Blind Human Odometry Assistant Device Algorithm for Recognizing Cross Road Obstacles

Youssef Ashraf Edwar Amin
October 6 University, Egypt, you.ashraf.201906609@o6u.edu.eg

Supervisor: AHMED E. NEWIR, Lecturer of Mechatronics Engineering
Faculty of Engineering, October 6 University, Egypt, ahmed_newir.eng@o6u.edu.eg

Abstract–Odometry is one of the basic features for any blind assistance device to be able to navigate its location in indoor and outdoor environments. Hence, many papers try to cover this topic in many ways. This paper aims to provide three different ways to find odometry with optimum accuracy and precision. The first two methods that depend on tracking anybody by x-shape method and special color pixels by optical flow method, can interact with each other due to different conditions. The third method can work at any condition because it depends on the difference between frames and fitting between them by the machine learning system. Also, one of the needed features is to define the relation between obstacles and floor level to give the ability to avoid them without any problem from detecting the floor as an obstacle. This paper also deduced from this feature a crossroad algorithm to aid blind and impaired people to cross car roads with more safety.

Keywords-- blind assistance device; human odometry; crossroad obstacles algorithm.

I. INTRODUCTION

Scope of this research for devices working to assist blinds and impaired people. This type of research has a large variety of features that must be done to complete this a complex project. One of these features or can consider it as the basics of project is the Odometry. Odometry means that the walked distance for anybody whether vehicles or robots or that covered by the research. It's a very important issue to make the ability of localize our spatial, basically! Where am I now and what is the distance, I walked. Also, there are another basic issue that must be prepared is avoiding any obstacles can be harmful for the blind user, later, we will demonstrate this in detail.

Up to date not many authors write about odometry area. There are who tries and succeeded to make a mechanism integrated with the human leg joint to find the odometry from it, also there are who get the odometry from tracking objects and estimating the distance from him to object.

For obstacles detection, some papers made a stick provided with many sensors along it to expect any obstacles can the user faces along the way.

In this research three different techniques created to find the odometry.

The first approach is to find the odometry according to any fixed body if it was available, the second approach is to use a special color pixel (optical flow) and track them to find the odometry, and the third method is by finding the odometry from sequential frames. These three methods work with each other to avoid any change in the environment or the conditions for each method. expecting to find the odometry with the same value in all approaches and optimize the best method it must use now to avoid any errors or non-taken distance that my caused from noise or unexpected events. Hence will use this odometry data and defining obstacles with a special and effective way to apply a very sensitive feature that is helping blind user to crossroad with a high response and high accuracy decisions.

II. LITREATURE REVIEW

Some many papers and researches dealt with assisting devices or ways for blinds people. These researches have various navigational aids for blind individuals.

To finding odometry target, Usenko et al. [1] used a stereo camera to obtain the depth by using a specific calculus, then detecting edges and tracking it, this technique helps to find odometry but there are withdraw that if there aren't any landmarks or specific point this system will fail. Hu et al. [2] talked about finding odometry with a panoramic lens which is act as a stereo camera but with different calculus according to the differentiate of lens shape and the reflection of the light wave, this system really works efficient but with the indoor environment only because of the position of this lens looks up in the direction of the sky and it depends on the change in color frames so when it tests into the outdoor environment it won't work because the change of colors in the sky is hardly detected.

Fusco et al. [3] found the indoor environment odometry with depending on signs of the building like a stop sign or emergency sign and parallel starts to guide a blind user to these signs parallel with voice commands, also if there aren't any signs it finds the odometry with visual-inertial odometry that is performing with IMU sensor. IMU sensor can't depend on it to find a highly accurate result because it effects from the fast shake, which can occur with the head of user and effects from any magnetic body and the gravity of regions if there aren't exist magnetometer sensor to take feedback from it and applies the filter to correct the values. Another topic is avoiding the obstacles. There are many authors write about this basic feature with many different ways.

