Study and Development of a Sonar Obstacle Recognition Algorithm for Outdoor Blind Navigation

Apostolos Meliones
Department of Digital Systems
School of Information and Communication Technologies
University of Piraeus, Greece
meliones@unipi.gr

Jairo Lozano Llorente
Department of Digital Systems
School of Information and Communication Technologies
University of Piraeus, Greece
jairolozanollorente@gmail.com

ABSTRACT

The paper presents an obstacle detection algorithm for a mobile application that will analyze in real time the data received by an external sonar device. Together, the smartphone application and the external device will serve as a wearable that will help the navigation and guidance of blind people. The external device will collect information using an ultrasonic sensor and a GPS module. The device has been developed as an assistance device, complementary to the cane or any other type of tool used by the blind, to be an additional safety instrument during autonomous outdoor blind navigation assisted by a smartphone application. Its main objective is to detect the existence of obstacles in the path of the user and to provide information, through oral instructions, about the distance to which it is located, its size, potential motion and to advise how it could be avoided.

CCS CONCEPTS

• Applied computing → Life and medical sciences → Health informatics • Applied computing → Life and medical sciences → Health care information systems • Computer systems organization → Embedded and cyber-physical systems → Embedded systems.

KEYWORDS

Assistive Application; Outdoor Blind Navigation; Blindness; Vision Impairment; Obstacle Detection; Sonar; Smartphone;

1 INTRODUCTION

In the last decades, there has been a great development in the field of technology. Throughout the computer science history, Moore's law validates that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress, such as processing speed or the price of electronic products, allowing to create more powerful devices in smaller form factors. Some of the most striking advances that form a turning point in recent years are the internet and the smartphone. Thanks to these two advances, many applications and devices have been developed, which, in addition to having leisure functions, also serve as a great help in our daily tasks. In fact, this technology, in many cases has made us dependent on them, since it simplifies our lives.

A few years ago, the main use of the mobile phone was making calls to other contacts. Now the phone has become a useful gadget to make purchases, manage jobs, entertain ourselves or learn. There have been a lot of developments such as wearable wristbands that measure the quality of sleep, speed at which we move or steps we take along a day, on-board navigators in cars, intelligent refrigerators that are able to order a buy when they detect that there is a shortage of some food. Quality of life can be increased thanks to technological advances.

There is a large group of people who suffer some disability and the accomplishment of daily tasks is complicated for them. Within this group, we find people with visual impairment (BVI), who are guided only by their sense of hearing and touch. According to the World Health Organization, there are currently estimated to be 285 million people affected by visual impairment, of which 39 million are blind. We are faced with the need to respond to these people and solve a problem that has been limiting their lives; insecurity on public roads [1]. The UN in 2006, in the Convention on the Rights of Persons with Disabilities (Article 24), manifests the need of a "universal accessibility", that is, that all citizens have the same opportunities, goods and services to carry out any type of activity, including the basic need to be able to go out on their own [2].

So far, the most common orientation and navigation tools that have been able to help blind people to get out safely are the dog and the cane. However, the assistance provided by these two tools for safe mobility is limited and lacking reliable information. Multiple devices have been developed to help those people navigate in outdoor and indoor places. The MANTO project develops blind escort apps on smartphones which provide a self-guided outdoor

/PETRA '19,/ June 5-7, 2019, Rhodes, Greece

ACM ISBN 978-1-4503-6232-0/19/06…$15.00
https://doi.org/10.1145/3316782.3316788

and indoor navigation service with voice instructions [3]. The MANTO project significantly extends the BlindHelper pedestrian navigation system presented in [4], which received the best innovation paper award in PETRA'16. Ref. [3] includes also a rich literature review of blind indoor navigation systems. The work in this paper presents in detail an algorithm to analyze the data obtained by an ultrasonic sensor and decide the need to warn the BVI about near field obstacles. The proposed algorithm can equip an orientation and navigation device that allows the BVI to move and be informed of the near field obstacles. Such a device will help the BVI feel more confident, as they will be able to detect obstacles via hearing before they can feel them with the cane, and they will be warned of their size, position and optimal way to avoid them.

