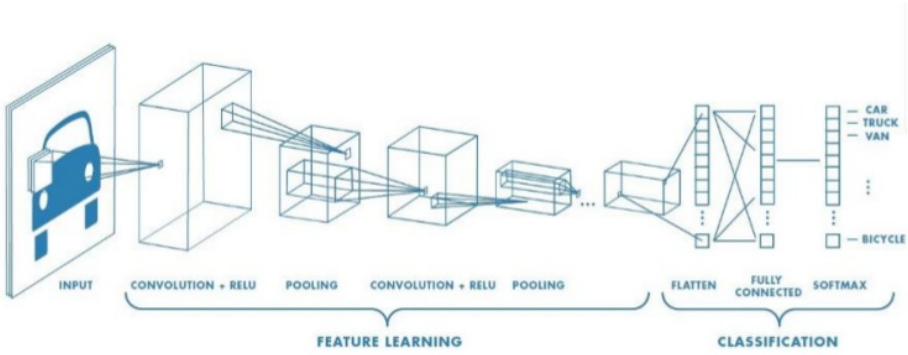


Fig 3.1

3.2 Convolution Layer:

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel. Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters. The below example shows various convolution image after applying different types of filters.

There are two types of CNN:

1. Valid CNN – Dimensions of the output matrix is less than the input matrix.
2. Same CNN – Padding of input matrix is done to keep the dimensions of input and output same.

The first step of the process is the detection of objects in the surrounding area. This is an essential part of the process as the detection and classification of objects will be useful in making decisions for the user. For this, we have employed the "YOLO" (You Only Look Once) algorithm. As the name suggests, the algorithm requires to "see" the image only once and goes through the network once and detects objects, thus making it extremely fast. The algorithm uses deep learning and Convolutional Neural Networks (CNN) for object detection.

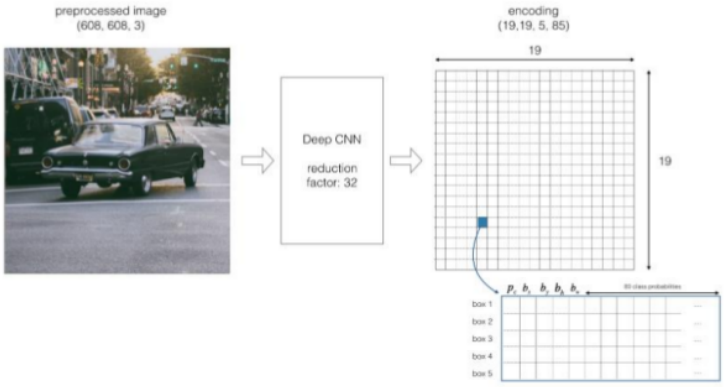


Fig 3.2

3.3 Working of YOLO:

1. The algorithm first divides the image into a grid of SxS blocks.
2. Each block predicts N possible objects using image classification and localization and places a bounding box on it, calculates the level of certainty(confidence) of each box. Thus, S x S x N boxes are created and calculated.
3. The majority of such boxes will have a very low level of certainty; thus, the algorithm deletes those boxes whose certainty level is lower than the set threshold level.
4. The remaining boxes are then passed through a "non-max suppression" that deletes objects that have been detected more than once (duplicate objects) and leaves only the most exact one of them.

For object detection, we used the MS COCO pre-trained dataset. This dataset was trained using the DarkNet code base and can classify up to 80 different objects from dogs and cats to balls, laptops, cars, etc. Such an extensive dataset helps us to classify and recognize the objects with greater precision and efficiency.

3.3.1 YOLO Architecture:

The detection layer consists of 24 convolutional layers followed by 2 fully connected layers. Alternate 1x1 convolutional layers reduce the features space from preceding layers. The final output of the network is a 7x7x30 tensor of predictions.

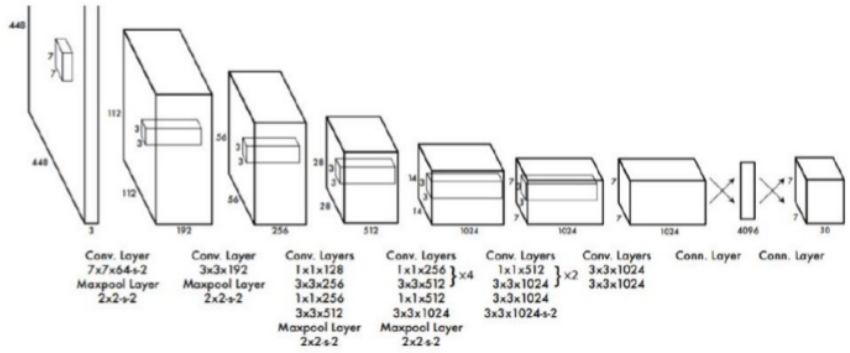
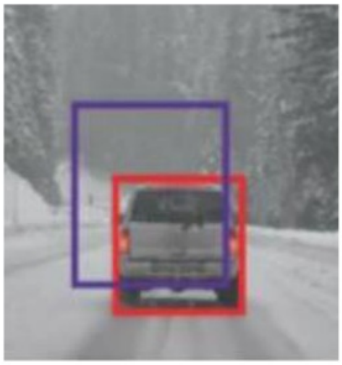


Fig 3.3

3.3.2 Intersection Over Union:

Intersection over Union calculates the ratio of the intersecting area of the actual bounding box and the predicted bounding box for an object to the union of the areas of the bounding boxes.