Kaiser et al. [4] used the same way of the navigation system with robot car which is called SLAM (simultaneously localization and mapping), so it starts to make a mapping with

a laser range finder to draw a 2D map, hence makes a path planning to avoid any obstacles, also finding odometry with a white cane attached with IMU sensor. Another approach [5] used a microwave sensor to find the obstacles with a wide range and a higher expectation and accuracy, this sensor was attached in white cane too. Bousbia-Salah [6] create a technique that using two ultrasound sensors each of them attached to the shoulder of a blind user to detect any sides obstacles, and another ultrasonic sensor at the end of the cane to detect the low-level obstacles, also finding odometry with IMU sensor and foot switch to count the steps, really this technique doesn't provide enough data to save the blind user from any obstacles at any side or any events can be occurred.
Another way used artificial intelligence [7] is by making an object recognition module to detect any trained module that can harm a blind user or shaping an obstacle for him, hence start to warn him without finding the way how to avoid it. this way has many withdraws. first, it needs a large data set to train all expecting bodies in outdoor and indoor environments, hence it requires very high-performance computing because of the huge comparisons due to the number of classes of models. Khalid et al. [8] use another way, this approach depends on face recognition to locate peoples that can hinder the blind user.

Another topic is the crossroad technique [9], this approach finds a zebra crossing on street from satellites images and check it with google map data set. it responsible for spotting zebra crossing to guide blind user it, then detecting the traffic lights to give the command to a crossroad or not.

Another assistive way is recognizing objects [10] with trained models to help blind users to recognize any object carried by them, then start to give a voice command to the blind user what is detect the object.

III. COMPONENTS

Intel® RealSense™ depth camera is used, specially D455 (Optimal range 4m :6m, FOV 58V & 87H, provided with IMU, indoor & outdoor environment), this camera recommended because it has double depth and better performance unlike another versions. Also, will need two webcams at least work with resolution (480*640), finally need a high-performance computing and high properties hardware to work in real time without any lag.

IV. PREREQUISITE

First, python programming language needed (python 2.7 or later) or any different language like C++ for example. Also, will need this library (math, numpy, threading, time). also need (pyrealsense2) library to communicate with Intel® RealSense™. Finally, OpenCV library (open-source computer vision) and YOLO V4 Platform, after installing all packages can go to the next step.

V. THE FIRST METHOD (TRACKING BODY)

This method has high priority and accurate results. So will let this method in the first. But before demonstrating it, should know its working conditions.

For this method must have a solid block or anything have the same shape to find the odometry according to it, also this solid block must be fixed by the meaning it must be static not dynamic so have a challenge how to compare between static and dynamic block. The next condition is the block must be smooth no have a curved shape for example like a wall vice versa for example a coach which have a curved shape.

A. Config Intel® RealSense™ *camera*

At the first of program must define the camera to produce color and depth frames, so will use (pyrealsense2), next will launch the class of pipeline and the Configuration class. At the Configuration class need to insert our configuration of the camera. At this paper will work with (480*640) color and depth frame resolution, with 30 FBS for each of them. Also need to extract the IMU frames from camera, by waiting motion frames classes. Generally finished the configurations of camera but at this application need a high-speed execution so will import threading library and extract the frames at a new thread away from thread of main program.

B. Scan the existence of block

Will start coding at the main program under infinity loop to make the cycle continuous. Then will check the depth of each point at the med raw if it in the range of the camera or not, this scan will give there are a wall or not. after that need to scan the block by consisting of a X shape of points, that's meaning that each point on X shape have a certain depth, by taking the difference of each point with the center point of X shape we can know the smoothness of the surface and the angle of the deflection. Can observe from Figure 1 that the center point of X shape is the med raw of an image.

Also, can give a range to this deflection by comparing each point and the center point with a critical number.

Figure 1 : x-shape each red point shows a reflection with a certain angle. can observe the reflection of wall with all directions.

C. Get the odometry

Up to date, having a straight smooth block or wall, to find the odometry the needed information is what is my old place and my new place. So will consider my new place is the first depth to me according to the average total depth

of the scanned block or wall, the new place is the new incoming depth, by subtracting them can find the change in the distance, hence, to find the rotation will track the deflection of the point of shape X but in the view of existence wall. Otherwise, can find the rotation from the IMU vice versa the linear distance IMU doesn't produce it.