Several sonar-based obstacle detection systems have been presented in the literature. Ref. [5] describes a microprocessor-based system replacing the Braille keyboard with speech technology and introducing a joystick for direction selection and an ultrasonic sensor for obstacle detection. Another aged approach based on a microprocessor with synthetic speech output featuring an obstacle detection system using ultrasound is presented in [6]. A sonar unit called the Smart Guide, which is a low-cost easy to use commercial product, has been developed by the Lighthouse for the Blind of Greece, a partner of the MANTO project [7]. Sonar-based obstacle perception has been proposed as well in indoor navigation applications. Ref. [8] presents an indoor navigation wearable system based on visual marker recognition and ultrasonic obstacle perception used as an audio assistance for BVIs. The EU Horizon 2020 Sound of Vision project [9] implements a non-invasive hardware and software system to assist BVIs by creating and conveying an auditory representation of the surrounding environment (indoor/outdoor) to a blind person continuously and in real time, without the need for predefined tags/sensors located in the surroundings. The main focus of the project is on design and implementation of optimum algorithms for the generation of a 3D model of the environment and for rendering the model using spatial sound signals. Ref. [10] presents a mobile wearable context-aware indoor maps and navigation system with obstacle detection and avoidance for the BVI using a depth sensor. Ref. [11] developed a sensor module that can be handled like a flashlight by a blind user and can be used for searching within the 3D environment. Inquiries concerning object characteristics, position, orientation and navigation can be sent to a connected portable computer, or to a federation of data servers providing models of the environment. Finally, these inquiries are acoustically answered over a text-to-speech engine.

The structure of the paper is as follows. The obstacle detection device of the developed outdoor blind navigation application is presented in Section 2, with a detailed description of the obstacle detection algorithm following in Section 3. Section 4 provides a classification of typical obstacles that a BVI could come across along an outdoor pedestrian travel and explain how the proposed algorithm would react in each case. The paper ends with a Conclusions Section.

2 OBSTACLE DETECTION DEVICE

The outdoor blind navigation application includes an external embedded device comprising a microcontroller, a Bluetooth module, a GPS module, a servo motor and an ultrasound sensor. The microcontroller will send the samples taken by the sensor and the GPS information via Bluetooth to the user's smartphone application. The smartphone application is responsible for making the calculations and estimations necessary to obtain the distance to which the obstacle is located, its size, and where it is a moving object. With all this data, the application will decide whether to warn the user or not. In the ensuing, we analyze briefly the components, focusing mainly on the operation of the ultrasonic sensor.

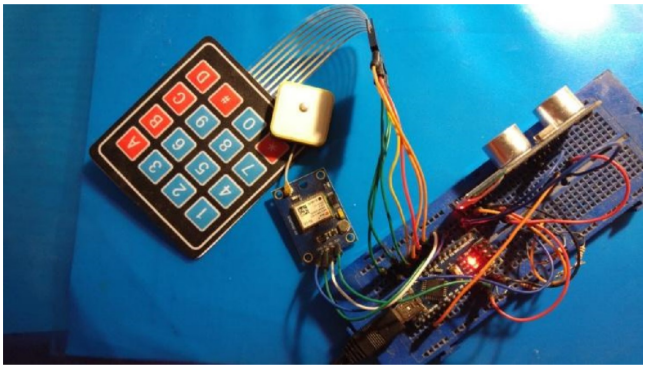

Figure 1: Prototype external embedded device of outdoor blind navigation application.

The Atmega328p uC is the cornerstone of the integrated application, since it executes the code responsible for receiving the geographical coordinates of the person in motion, the sonar operation, the use of the keyboard, the user commands, the rotation of the servo motor, as well as the sending of data from the application to the smartphone application via Bluetooth. The correct choice of the microcontroller is a definitive step towards the implementation of the system, with an impact on the total cost and future updates of the embedded application.

The Bluetooth module is interconnected with the microcontroller through a UART interface in 9600 baudrate, which promises to send the application data from the uC to the smartphone application through serial communication. It is a basic component, since as mentioned, the obstacle detection information will be collected by the sonar device and the Bluetooth module will be responsible for sending it to the application.

The external device integrates a high precision GPS receiver. It is based on the NEO-6M chip exploiting up to 16 geostationary satellites and achieving a location accuracy of 0.11m in the demanding context of pedestrian navigation for people with visual disabilities. The trials reported in [3] measured deviations, which that are crucial for pedestrian mobility, of the order of 10m between the locations reported by the smartphone integrated GPS tracker and the corresponding real geographic coordinates, while the deviations of the external GPS tracker were less than 0.4 m,

receiving signal from 11 satellites. This module provides useful information to the smartphone outdoor blind navigation application about the user's location. The app is aware of the latitude, altitude, speed, bearing, date, time and number of satellites used.