Consider the actual bounding box(red) and the predicted bounding box(blue) for the car shown below. The intersection over union will be calculated as:



$$IoU = \frac{Area\ of\ intersection}{Area\ of\ union}$$

Fig 3.4

If IoU is greater than 0.5, the prediction is good enough. The value 0.5 is taken arbitrarily and can be changed according to the requirement. The higher the value of the threshold, the better the predictions become.

3.3.3 Non-max suppression:

This algorithm is used to remove the multiple bounding boxes formed around the same object. First, the boxes with confidences less than the threshold set, are removed. Out of the remaining boxes, the one with the highest confidence value is selected. The IoU value for all the remaining boxes for the same object is then calculated. If the IoU value for any such box is greater than the pre decided threshold value, such boxes are then suppressed. These steps are then repeated, until all the detected objects are bound by only a single box.

Fig 3.5

3.3.4 Prediction Vector:

The input image is divided into SxS grid cells. Each grid cell predicts B bounding boxes and C class probabilities. There are 5 components in the bounding box prediction: {x, y, w, h, confidence}. Here, (x,y) represents the coordinates of the center pixel of the object with respect to the grid cell location. These coordinates have been normalized to [0,1]. (w,h) provide the width and height, i.e. the dimensions of the box which have also been normalized to [0,1] relative to the image size. The confidence prediction is defined as the IoU between the predicted box and the ground truth box. If no object exists in a cell, the confidence is zero.

Each grid vector also predicts C conditional class probabilities. Regardless of the number of bounding boxes B, only one set of class probabilities per grid cell is calculated. During the test time, the conditional class probabilities and the individual box predictions are multiplied to

produce a class-specific confidence values for each box. These values define the probability of the class appearing in the box as well as how well the predicted bounding box fits the object.

$$\Pr(Class\ i|Object) \times \Pr(Object) \times IoU(pred, truth) = \Pr(Class\ i) \times IoU(pred, truth)$$

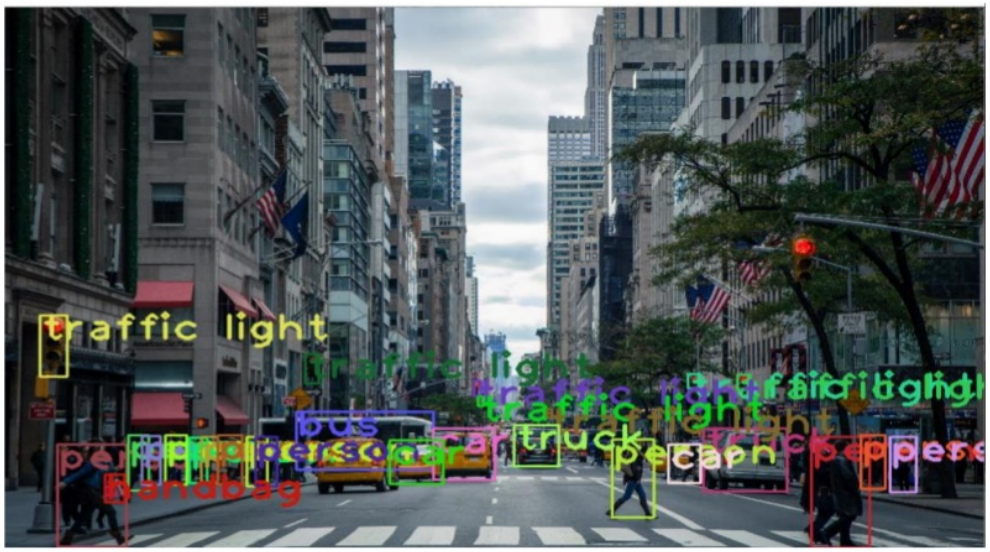


Fig 3.6

3.4 YOLO vs Tiny YOLO:

The YOLO algorithm requires large processing power and more memory. The main reason for this is large model size and less inference speed. This makes it unsuitable for its use on embedded devices like the Raspberry Pi. Thus, to overcome this, we used Tiny YOLO algorithm.