D. Differentiate between dynamic & static body

This comparison is so important because in the condition of detect a dynamic block, the odometry may be result with imperfect distance or it can result with minus.

So, to beat on this problem must compare between the speed of walking user and the speed of walking body, so to achieve that need to check the speed of average distance of the X shape and the standard speed of blind walking, to deduce that if the speed of body is higher than speed of blind it produces a message error and neglect this bode until finding a new body achieve all requirements.

Figure 2: odometry from first method which gives very accurate values.

VI. THE SECOND METHOD (TRACKING SPECIAL COLOR)

This technique is the second method that when computer fails to find the first method succeeded, it automatically switches to this technique which is working with all conditions but without the same accurate of the first method, and this method can produce accuracy with error ±5cm per 1 meter. This number can consider as a less error because we are talking about a human which his motion may be half meter per step so will neglect this error which comes from the noise which may occurs while tracking colours pixels. So can demonstrate how to apply this method now.

A. Finding the special color pixels and track each point

This step is the main step for this method, finding a special color pixel to have the ability to track it, so to find them will scan all color image that come from camera but with some conditions. first will take the point which all pixels around it have a different color, but this is difficult to achieve in real life so must make an error percentage and avoid (no found points) error so will make a parameter called size, size parameter is refers to the size of special color, also will make another parameter called scan size this parameter will refer to the width of behind scanning pixels. Observe that whenever scan size increases, the tracking will be more stable, but the number of certain points will decrease.

After finding special pixels on frame, need to track this point. so, to track each point must store each pixel with its color and its position in array, after that will make a full scan on all pixels of frame trying to find the stored pixel that may be shifted from its position, or it may be constant.

This is for my own way it works very effectively but it's a little bit slow. So, we can use a ready function comes with open cv. this function makes a theory called optical flow have the same concept and faster.

Figure 3: shows the linear odometry by using second method (tracking special color)

B. Get the odometry

How to find the odometry from this data. Need at least one parameter from real life by the meaning a correct quantity from image. So have a depth frame which each pixel of it has a real distance from real life. so, the first step will get the depth of each special point collected for

one time at the beginning of finding points only, while tracking points won't need to take the depth data again.

After that will make a relation between the motion of frames along y-axis only because need the vertical motion only from frame.

Can form the relation as a linear equation calibrated with a multiplication of a specific factor.

So, the relation will be:

The_new_real_distance=((The_old_real_distance*new_postion_of_special_pixels) /old_postion_of_special_pixels) *factor

So by getting the new distance for each pixel can subtract the new real distance with the old real distance for each pixels, after that will observe a very nearly values with each other can make a curve fitting to it or we easily takes the average to all of them. Also neglecting the angle of motion pixels because can observe that if we take the old position with an angle and the new position with an angle, the angle will be neglected because it will be constant due to the angle of view.

Finally have an odometry but this operation will succeed in ideal conditions only without any problems or any noise. But in real life we expected all errors and conditions which my occur at any time surrounding us.

C. Overcoming The Errors

Many errors will be occurred when calculating the odometry.

will demonstrate it, the first error is the special point may refer to dynamic body not a fixed body along the motion .so to overcome this error must calculate the velocity of each point from old position to a new position with a certain time, then will give a range for this velocity it will be proportional with the average velocity of blind user.

The next error is the rotation motion car harm the tracking of pixels and gives a fake distance so will detect the rotation motion from IMU of camera, if there are any motion will reset and scan another new point with neglecting the fake distance that produced from rotation of the head.

The third error is the purity of lens causes a fast change in pixel color with noise like salt and paper so the result of this error may cause a large step of pixel from location to another location. so, to overcome this error without any filters will get the difference from two positions and give them a range only if it exists this range will delete this point from average calculation of real distance.