2.1 Sonar Sensor

The outdoor blind navigation application is using an ultrasonic ranging module, which is integrated in the external embedded device of the smartphone application, to determine the distance between the sensor and the closest object in its path. The sensor transmits a sound wave at a specific frequency. It then listens for that specific sound wave to bounce off at an object and come back. This sensor provides the pulse information required. This information is analyzed by the application in order to know the distance and the size of the object, to warn the blind user and to report a way to avoid it.

Figure 2: Sonar sensor.

The selected HC-SR04 sonar sensor provides a 2cm - 400cm measurement function via a 15° effective angle, and its ranging accuracy can reach 3mm. It has been selected for the development of the obstacle detection device mainly because of its low cost (less than $2), ease of use, sampling speed, and its small weight and size, which make it a good choice for wearables. It has a single control pin responsible for transmitting the ultrasonic burst. This pin should be set to high for 10 us, at which point the sonar sensor will transmit an eight-cycle sonic burst at 40kHZ (3.125us). After that, the Echo pulse output data pin, used in taking distance measurements, will be set to high. It will stay high until the ultrasonic burst is detected back, at which point it will be set to low.

The distance to the object is calculated by keeping track of how long the Echo pin stays high. The time Echo stays high is the time the burst spent traveling. The time spent until the burst finds the object is half, because the wave travels the same distance twice. As an example, consider that the Echo pin stays high for 12ms. The distance between the sonar and the object is:

$$S = V \times T \rightarrow S = 343\,\frac{m}{s} \times 6\,ms \rightarrow S = 2.058\,m$$

Another case is when the sensor does not detect any object. The maximum time that the sensor can wait for the Echo pulse is 38ms. After that time, the sensor determines there is not object in front. The ability of the sensor to detect an object depends on the object's orientation to the sensor. If an object does not present a flat surface to the sensor, then it is possible the sound wave will bounce off the object in a way that it does not return to the sensor.

3 OBSTACLE DETECTION ALGORITHM

Following the introduction of the different components of our obstacle detection device, this section explains the operation of the implemented obstacle detection algorithm. The angle of the sensor is increased in order to cover additional space. Therefore, the sensor is mounted on a servo mechanism, which will allow the sonar to rotate in such a way that it covers 15 degrees to the left and 15 degrees to the right from the global position, that is, the view angle of the sonar will be 45 degrees.

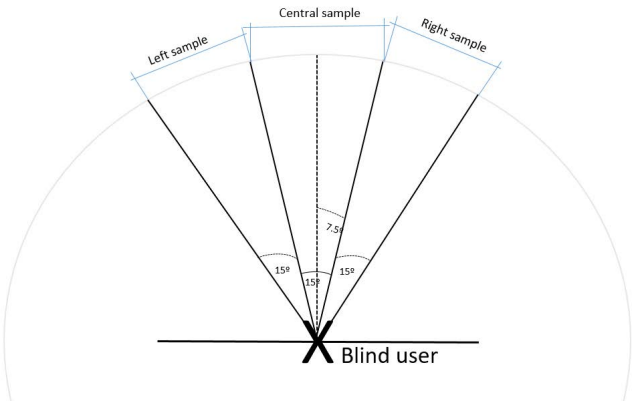

Figure 3: Angle detection of the servo-sonar system.

The sensor will take nF1 frontal samples, then it will take nL left samples, then it will take nR right samples, and finally, the sensor will take another nF2 frontal samples. All data (sensor data and map data) will be stored in an array and will be sent to the smartphone application via Bluetooth. Then the application will analyze the detection data and make the calculation in order to determine the necessary parameters. An example of a loop that the microprocessor will execute is the following:

1. Instructions for obtaining GPS data.
2. Instructions for obtaining the nF1 frontal samples.
3. Turn left the servo motor.
4. Instructions for obtaining the nL left samples.
5. Turn right the servo motor.
6. Instructions for obtaining the nR right samples.
7. Turn the servo motor to the initial position.
8. Instructions for obtaining the nF2 frontal samples.
9. Send the data to the application via Bluetooth.

The application will create a new data object each time it receives data via Bluetooth. With that object, the pertinent calculations will be carried out and, if appropriate, the object will be stored in an arrayList. The objective is to have an arrayList with objects of the same obstacle. For example, after the first reception of data, it can

be determined that the obstacle is 3 meters away. The next data object received determines the object distance at 2 meters, and the next one at 1 meter. These three objects are stored in the same arrayList, because they belong to the same obstacle. If the distance of the next data object received is bigger that the distance of the last element in the arrayList, it means that it is a new obstacle, and the previous data will be eliminated. Having the data of the same obstacle in an arrayList further helps determine whether the object is moving. In addition, consulting previous samples helps specify in a more exact way the width of the obstacle. These details will be analyzed in the ensuing.