Tiny YOLO is a smaller version of the YOLOv2 model and is made up of 9 convolutional layers and 6 max-pooling layers. The model accepts the input image of the shape 416x416x3. The output produced is a 125x13x13 tensor with the image being divided into 13x13 grid cells. Each grid corresponds to 125 channels, made from 5 bounding boxes that are predicted by the grid cell and the 25 data elements that describe each bounding box.

The major advantage of this model are:

1. It has a smaller model size, thus requires lesser memory.

2. Instead of using TensorFlow on the backend, it uses TensorFlow Lite, which has been developed to enable on -device machine learning inference with a small binary size and low latency.

Neural Network	Input Resolution	Iterations	Avg loss	average IoU(%)	mAP (%)
Tiny Yolo V3	416x416	42000	0.1983	45.59%	61.18%
Tiny Yolo V3	608x608	21700	0.3469	46.39%	61.29%
Tiny Yolo V3	832x832	55200	0.2311	48.69%	56.77%
Yolo V3	416x416	19800	0.1945	0.00%	0.00%
Yolo V3	608x608	2900	0.71	42.63%	23.46%
Yolo V3	832x832	5600	0.3324	38.77%	41.20%

Fig 3.7

Due to these factors Tiny YOLO becomes perfect for use on the Raspberry Pi, it provides with faster processing and higher frame rate.

The next phase of the project included the calculation of distance of identified objects from the user. For calculating distance through image processing, we found out 3 approaches namely,

1. Distance measurement using similarity of triangles
2. Point Triangulation Method
3. Stereo Reconstruction using Disparity maps

Distance measurement using similarity of triangles method using only one camera. It is based on the geometric principle which states that the ratios of sides of similar triangles are equal. Here, the distance is measured by comparing the ratios of lengths of two triangles that are formed. This method has been explained in chapter 4 extensively. The triangulation method and stereo vision approach follow a similar set of principles and they use 2 cameras for finding the distance accurately. Since they use two cameras, camera calibration is required. Both the binocular distance measurement approaches are based on the concept of stereo vision.

3.5 Camera Calibration:

Cameras are often untuned and thus suffer from various image distortion problems. The camera that we used suffered from radial distortion and for it to be used, it had to be rectified.

This is done by using a chess board. About 50 images of the chess board are acquired from different poses of the camera. The real-world distances between each block on the chess board is known. Then by matching the points on the chess board on all the images and by computing their differences , we were able to find the intrinsic and extrinsic parameters of the camera like camera matrix, projection matrix , rotation and translation vectors, etc.

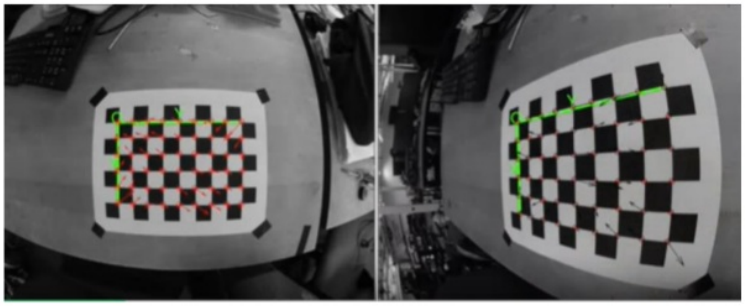


Fig 3.8 Point Matching

By finding the error level of each image of the camera, we were able to remove the problem of radial distortion and finally produce planar images. The rectified image is shown below.

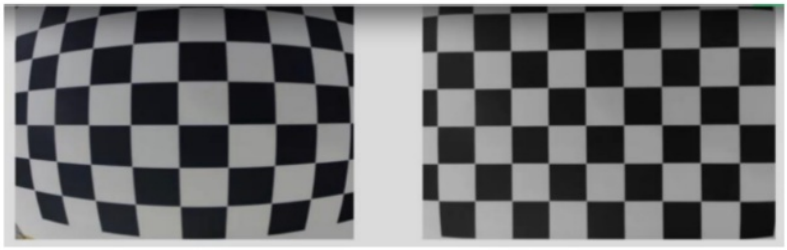


Fig 3.9 Rectified Pair

3.6 Stereo Vision:

Humans having two eyes in the front of their heads, helps them to determine the depth of the objects around them. We are able to identify, which objects are nearer to us and which are further away. The reconstruction of the 3D geometry of a scene requires at least 2 images of the scene from different views. This is defined as Stereo Vision Reconstruction.