Also, must make a condition to computer for when scanning new points, it must find more than 2 points or more to give accurate values enough or it will rescan the image again.

VII. THE THIRD METHOD

This method is completely different from two previous methods, really this method can work alone without the assisting last two methods. But there are withdraws to this technique, it requires a high-performance computing to work in real time detection, this method can give a very accurate results with error ±2cm per 1 meter, also this method can be used with a human, vehicles, robots, planes because it depends on the differences between images.

Found that there is a relation between the difference of old frame and new frame with the odometry, so can deduce that the change in difference can be the odometry multiplied with factor comes from calibration operation.

As shown in Figure 4, the difference that occurs when moving the camera forward change proportional with the speed of walking. but this operation succeeds at the ideal condition, by the meaning when all bodies in frame fixed not move because if there are body moving it will result a value of odometry.

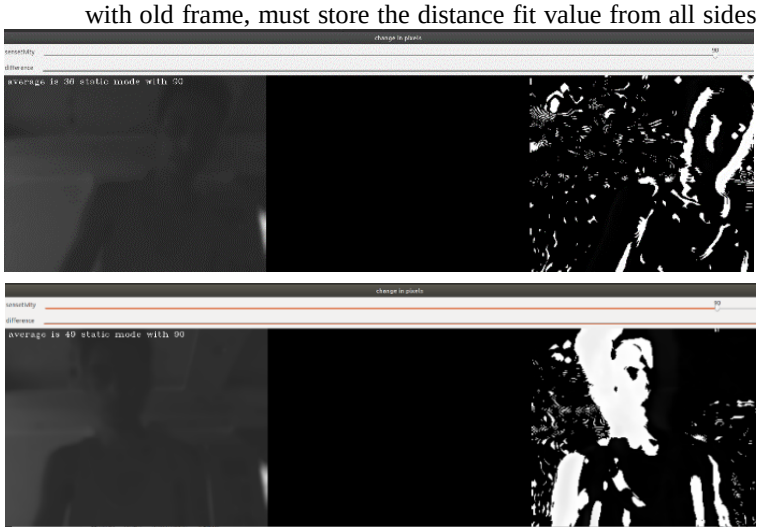
Figure 4: shows the difference (average) between frames in static mode (no change in odometry) and in active mode (change in odometry)

A. Fitting The Frames

It's very rated to face any dynamic bodies while taking the odometry or when blind user stops in his place, so need to overcome this problem. the technique used is fitting the new frame with old frame. To do that must know where the change between frames is, is the new frame zoomed in or zoomed out or became right shifted or up shifted …. Etc.

To do that will make a machine learning script to learn computer fit the new frame with old frame. that will be done by make a comparison to zoom in or zoom out the new frame and move it with the possibility's positions, with OpenCV can make a zoom in or zoom out with resize the frames and crop it so each resize will crop from different places but with respect to the ratio of original frame and the size of it (480*640).

After that must store the difference of each fit of new frame with old frame, must store the distance fit value from all sides



of frame. By obtaining the minimum difference after fitting operation, can observe from Figure 5, that when the fitting technique is applied the result will be the dynamic obstacles, which couldn't make a fit to it with cut and crop method.

Figure 5 : shows the dynamic obstacles which got it from fitting operation

Also in real life, the head of human have a rotation which may effects on the fitting because this fit will detect the change in x-axis and y-axis only, it can't detect the rotation in z-axis.

So, need to fit the rotation too. But at this point adding the comparison with rotation with respect to side distance it will be a very huge comparison and it will take a time. So will get the rotation from IMU of camera, then make a relation between the angle rotating with camera and the rotating of the image.

To do this must transfer the value 10: -10 to a value 0: 180 by make a straight-line equation. After that we will rotate the image first before any comparison with respect to rotation of camera to reach to the best fit.