Regarding the number of samples taken, it is a parameter that can be varied. All the samples will be in an array, and the first four elements of that array will indicate the amount of each type of sample that will be taken (front1, left side, right side and front2). An example with few samples of each type to help the understanding of the algorithm is the following:

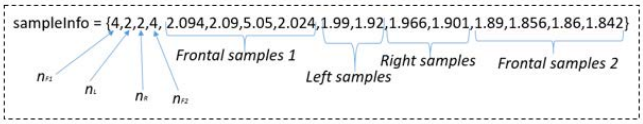

Figure 4: Sample data array structure.

Due to the parameters of the number of samples (nF1, nL, nR, nF2), the application can dynamically determine the number of samples that the microcontroller will order the sensor to take in each loop, allowing greater versatility in the amount of data that we need to obtain. This dynamic allocation will be analyzed later.

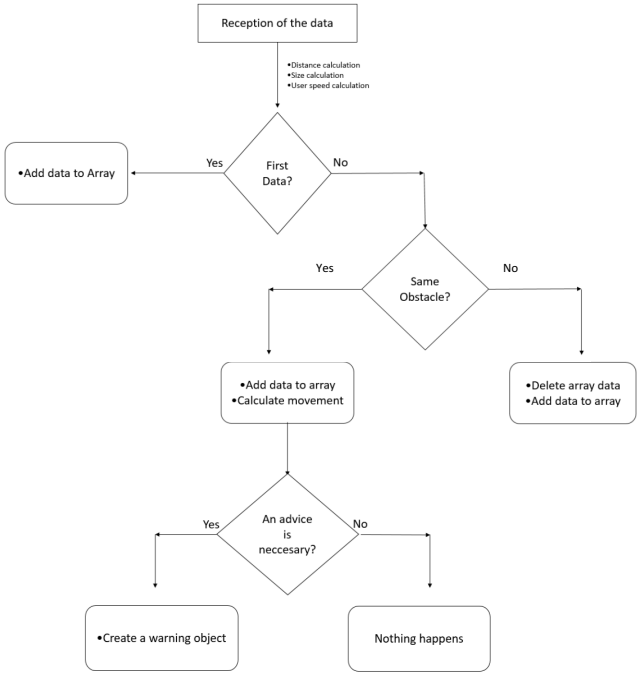

Figure 5: Flowchart of the proposed detection algorithm.

First, the sonar data will be received by the application. This data is used to calculate the user's distance to the obstacle, the size of the obstacle and the speed of the user. Subsequently, the data received is compared with the previous data stored in the data array. In that data array, there is only data referring to the same obstacle. After the comparison, it is determined whether the new detected obstacle is the same as the obstacle stored in the array. If it is not, the array data is deleted, and the new data received is added to the array. If the algorithm determines that it belongs to the same obstacle, the received data is added to the array. By comparison with the other data in the array, the movement of the obstacle is calculated, and in addition, the new number of samples for the next burst is established. The algorithm then determines if it is necessary to warn the user or not. The following paragraphs analyze how the calculations are made, starting with the distance calculation.

3.1 Obstacle Distance Calculation

The calculation of the distance is based on the correct interpretation of the front samples. The lateral samples are not used to calculate the distance the obstacle is located. These samples will be useful when calculating the size. The algorithm analyzes whether the next sample is in a close range comparing it with the previous samples and at the same time performs an averaging of the samples giving greater importance to the newest samples.

Let us consider the *sampleInfo* vector depicted in figure 4. It has 4 frontal samples. The algorithm will detect the first two samples, compare their distances and if they are similar, it will determine the distance of the object as the average between the two values. With the following samples, the algorithm will compare the distance determined previously with that of the new sample. If the new distance is within a range $-\varepsilon<0<+\varepsilon$, then the average will be made between the distance we had obtained and the new sample. Otherwise, it will be ignored. In this example, the third sample (position 6 of the array) is out of the range (5.05 meters), therefore it would not be considered for the distance calculation. The fourth sample, on the contrary, would be considered in the *finalDistance* calculation.