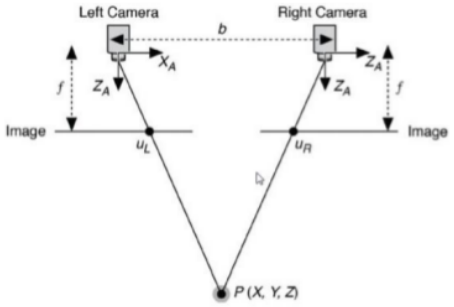


Fig 3.10

To implement this, the relative orientations of both the cameras must be known. Since the camera lenses will produce some distortions, a stereo calibration in addition to the normal camera calibration will be needed to be done.

To do this, images of the same object from different poses of the cameras are captured. Then by matching the points on various image pairs, the cameras are brought in sync with each other and the radial distortion is also removed. The rectified image produced will have images from both the cameras reconstructed on the same projection plane. This resulting geometry is called epipolar standard geometry. This reduces the computations required to transform image scenes into 3D scenes.

3.7 Disparity Maps:

Disparity maps is defined as the apparent pixel difference or motion between a pair of stereo images. Such a map can be used to make an intensity map out of the measurements, which can be used to calculate the depth of objects in an image and thus find their distance from the camera centre. Disparity maps use the concept of stereo reconstruction and epipolar geometry to produce a single map with different pixel intensities that define different depths of objects in the scene. The objects closer to the camera will have a lighter hue and the objects that are far away will have a darker hue.

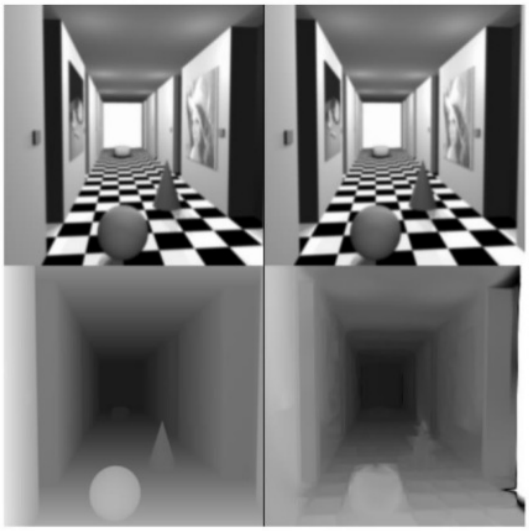


Fig 3.11

The method finds the differences in the x-coordinates on the image plane of the same feature viewed in the left and the right cameras: x_{left}-$x_{right.}$ The disparity computation can be done by using one of the 2 algorithms: Block Matching (BM) and Semi Global Block Matching (SGBM). In order to obtain the depth mapping of the scene, both the algorithms were tested upon the images to determine the best possible algorithm.

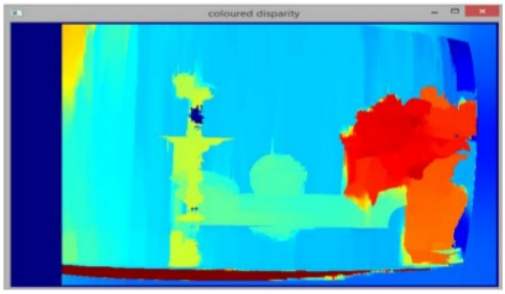


Fig 3.12 Block Matching

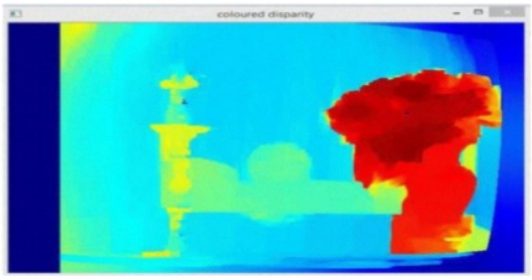


Fig 3.13 Semi Global Block Matching

The disparity intensity value obtained is then subjected to binary thresholding to define the levels of proximity. Based on these levels, certain feedback signals can be generated and then can be sent to the user. This method when paired up with the object detection and classification method, becomes quite ergonomic for the user and they are easily able to navigate in their surrounding environment.

Chapter 4

Proposed System Implementation

The basic block diagram of system implementation is shown in Fig.4.1.

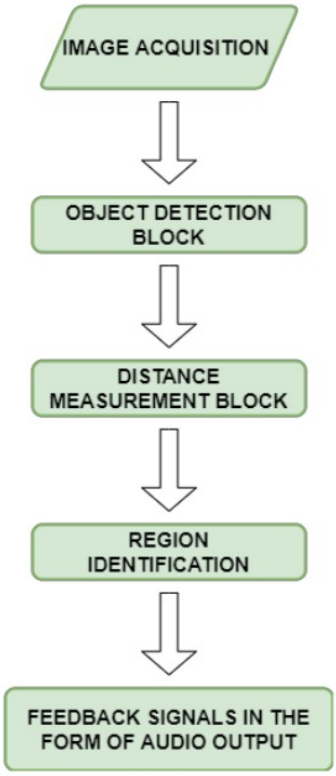


Fig.4.1

The system implementation happens in the following steps:

Step 1: Data acquisition

Data in our system is in the form of images. A human eye sees an image or object and further perceives it in the brain after the optical neuron sends the signal to the brain. A similar concept is used and implemented in a computer, the image is broken down into a matrix of pixels and stored in the memory of a computer. The matrix pixels have a value depending on the type of colour spaces used, i.e. HSV (Hue Saturation Value), RGB (Red Green Blue) and Grayscale. In our implementation we have used RGB colour space in order to process them for recognition.