B. Get The Odometry

After finding the best fit and detecting the dynamic body, it becomes easily to find the odometry by get the distance fit values from all sides with respect to minimum difference value. by make a calibration operation we can deduce the factor that should multiplied with each distance fit value to get the real moved distance, the results will be with this shape (right side=value, left side = value, upside= value downside = value). up value will refer to the rotation up, down value refers to rotation down, left and right side refer to left and right rotation. Also, the zoom out value with respect to the minimum difference will refer to the forward distance, zoom in value will refer to beyond distance. As shown in Figure 6.

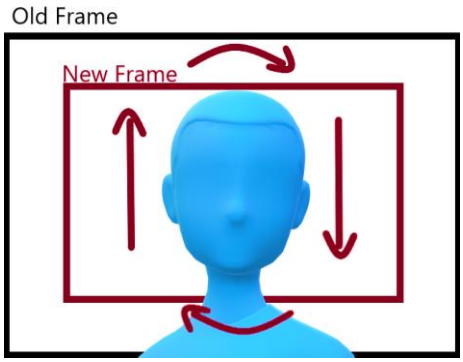


Figure 6 : the relation between direction of distance fit value and motion of the head

VIII. RECOGNIZING OBSTACLES

One of the most important things in defining or detect the obstacles are comparing very well between the floor level and sky level. This comparison is so important to prevent program from detecting the floor as an obstacle.

There is a new technique created help to compare between different levels and detecting the obstacles.

This technique depends on theoretical measuring of the depth what must be. To understand this from Figure 7,

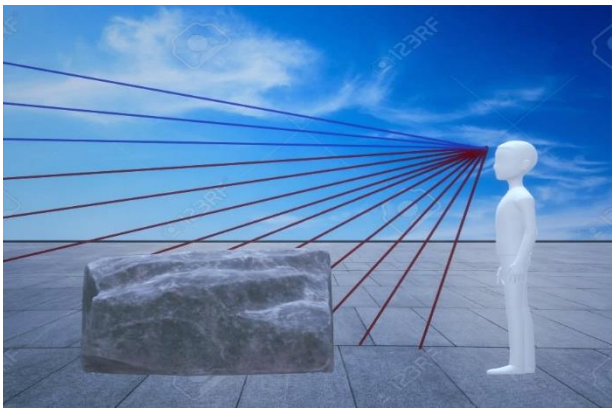


Figure 7 : Theoretical measuring of depth

To find theoretical depth, first we need to calibrate the tall of person, because it will be the only parameter that calculus will depends on it. So, to know the tall of person by rotating camera to downside until being perpendicular with the floor, can estimating the length of person from high point of his head to the point of floor.

Will take the depth average for all pixels but may exist any obstacles during calibrating operation. so, to avoid existence of any obstacles that can harm the result, will separate all depth pixels that larger than zero in array, because camera when there are a very near body to depth sensor it gives zero values of depth. so, by getting depth pixels will divide them by the number of total pixels which is (480*640) to know the percentage of available area without and near body. can compare this range with appropriate number.

After having a tall of user, will find the theoretical depth by Pythagoras algorithm and applying cos method with math library can find theoretical depth as the Figure 8 below

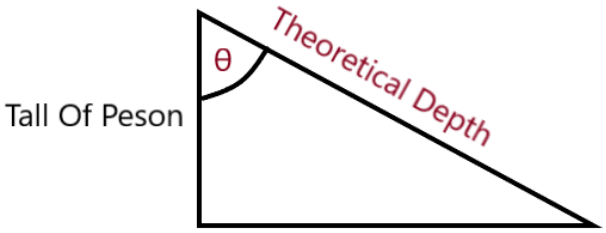


Figure 8 : Applying Pythagoras Method To Find Theoretical Depth

The limitation of the angle depends on the field of view of camera and need about 480 values of angle for each pixel along vertical line, so angle will be start from 0° to 58° with a steps equal value of FOV divided by length of frame.

$$\text{Angle} = \text{Angle} + \left(\frac{FOV}{len(color\_frame)}\right)$$

After finishing the loop, array will contain a 480 value of angle. Also, the reason for getting the depth with an individual line on frame that the reflection of the floor nearly equal with all the level according to the angle of view. From Figure 9, can observe the green line is the calculate depth pixels and the read lines are the reflection of the floor according to angle of view. So will assume that the theoretical depth along each row equal to the center theoretical depth is calculated.