Subsequently, the *finalDistance* determined by the first burst of front samples is averaged with the information of the second burst of front samples in a similar way. More importance will be given to the final samples (samples that have happened less time ago) than to the older ones. If the number of samples taken increases, the information will be more reliable, however, the microcontroller loop will take more time to take the data and the transmission delay will increase. In addition, the delay of Bluetooth communication must be considered, since the user during that period is also moving. Assuming a uniform user movement, therefore (eq. 1):

$$finalDistance = finalDistance - userSpeed * delayOfComm$$

The *userSpeed* and *delayOfComm* being parameters determined thanks to the GPS module and timestamp respectively. In the example of figure 2, the calculated *finalDistance* will be 1.86475m, while for a user speed of 0.78 knots (approx. 0.4m /s) and a communication delay of 20ms, *finalDistance* = 1.85695m.

3.2 Obstacle Width Calculation

This paragraph describes the process of the obstacle angle detection and width calculation. The angle calculation is necessary to calculate the width.

3.2.1 Obstacle angle detection. First, it is necessary to distinguish if an object has been detected in the left lateral samples and if an object has been detected in the right lateral samples. This is done by comparing the obtained measure with a range around the previously calculated *finalDistance*. If it is within the range, then the algorithm increases by one a reliable sample counter of the analyzed side. At the end, if the counter is greater than half the amount of the side samples, it is determined that there is an obstacle on that side. Subsequently, the *detectionOfSample* attribute will be assigned a value depending on whether the side samples have detected an obstacle:

0 → Right and left side samples have not detected an obstacle.

1 → Just the left samples have detected an obstacle.

2 → Just the right samples have detected an obstacle.

3 → Left and right samples have detected an obstacle.

Once this is known, it is possible to interpret the angles at which the sensor pointed it has detected an obstacle. In the next paragraph, the mathematical calculation of the width of an object according to its attributes *finalDistance* and *detectionOfSamples* will be explained.

3.2.2 Obstacle width calculation. In order to explain the procedure, the simplest case is assumed, being the lateral samples not having detected any obstacle. In that case, *detectionOfSample* = 0 and *data.angle* = 15º. What happens is that the exact position in which the object is found is not known, because if the sensor covers an angle of 15º, mathematically, there are infinite points at 1.86475 meters (the example distance that we have already obtained) in which the obstacle can be found. We can see this in the figure 6.

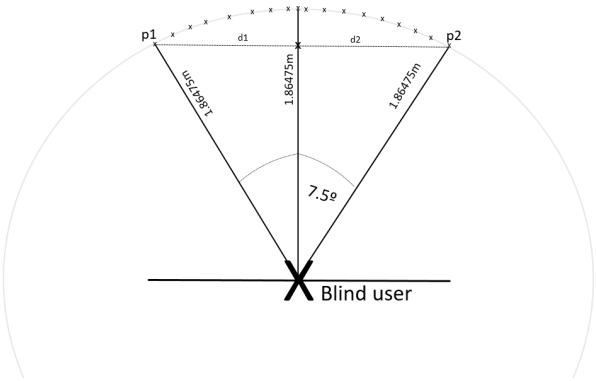

Figure 6: Different frontal positions of the obstacle.

Knowing that the arc is the one formed by an angle of 15º in a circle of radius r = 1.86475m, the distance between the leftmost point (p1) and the rightmost point (p2) can be easily calculated by applying some simple trigonometric formulas.

$$sin\left(\frac{\alpha}{2}\right) = \frac{d_1}{r} \quad (2)$$

Being α = 15º, r = 1.86475m, d_1=d_2 .

Substituting at equation 2 yields d_1=0.2434m.

Therefore, dp_1p_2 = 0.48679m.

With this data, for the time being it can be assumed that the object has a width of 0.48679m, and that to the left and right of that distance there is no object, so the user could avoid that obstacle simply by moving that distance to the left or to the right. This will be discussed in the ensuing.

The procedure if the lateral samples have detected an obstacle is similar. The only difference is that the angle that the sensor is covering (α) is not 15º. If the obstacle is detected in one lateral position, the angle is 30º. If it is detected in the two lateral positions, the angle will be 45º. Table 1 depicts possible widths for certain distances and for certain detections in lateral samples.

Table 1: Object size for different distances and angles.

An angle of 15º means the sensor has not detected anything in the lateral samples.
An angle of 30º means the sensor only has detected something in left or right.
An angle of 45º means the sensor has detected something in two laterals.