Fig. 4.2

Due to constraints on the availability of large number of labelled dataset and limited computational power, we choose to use a pretrained model of object recognition based on YOLO. While pretraining the model, it was found that the best results were obtained by taking input image by using RGB color space. Since the test data should match the train data, the input that we are taking is also in RGB. The convolutional neural network kernel is slid over each individual channel to identify objects, if present in the frame.

Step 2: Object Recognition

Object detection, in our project, is done by using computer vision for identifying objects' instances in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer. Object detection uses a number of machine learning algorithms and the one which we have used is an advanced form of a neural network called as Convolutional Neural Networks which is used to implement the YOLO algorithm. Recognizing an object and identifying it is a very important step for our application since the feedback signal heavily depends on the type of obstacle in front of the human. In order to recognize the object, we need a labelled dataset which we obtain from a library called COCO. The COCO dataset has labelled data consisting of 80 distinct classes which map to a number of similar images and hence it proves to be effective in our application. Machine Learning algorithm of CNN is used to identify the object and map the features of the image to the corresponding features of the trained dataset.

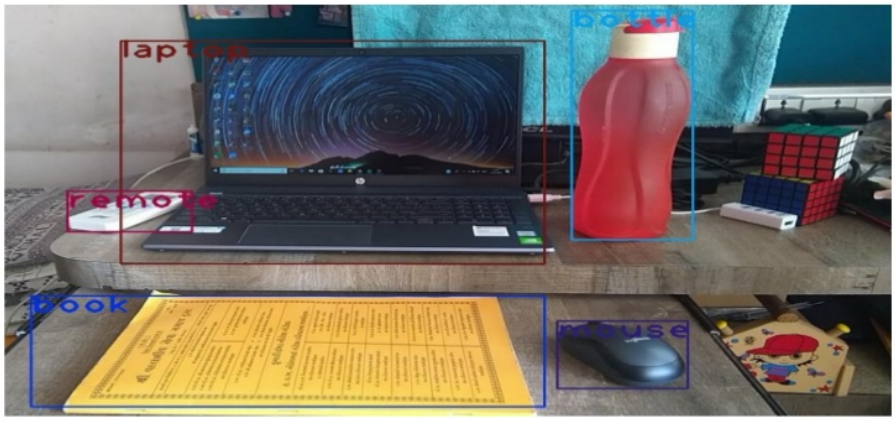


Fig. 4.3

Fig. 4.4

YOLO algorithm: The technique used in earlier detection frameworks was to look at different parts of the image multiple times at different scales and repurposed image classification technique to detect objects. Even though the accuracy of this is high, it must run the entire image classification algorithm over the same image multiple times thereby making it slower and inefficient.

YOLO (You Only Look Once) takes an entirely different approach. It looks at the entire image only once and goes through the network once and detects objects making it much faster than the previous algorithms.

CNN: In neural networks, Convolutional Neural Networks (ConvNets or CNNs) is one of the main categories to do image recognition, image classifications. Convolutional Neural Networks uses multiple CNN layers and MaxPool layers to complete the task in hand. Various implementations of Convolutional Neural Networks are Objects detections, pattern recognition, face recognition etc. CNN image classifications take an input image, process it and classify it under certain categories (e.g., Dog, Cat, Person, Car). The function of each layer of the ConvNets is to detection and extracting various features present in the input image. These features are then combined to recognize the object by mapping it to predefined classes.

Since the accuracy of the model is not 100%, there are chances of false recognition i.e. the detected object is not mapped to the right class label. To minimize this error, we calculated how confidently the model is recognizing the object. Only the objects that have confidence level

above a set threshold are further passed to create the bounding box. All the detections with confidence level below threshold are discarded at this stage. After trial and error, we found out the threshold to be 0.7 i.e. the model is 70% confident that the detected object is rightly recognized.

Step 2.1: Locating the Position of detected object:

Once all the objects are detected, the next task was to identify its height in terms of pixels and the position of object with respect to the frame. The height calculated is further used for getting the relative distance of the object and the positional coordinates are used to identify whether the object is towards right, left or in front of the user. After detecting the object, a bounding box is formed around the detected object. The height of the box gives us the height of that object in terms of pixels.