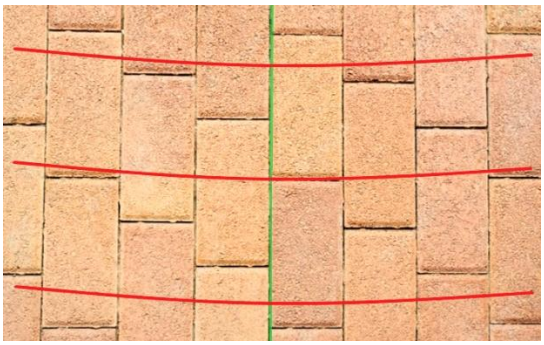


Figure 9 : studying the reflection of floor level according to angle of view

Can deduce that all theoretical depth value is the value when the camera is perpendicular with the floor by the meaning when user is looking down to floor, so need to calculate the remain theoretical depth for all possible views of the head. So, to that very easily, will increase the loop instead of range (0,480), it will be in range (0, 1440). To increase angle from 0° to 174°. This is very appropriate for the limit up motion for human head.

Secondly, after having a 1440 theoretical depth values, need to control which 480 values must be compared with the original depth image. So will take the rotation data from IMU of camera and control the angle of theoretical depth according to rotation of camera.

A. Comparing between obstacles and floor level

After getting theoretical depth with its true view, next will make an image format from this data with size (480*640). By subtracting it, get a difference depth image. can observe that when camera have a floor view the pixels of final depth image frame of floor will equal zero after making a range of error to avoid the reflection of angle of view. And the sky will equal to a value out of range camera, but the important thing here that it will differentiate very well between floor and any obstacles on the floor as expected. This occurs for the reason the original depth frame when it subtracted with the theoretical depth frame which have the expected depth of floor it gives zero value.

After comparing between floor and obstacles as shown in Figure 10 need to find obstacles real depth because the result from subtract operation doesn't give a real value of obstacles it only gives that there are obstacles or not so will get the real depth of obstacles from original depth frame.

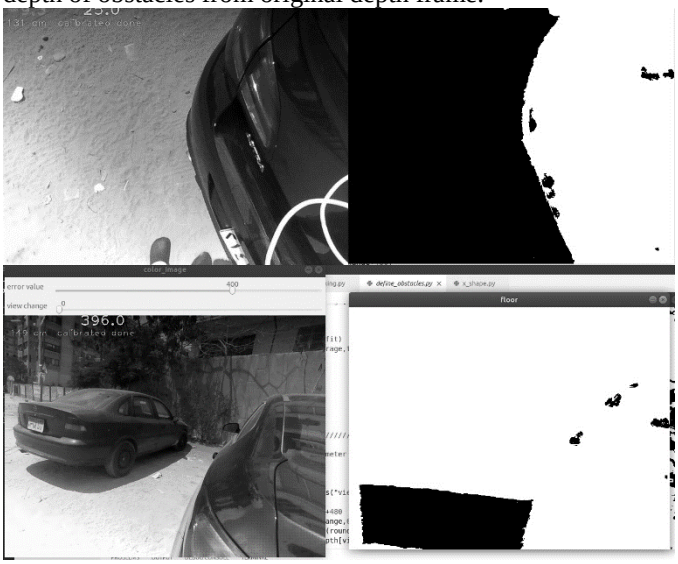


Figure 10: showing the defining obstacles technique, it differentiates very good between car and street.

IX. CROSS ROAD APPLICATION

Cross road option one of an advanced features in blind assist device that will help blind user to cross any road with a high factor of safety and high accurate of results.

First, need to know the hardware shape, Intel® RealSense™ depth camera will be at the front of view and another two web cams, one will be at the right side and the other webcam will be at the left side.

This shape is chose as Figure 11 explained, the blue views in the two webcams and the red view is the view on Intel® RealSense™ camera. Each camera will have a special job to integrate with each other to give the ability for user crossroad or not.