Distance, Radius(m): r	Angle(º): α	Object's size(m): s
1	15 (Just the front)	0.261052
1	30 (front + 1 lateral)	0.517638
1	45 (front + 2 lateral)	0.765366
2	15	0.522104
2	30	1.035276
2	45	1.530733
3	15	0.783157
3	30	1.552914
3	45	2.2961
4	15	1.044209
4	30	2.070552
4	45	3.061467

2*sin (α/2) = s/r

It can be observed that the estimated size which is calculated depends on the finalDistance previously calculated. Therefore, an object detected at 4 meters with 3 angles will have a size of 3.061467m. However, in the next burst, the same obstacle can be detected at 1 meter, with a size of 0.765366m. This problem is solved by storing the data of the same obstacle in the arrayList, and when the warning is elaborated, the largest size within the arrayList that is stored will be the one determined.

3.3 User Speed Calculation

The data that is received from the GPS module follows the NMEA protocol (acronym of National Marine Electronics Association),

which are standard sentences for the reception of GPS data. One of them and the most used are the $ GPRMC sentences, which have the following structure:

$GPRMC,044235.000,A,4322.0289,N,00824.5210,W,0.39,65.46,020615,,,A*44

This data frame contains the following variables:

- 044235.000 represents the time GMT (04:42:35).
- "A" is the indication that the position data is fixed and is correct. "V" would be invalid.
- 4322.0289 represents the longitude (43º 22.0289´).
- N represents the North.
- 00824.5210 represents the latitude (8º 24.5210´).
- W represents the West.
- 0.39 represents the speed in knots.
- 65.46 represents the orientation in degrees.
- 020615 represents the date (2nd of June of 2015).

Therefore, the speed in knots reported by the GPS module can be obtained directly. The external embedded device sends this information to the smartphone application. Knowing that 1 knot is equal to 0.51444 m/s, the movement of the user can be calculated easily: 0.200633 m/s. This information is stored in the *userSpeed* attribute.

3.4 Adding the Object to the Calculation

Once the distance and the width of the obstacle have been calculated, all the necessary basic parameters are obtained, and it is possible to store the data in an array structure. As it was said before, the idea is to store in an arrayList the objects belonging to the same obstacle. With the distance of the obstacle, it is possible to compare it with the distance of previously received data and determine if it is the same or a different obstacle.

What the algorithm will do first is determine the size of the data array. If it is empty, it is because it is the first data reception, therefore the new data object will simply be added to the arrayList. If it is not empty, it will obtain the final distance of the last data element of the array, i.e. the most recent data. If that distance is greater than the distance of the new element, that means that the obstacle is the same, so the data object will be added to the array. In addition, the function *SpeedObstacle* will be called. This function will calculate the speed of the obstacle based on the previous data collected (see following paragraph). If the distance is not bigger, that means that it is a new obstacle. The algorithm will then delete the components of the array and set the new data received as the first element.

3.5 Obstacle Motion Calculation

Once it is known that the new obstacle detected is the same as the previous one, the kind of motion it performs can be detected. This information can be figured out because the user speed as well as the current and previous object distance are known. The time that elapses between each data reception is taken into account through a counter called *delayBetweenFirstSamples*. Therefore, it is possible to estimate the distance at which the obstacle should be found in the current sample. The algorithm estimates the position in which the obstacle should be found taking into account the data of the previous reception. There are three different cases:

1. The real distance is shorter than the expected distance of a static obstacle (minus a toleration range), which means that the obstacle is approaching. The variable *kindOfMovement* will be assigned the value "1".
2. The real distance is larger than the expected distance of a static obstacle (plus a toleration range). The obstacle is moving away, and it is faster than the user, therefore there is no need to warn him. For this situation, the variable kindOfMovement will be assigned the value "2". In case the object moves away at a lower speed, it will be detected as a static obstacle. The difference will be that the smartphone application will collect many more data objects of that same obstacle.
3. Neither of the two previous assumptions is fulfilled, the expected distance is similar to the real one, which means that the object is static. The variable *kindOfMovement* will be assigned the value "0".

In cases 1 and 2 it is not necessary to warn the user, since an obstacle that moves away at a higher speed is not a problem and an obstacle that is approaching in 99% of cases will likely be a human person who can see the blind person and move away. Case 3 requires warning the user. The determination of the warning to the user will be explained at the next paragraph.

3.6 Warning the User

If case 3 mentioned before, the application must notify the user. However, it is not necessary to warn the user that s/he will run across an obstacle within 4 meters, since that obstacle will be detected again in the next iteration at a shorter distance. The algorithm sets the notification threshold at 3 meters. If these assumptions are met, a new object of the warning class is instantiated. The algorithm will set the distance and size used by the instantiated warning class as the distance calculated in the last component of the arrayList, and as the largest size in the arrayList respectively. To avoid the obstacle, the *detectionOfSample* variable is used, which indicates the angle at which an obstacle was detected, and through a switch instruction the cases are distinguished, and the user is notified.