Step 3: Distance Measurement

There are 2 major approaches in finding the distance of an object from the user. The first and the most known approach is by using a sensor in order to find the distance. Sensors like Time of Flight, Infrared sensor and Ultrasonic sensor are employed in order to find the distances of an object from the source. While the processing and computation speed of these sensors can be much faster than an average processor on which image processing is being employed, these sensors always lack the consistency required to produce an accurate product. In order to obtain the distance of multiple objects, sensor fusion of all the sensor output must be used along with filters in order to remove the noise from output data. On top of the described disadvantages of the sensor approach, the cost of sensors is extremely high and hence we plan to measure the distance of the object using image processing algorithms. We are using camera for object detection purpose and using the same for calculating the distance will cut down on hardware cost.

Considering the advantages and disadvantages of other methods, we choose to implement the Pixel based distance measurement method based on similarity of triangles.

1. **Distance measurement using similarity of triangles:**

The concept of geometry namely similarity of triangles can be used to calculate the distance between the user and the detected object.

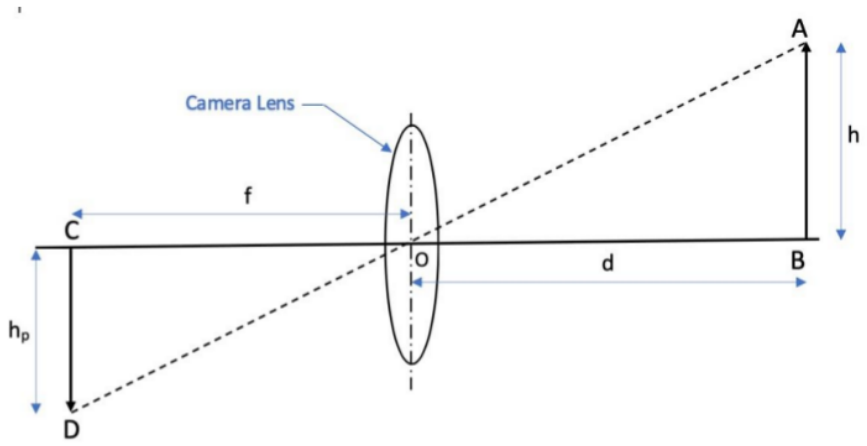


Fig. 4.5

Here, h_p = height in terms of pixels

F = focal length

d = distance of object from the user

h = actual height of the object

The two inverter triangles formed by the actual height and distance (ABO), and the focal length of lens and the height in terms of pixels (DCO), are similar to each other. Thus, by property of similarity we can say that,

$$\frac{F}{hp} = \frac{d}{h}$$

$$\therefore d = \frac{F * h}{hp}$$

In our case, we knew the focal length of camera and height in terms of pixels. One of the major drawbacks of this method that the actual height of detected object must be known in order to get the distance. To overcome this, we split our scope of objects being detected in two parts, living and non-living things. Then we created a dictionary to store the average height of those non-living things which do not vary much in height e.g. mobile phone, bus, car, etc. If the detected object is present in the predefined dictionary of heights, then its distance is calculated using the above formula. For the objects that are not present in the dictionary, only the position of such objects is given to the feedback block.

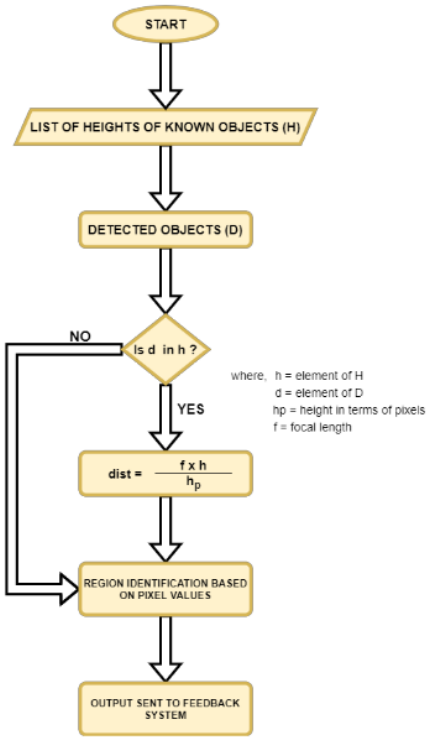


Fig 4.6

The advantage of this method is that it uses only one camera thereby cutting down upon hardware cost to great extent. Since the actual height used in the calculation is the average height, it doesn't give the exact distance of the objects, but it can accurately predict their relative distances. Using this relative distance, five closest objects are then passed to region identification block alone with other objects whose distances cannot be determined.

Step 4: Region Identification

The image is split into three parts namely right, left and center. The width of image is divided in 5 equal parts. The x-coordinate and y-coordinate of the center of detected object is used to locate the position of that object. If the center of object in consideration is in the first part i.e. in between 0 and width/5, then the object is said to be towards the right of the user. Similarly, if the center of detected object is in between pixels that range from width/4 and width, then the object is to the left of user. Otherwise, the detect is in front of the user.