Will need an object recognition module to recognizing any vehicle (car, truck, bike, motorbike) all this vehicle needs to be detected. Hence, the role of two web cams will be recognizing all vehicles object. but making a processing with two webcams will make a system a little slow which is not recommended for a sensitive feature like that we need a real

time without any lag so will make a small trick to enable one of two cams which will be needed, by knowing the direction of running cars will enable the cam that have the inverse direction of the direction of running cars. By the meaning when cars go to the right side the left side camera will be enabled and vice versa. This because want to detect cars incoming cars before passes in front of blind user. But before that need to know this view is crosswalk or not before enable any webcam. it very easily to do it by power on the object recognition module on front camera which will track a car its direction and its ration area. Logical when the width larger than length car is with its side shape, that's meaning there are a crossroad, and the web cam will be enabled according to this car's direction.

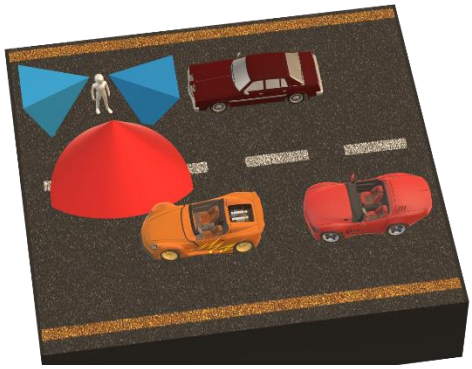


Figure 11 : view of each camera

A. *Tracking Vehicles and Calculating speed*

Tracking vehicles so important to know the new incoming bound boxes refer to which car and track the car along its motion in front of camera view, after that can determine the speed of each car without any switching of velocity with another cars.

So, to do that will track the position of each car along frames. To do that will store all bounding boxes from old frame also the new bounding boxes of the next frame, with a note that old frame updated each cycle of loop. After that will compare the distance of x-axis for each bounding boxes. By make a comparison between old and new bounding boxes will track each car by getting the least difference of distance of x-axis between boxes. After found a box that carry a difference doesn't exceed the range of error, it will consider as the car is detected, and program will give each car a tag id.

After tracking each car want to calculate the velocity of each car in frame, so will use the difference between old position in old bounding box and the new position in new bounding box for each car, but this difference according to pixels need to find this difference in real distance what will be. So will make a calibration to the area of car with a certain space, it made and found that approximately area of pixels is 160000 at two-meter distance. By getting the area of the same car with old frame and the new frame and getting the distance from Inverse relation equation.

$$distance = \frac{160000}{2 * Area}$$

B. *Algorithm Cross Road*

Will demonstrate the function of each camera what will do with steps, first the front depth camera is always focusing on defining obstacles and warn the blind user if any existence if obstacles parallel with searching the direction of car and ratio area, if a crossroad found as demonstrated previously, it starts to choose which webcam will be enabled after that webcam will start to recognize all of cars and calculate its speed. to help blind user to crossroad will calculate each car's time to reach to a safety distance far from blind user about 2-meter factor of safety, then the time that blind user will take to cross a distance equal to the width of the car that will be calculated with the odometry module or there is a sidewalk detected by defining obstacles module. the time will determined with respect to the standard velocity of blind people , will comparing it with the time of the car to reach to the point of factor of safety 2 meters , if the time of car large than time of blind user it will able to cross this road , else it will warning him not to crossing now until this car crossed and calculate the velocity of the next car to check or if found a traffic light it will have the higher priority , as shown in Figure 12 , Finally after front camera doesn't detect any crossroad , program will disable webcam .

Figure 12: tracking all road cars with a certain ID for each car and calculating the speed for each of them. it shows speed zero because all this car is parking.