3.7 Setting the New Number of Samples

It has been said that the number of samples can be variable and that the application can set it. This is explained in the ensuing. An iterator traverses the arrayList. The *detectionOfSample* parameter indicates, as explained before, the position of the detected obstacle.

In case it has been detected that the obstacle is in the frontal position and e.g. on the right, for the next sample the lateral samples on the

right would not be needed, since it is assumed that the obstacle will remain on the right. It is expected that the object will be closer and perhaps another obstacle appears to the left, therefore the microcontroller will be instructed not to take samples in that direction.

In case it is detected that there is an object on both sides, the application will tell the microcontroller not to calculate lateral samples, because only the frontal ones are of interest.

In case a new obstacle is detected, the sample number parameters are reset to the default parameters. This is done after deleting the data array.

4 TYPES OF OBSTACLES

Having explained the details of the detection algorithm, this section distinguishes the different types of typical obstacles that a BVI could come across along his route and explain how the algorithm would react. Finally, we present some types of obstacles which cannot be detected by the device.

4.1 Fixed Obstacles

These are static obstacles, which according to their exact position will require a warning to the user or not.

4.1.1 Small width obstacles. Such obstacles fit within 15 degrees of sensor view, therefore the lateral sampling does not detect anything. Such objects can be easily detected, and the user can be warned accordingly. The sensor will detect the object through the frontal samples, however the device will not detect anything with the lateral samples. The first object detection will be stored in an arrayList. The application will receive more object detections, and when the distance is less than 3 meters, the user will be warned that he can avoid the obstacle by the left or by the right.

Figure 7: Small (left) and medium (right) width objects.

4.1.2 Medium width obstacles. These obstacles are detected by the sensor in the central position and at least one of the lateral positions. In figure 7 (right), the sensor will detect that there is something in front through the frontal samples. Furthermore, it will detect that there is an obstacle in the left-lateral samples and will not detect anything on the right. In that case, the application will decide that it is not required to take more left-lateral samples, because the size of the obstacle will be calculated through the distance of the farthest object whose range cover a bigger angle. For example, if this object is detected at 4 meters, and left-lateral samples have detected an obstacle, the size will be 2.07 (according to table 1). The next object detection will be received at a shorter distance, for example 2 meters, and the size calculated would be 1.04 (according to table 1). Therefore, it is not useful to take the left-lateral samples once the previous left-lateral samples have been detected.

4.1.3 Large width obstacles. These obstacles are detected by the sensor in all positions. Both the front and side samples detect an obstacle. In that case, the indication reported to the user is that there is an obstacle of large width, which could be a wall. It is expected that this type of situation will be handled through another type of technology, as e.g. through consulting the GPS and map info or by photographic interpretation of the outdoor, before, eventually, the BVI sensing in close proximity the large object through the cane.

Figure 8: Large width objects

4.2 Not Fixed Obstacles

These are moving objects which can be distinguished depending their trajectory.

4.2.*1* Movement in the same direction.

4.2.1.1 Faster than the user. It can be a person who is walking in the same direction as the user but at a higher speed. These kinds of obstacles are not important because there is not a risk of collision. Therefore, a warning will not be sent by the application to the user.

4.2.1.2 Slower than the user. Such an obstacle will move at a fairly

slow speed and will be treated as a static obstacle. A warning will be sent by the application to the user.

4.2.2 Movement in the opposite direction. These obstacles are directed towards the user. 99% of cases will likely be a human coming close. In this case, the application is not warning the user, as the system considers that the moving obstacle is able to detect the presence of the user and avoid him/her.

4.3 Special Cases

This section describes obstacles that are not as easy to detect or are not as easy to calculate the size or the correct way to avoid them. It can be:

4.3.1 Not detected obstacles. In this frame we can find obstacles that unfortunately will not be detected by the sonar sensor, either because they do not have an adequate height or because their width is very small. Let's see some examples:

Figure 9: Not detected obstacles.

In the case of figure 9 (left), the sensor will never be able to detect that there are stairs in front because they are too low. Similarly, in the case of case of figure 9 (right), the sensor will send ultrasonic pulses that will not find an obstacle where to bounce and receive the pulse back, therefore it will determine that there are no obstacles. In both cased the sensor samples will never detect any obstacle, therefore the user will not be warned. Another obstacle that probably cannot be detected by the sonar is when in the case of figure 7 (left) the post width is very small, as e.g. the pole of a signpost (see figure 8 (left)). In that case, the sensor will send pulses which will not bounce correctly in the signpost because its width is too small. This can lead to the mistaken determination that there is no object in front.