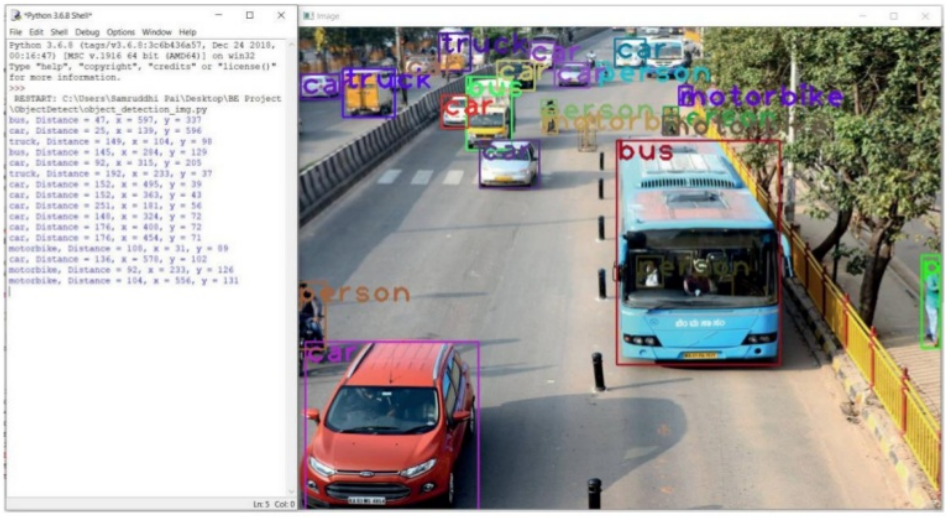


Fig. 4.7

Step 5: Feedback signal

The processor and the algorithm, based on the obstacles and the distance, returns an output which tell us whether the obstacle near or far from the observer. At the same time, it returns a vocal output to the user as a feedback mechanism. This enables the user to understand the positions of objects detected in his/her nearby vicinity. The speech output is implemented by a text to speech converter and the output is given by a voice output module e.g. speaker.

The text to speech conversion is done by using a python library. Since these libraries require supporting synthesizers, it depends on the computing platform that is being used. For implementation on windows device, we used the pyttsx3 library. The functions of this library can be accessed by creating an engine instance. After initialization, the passed string is converted into speech and the output is given by using speaker. The advantage of this library over other state of art libraries is that it doesn't require internet connection for the conversion. Considering our application, keeping the devise constantly connected with internet was not possible and hence we choose this to provide the feedback signal to the users.

After detecting objects in the frame and calculating their distances, only the objects that are in the close proximity of the user are passed on to the audio function. The speech output acts as the feedback for the user and helps him/her in perceiving the surrounding. The audio output specify which object is detected and where is it located (right, left or center) with respect to the user.

In order to implement the same on a hardware platform, we used the library named festivals since pyttsx3 could not support RPi platform. Raspberry Pi contains the “festivals” voice module package for producing voice outputs. The output device was connected to the 3.5mm Audio jack available on the Raspberry pi for the voice output.

Fig. 4.8.1

Fig 4.8.2

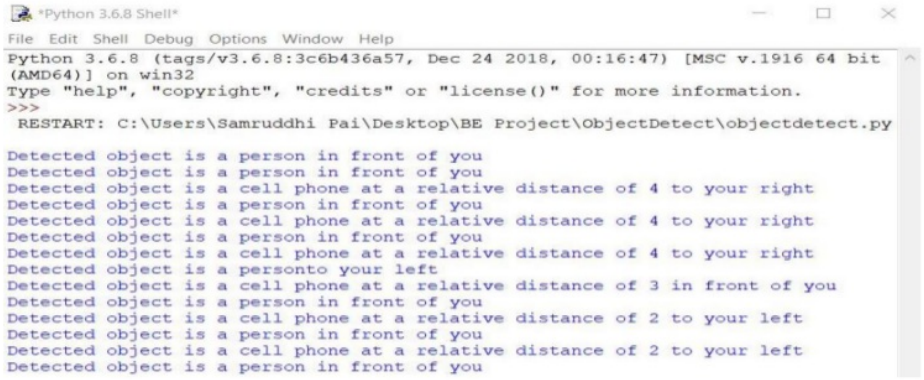


Fig 4.8.3

Step 6: Developing a scalable product

After testing the entire system on software in a high processing capability computer, we aim to make this system into a fully functional, scalable and cost-effective product in order to provide assistance to the visually disabled. The product will be used as an eyeglass by the disabled person and it will have a camera and a processor embedded into it in order to view the surroundings and process the output. It will also have an audio codec for speech output of the feedback reaction to be taken. As a part of the implementation of this future scope, we deployed a part of code on RPi and are currently working on making a finished product. The object

detection model was also implemented on a Raspberry Pi 3B+ board. The model was able to detect and classify objects and produce audio output. Here, due to the limitation of lower processing power available, a different version of the YOLO algorithms (Tiny YOLO) was implemented.