X. CONCLUSION

In this paper, the first method proved a very accurate odometry with Intel® RealSenseTM camera by tracking any static body and differentiate between static and dynamic bodies, Due to the reflection of the body from the x-shape method, found the odometry from an old distance at the previous frame and current frame. The second method used an optical flow technique with Intel® RealSenseTM camera to give a real depth for each pixel and determine the odometry from the average depth of all subtracts from the previous frame and new frame. Due to working conditions for each method computer will choose automatically which is the best method from the first two methods. The third method depends on finding odometry from the difference between images, but concerning static objects only. To give the correct value of Odometry and avoid errors. Made a machine learning system

to learn to fit an old frame with a new frame. This operation gives a rotation of the head in all directions and gives the dynamic body which will be neglected from the image to find liner odometry. Differentiate between floor level and obstacles is challenging to do it. so, it found from the determination of expectations of the depth of floor level at all views of the camera, then comparing it with a real depth from Intel® RealSenseTM camera with a percent of little bit error can find from subtracting theoretical depth and real depth zero values for the pixels of floor and found obstacles with a certain value large than zero, but it's not its real depth so it only detected obstacles to find real depth can find bitwise from real depth frame.

From all this, data can find crossroad algorithms by choosing the webcam from the right and the left side will be enabled and starting to recognize and vehicle and tracking each car to know its speed. Determining the expected time for a blind user to crossroad and compare it with the time of each car to reach to safety distance of blind user parallel with defining obstacles can give a command to the blind user the availability of crossing road or not and follow its steps by founding the odometry during crossing road.

XI. ACKNOWLEDGMENT

Would like to thank Dr. AHMED E. NEWIR, Lecturer of Mechatronics Engineering for aiding and supporting me from starting till the end with all needed components and steep curve for making any paper. Also, for providing enough scientific information from his impressive experience in mechatronics fields to complete this paper. He taught me no need to start any innovation from scratch which was a killing point for me.

XII. REFERENCES

[1] V. E. J. S. J. &. C. D. Usenko, "Direct visual-inertial odometry with stereo cameras," *IEEE International Conference on Robotics and Automation (ICRA),* pp. 1885-1892, 2016, May.

[2] W. W. K. C. H. C. R. &. Y. K. Hu, "An indoor positioning framework based on panoramic visual odometry for visually impaired people," *Measurement Science and Technology,* vol. 31, no. 1, p. 014006, 2019.

[3] G. &. C. J. M. Fusco, "Indoor localization using computer vision and visual-inertial odometry," in *International conference on computers helping people with special needs (pp. 86-93).* , Springer, Cham., 2018, July.

[4] E. B. &. L. M. Kaiser, "Wearable navigation system for the visually impaired and blind people," in *In 2012 IEEE/ACIS 11th International Conference on Computer and Information Science (pp. 230-233). IEEE.*, 2012, May.

[5] E. D. M. V. M. G. R. P. D. L. A. C. A. &. C. G. Cardillo, "An electromagnetic sensor prototype to assist visually impaired and blind people in autonomous walking," *IEEE Sensors Journal,* vol. 18, no. 6, pp. 2568-2576., 2018.

[6] M. B. M. &. L. A. Bousbia-Salah, "A navigation aid for blind people," *Journal of Intelligent & Robotic Systems,* vol. 64, no. 3, pp. 387-400, 2011.

[7] A. R. S. S. S. &. K. V. Kumar, "An object detection technique for blind people in real-time using deep neural network," *Fifth International Conference on Image Information Processing (ICIIP),* vol. IEEE, pp. 292-297, 2019, November.

[8] Alkhalid, F. F., Oleiwi, B. K., & Muhsin, M. A., "Real Time Blind People Assistive System Based on OpenCV," *Journal of University of Babylon for Engineering Sciences,* vol. 28, pp. 25-33, 2020.

[9] D. M. R. C. J. M. &. M. S. Ahmetovic, "Zebra crossing spotter: Automatic population of spatial databases for increased safety of blind travelers," *In Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility,* pp. 251-258, 2015, October.

[10] H. B. F. &. A. H. Jabnoun, "Object detection and identification for blind people in video scene," *15th International Conference on Intelligent Systems Design and Applications (ISDA), IEEE,* pp. 363-367, 2015, December.