4.3.2 Obstacles that present some difficulty to analyze. In some cases, either due to the shape of the obstacle itself or due to the street surrounding, it is impossible to advise the user in a correct way. In the case of figure 10 (left), we observe that the sensor will detect that there are obstacles in any of the angles, and it will not be able to find an escape route due to how narrow the road between the car and the wall is. In the case of figure 10 (right), the sensor will first detect that there is an obstacle in front and to the right and warn the user that to avoid it he must go to the left. However, it will end up detecting that there is an object also on the left, and it cannot find a reliable path for the user. The case of large width objects, such as in figure 8, cannot be handled effectively as well, as the application cannot decide a reliable way for the user.

Figure 10: Obstacles difficult to analyze.

5 CONCLUSIONS

The proposed system intends to address BVI guidance needs in public outdoor places. The article describes an obstacle detection algorithm for a mobile application that will analyze the data received by an external embedded device integrating a sonar module. Together, the smartphone application and the device will serve as a wearable that will help the navigation and guidance of blind people. Its main objective is to detect the existence of obstacles in the path of the user and to provide detailed information, through oral instructions about obstacles along the route of the BVI.

An assessment of the presented system in real outdoor blind navigation conditions and use case scenarios is currently ongoing being integral part of the Blind RouteVision outdoor blind navigation application. The Blind RouteVision application is under development in the framework of the Greek R&D T1RCI-00593 MANTO project.

ACKNOWLEDGMENTS

This research work is supported by the Greek RTDI State Aid Action RESEARCH-CREATE-INNOVATE of the National Operational Programme Competitiveness, Entrepreneurship and

Innovation 2014-2020 in the framework of the T1RCI-00593 MANTO project. The authors would like to specifically thank the Lighthouse for the Blind of Greece for their fruitful collaboration in the framework of the MANTO project, as well as Mr. Costas Filios, who built the prototype external embedded device hardware and software, including the Bluetooth and UART driver. The publication of this paper is partially supported by the University of Piraeus Research Centre.

REFERENCES

[1] S.P. Mariotti, D. Pascolini, Global estimates of visual impairment: 2010, Br J Ophthalmology 2012, May, 96(5):614.

[2] United Nations Organization (2006), Convention on the rights of persons with disabilities, http://www.un.org/disabilities/documents/convention/convoptprot-e.pdf.

[3] A. Meliones and D. Sampson, Blind MuseumTourer: A System for Self-Guided Tours in Museums and Blind Indoor Navigation, Technologies 2018, 6(1), 4.

[4] A. Meliones and C. Filios, BlindHelper: A Pedestrian Navigation System for Blinds and Visually Impaired. In Proceedings of the 9th ACM International Conference on Pervasive Technologies Related to Assistive Environments, Corfu, Greece, 29 June–1 July 2016.

[5] Koley, S.; Mishra, R. Voice Operated Outdoor Navigation System for Visually Impaired Persons. Int. J. Eng. Trends Technol. 2012, 3, 153–157.

[6] Bousbia-Salah, M.; Fezari, M. A Navigation Tool for Blind People. In Innovations and Advanced Techniques in Computer and Information Sciences and Engineering; Springer: Berlin/Heidelberg, Germany, 2007; pp. 333–337.

[7] Smart Guide. Lighthouse for the Blind of Greece. Available online: https://www.fte.org.gr/index.php/en/links.

[8] Simoes, W.; Lucena, V. Blind user wearable audio assistance for indoor navigation based on visual markers and ultrasonic obstacle detection. In Proceedings of the 2016 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 7–11 January 2016.

[9] Sound of Vision. Research Project Funded under: H2020-EU.3.1. - SOCIETAL CHALLENGES - Health, Demographic Change and Well-Being.

[10] Li, B.; Munoz, P.; Rong, X.; Xiao, J.; Tian, Y.; Arditi, A. ISANA: Wearable Context-Aware Indoor Assistive Navigation with Obstacle Avoidance for the Blind. In Proceedings of the European Conference on Computer Vision (ECCV 2016), Amsterdam, The Netherlands, 8–16 October 2016; pp. 448–462.

[11] Hub, A.; Diepstraten, J.; Ertl, T. Design and Development of an Indoor Navigation and Object Identification System for the Blind. In Proceedings of the ACM SIGACCESS Conference on Computers and Accessibility (ASSETS'04), Atlanta, GA, USA, 18–20 October 2004.