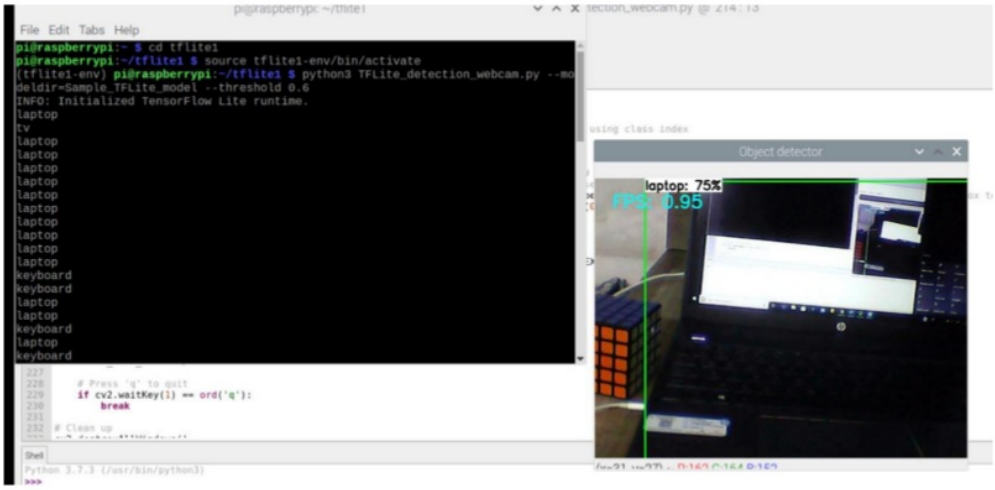


Fig 4.9

Chapter 5

Conclusion & Scope

5.1 Conclusion:

Since there is no technological and cost-effective product solution for the visually disabled, our project aims to fill the gap for this demand. We aim to serve a social cause in order to increase the efficiency and inclusion of the disabled people in the society. Through the use of various machine learning and computer vision algorithms, we have successfully implemented the vision aid project. By giving a voice output, we aim to provide a real-time feedback so that no additional help is needed in order to perceive the surroundings. By employing various algorithms and approaches for depth sensing, we get a deeper understanding in the field of depth perception and monocular vision. We further understand the drawbacks of a sensor- based approach and the possibilities of innovation in computer vision for perceiving surroundings.

5.2 Scope:

The applications of the system we have designed are immense and can be made scalable to a lot of other areas. By employing the voice output with depth sensing and object recognition, this particular product can be used by the military in order to use the headgear as night vision glasses. By adding Time of Flight sensors in the system as feedback for the computer vision output, we can make the system more robust and efficient by leveraging the advantages of sensor fusion and computer vision. A further advancement in this project can be the creation of our own synthetic dataset which can be created real time when the machine learns by the new types and features of objects it is in close vicinity to on a day to day basis. Facial recognition can further be employed so that the disabled person can recognize the people it meets on a daily basis, hence creating a feeling of familiarity.

Chapter 6

References and Bibliography

1. Samartha Koharwal, Samer Bani Awwad, Aparna Vyakaranam "Navigation System for Blind - Third Eye" International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-5 March, 2019

2. Adrian Rosebrock Find distance from camera to object/marker using Python and OpenCV.

   https://www.pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/

3. Joseph Redmon, Santosh Divvala, Ross Girshick , Ali Farhadi "You Only Look Once: Unified, Real-Time Object Detection" University of Washington, Allen Institute for AI , Facebook AI Research

4. DRISTI: Dynamic Ranging by Image Segmentation and Terrain Imaging- Samanth S Mokshagundam, Suhas G, Suhas T Shanbhogue, Suraj S

Chapter 7

Anti-Plagiarism Report

VISUAL AID FOR THE DISABLED

ORIGINALITY REPORT

10%	14%	8%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.irjet.net Internet Source	2%
2	Submitted to Engineers Australia Student Paper	2%
3	Submitted to University of Michigan-Shanghai Jiao Tong University Joint Institute Student Paper	1%
4	Submitted to Universiti Teknologi MARA Student Paper	1%
5	ijeit.com Internet Source	1%
6	es.mathworks.com Internet Source	1%
7	www.ijitee.org Internet Source	1%
8	homes.cs.washington.edu Internet Source	1%

Exclude quotes	On	Exclude matches	< 40 words
Exclude bibliography	On		