SENSEYE
SENSOR NETWORK FOR SAFE INDOOR NAVIGATION

Alessandro Carella, Giada Cavallo
Politecnico di Milano

This paper studies the development and evaluation of a device meant to help people with visual impairments navigate their house safely. The aim is to create a product capable of warning the customer about the presence of objects out of place that could represent obstacles on their way. The product consists of a set of ultrasonic proximity sensors positioned around the room at ankle height and controlled by a specific program. The sensors gather information about the environment in terms of distance from objects along different directions; the information is then processed by the code to be finally conveyed to the customer via an application on his/her mobile phone. The app wasn't implemented with the prototype just as the localization of the customer in the room. In the present study we describe the working principles of the device, analyse the behaviour of the prototype, and compare the solution to existing ones on the market.

INTRODUCTION

According to the World Health Organization almost 314 million people live with serious visual handicaps, 45 millions are blind, 269 millions are visually impaired. The causes of this condition are various and therapy is often used to preserve the state of the eye from additional visual damage and not to actually cure blindness (see glaucoma). Visual impairments prevent patients from conducting ordinary activities independently and the support of a caregiver is often necessary. Among these, simply walking around the house can place serious threats to the health of a patient, especially due to falls, particularly dangerous for the elderly. Because most patients affected by visual impairments are elderly the need for assistive technologies able to help patients navigate their house safely is dire. In a study published in 2015 [1] several visually impaired elders were interrogated about the causes of falls and they indicated changes to a previously familiar home environment as one of the main causes. Examples of these changes are moved furniture or objects left on the floor by visitors. All these changes, they explained, represent dangers to visually impaired people that navigate their house relying mostly on acquaintance with the familiar environment.
Assistive technologies could be developed to warn visually impaired people about the presence of such hazards on their way, thus making it possible for them to avoid falls and remove the obstacles autonomously. With the aim of introducing such a technology we built a device that would serve the purpose efficiently and at low costs.

RELATED WORKS

Previous works have tackled the same problem.
The result of one of these works is "Safe Walk" [2], a walking stick equipped with electronics and able to warn the customer If an object is present on his way with 1 cm error. The solution is flexible in the sense that the customer can use it wherever he goes but at the same time the need to carry the stick everywhere represents a limitation, considering dimensions and weight of the object. Our device, instead, is thought to be implemented in the customer's house and he/she doesn't need to carry any part of it.
Other solutions consist in mapping the environment through cameras and recognizing obstacles using computer imaging technology. This technology is arguably too invasive and, given the level of

complexity, also costly. The working principle of this kind of products is similar to that of our device but the use of proximity sensors and more straightforward working algorithm makes our solution non-invasive and low cost. Of course, the use of proximity sensors makes it extremely difficult to gather other information about the obstacle more than its approximate position.
In conclusion, our project differs from the products available in commerce and could represent a new solution to be investigated and improved further.

WORKING PRINCIPLE

SPATIAL ARRANGEMENT

The device was thought as a non-portable system, consisting of a set of ultrasonic proximity sensors localized around the room and connected to a control board. The number of sensors and their position depend on the spatial configuration of the environment to be monitored. Each sensor is positioned on a servo motor that rotates allowing for a wider field of measurement (depending on whether the sensor is located near a corner or a wall it goes from 90 to 180 degrees, for the single sensor).

MAPPING THE ENVIRONMENT

Each sensor scans the environment with rotations of 30 degrees and measures the distance of the closest object after each rotation. The results of such measurements – for each sensor - are then saved in an array so that each element contains the distance of the closest object in a precise direction. The directions along which distance is measured are identified by the angles they form with a reference direction, relative to the set-up position as shown in figure. In this way we have created a map of the environment in terms of distances from the sensor.

FAMILIAR ENVIRONMENT

This operation is firstly performed to map the room in the most familiar conditions, that is when no obstacles are present and everything is in its place. The record of this map – the arrays with the distances from each sensor - is kept in memory and is used in the process of detection of obstacles.

DETECTION OF OBSTACLES

At regular intervals each sensor scans the space around it and obtains a new array of measurements. At the same time this array is compared, element by element, with the reference array obtained from the same sensor in familiar conditions. When the distances measured along the same direction, which are the values stored in the same elements of the two arrays, vary by a quantity bigger than the threshold, the system takes note by increasing the value of a counter. There is a different counter for each direction of measurement (i.e. for each couple of distances to compare) so that the detection of changes in the environment can be delimited to a particular direction. When the value of a counter is increased to a specific limit value the system certifies the presence of an obstacle in the direction associated to such counter. The limit value needs to be enough to avoid evaluation mistakes due to measure errors. At this point the system knows the direction of the obstacle with respect to the set-up position of the sensor (the counter contains the information about directionality) and the distance of the obstacle from the sensor along this direction. In other words, the system possesses the polar coordinates of the obstacle and its position in space, knowing the position of the sensor that detected it, is fully determined.

POSITION OF THE CUSTOMER

To be able to warn the customer when he/she is approaching the obstacle, his/her position in the room is required. To obtain this, indoor tracking systems can be employed. The tracking system needs to be very accurate because the type of indoor areas at issue are generally small. Moreover the scanning interval must be small, so to evaluate the position with the smallest possible delay, almost in real time. Among indoor tracking technologies Bluetooth positioning technology and Ultrasonic positioning technology are those that best satisfy the specifications of the product. Both these technologies can be easily implemented with modern smartphones [3].

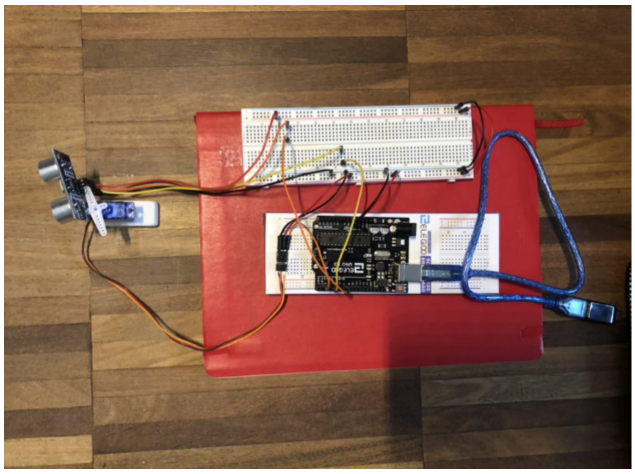
COMMUNICATION WITH THE CUSTOMER

Once the position of the obstacle and the one of the customer are determined, the system performs the calculation of the distance between the two points. This value of distance varies with time as the customer moves around the room and when it becomes smaller than a lower limit value the emergency protocol starts: the system opens a communication with the application on the customer's smartphone and the app promptly notifies the customer of the danger giving him/her information about his/her vicinity to the obstacle.

THE PROTOTYPE

DESIGN CHOICES

In the previous chapter we described the fundamentals of the final product of this study, which, however, wasn't realized. Instead, we built a prototype of this product with some of its features and limited in performance. The prototype consists of only one HC-SR04 ultrasonic sensor, placed on a Servo Motor SG90 and connected to a bread board and to an UNO R3 Controller Board with some wires of different length. All these components are part of a basic Arduino kit by Elegoo. The range of measure of the sensor is 2 cm – 400 cm, so it can ideally cover an area of approximately 25 m^2 .
The servo motor rotates the sensor by 30 degrees before it takes any measure, starting from the set-up direction: in this way, considering that from specifications the sensing angle is 30 degrees, the sensor is sensible to a continuous range of angles for a total covered area of 150 degrees.



TESTING

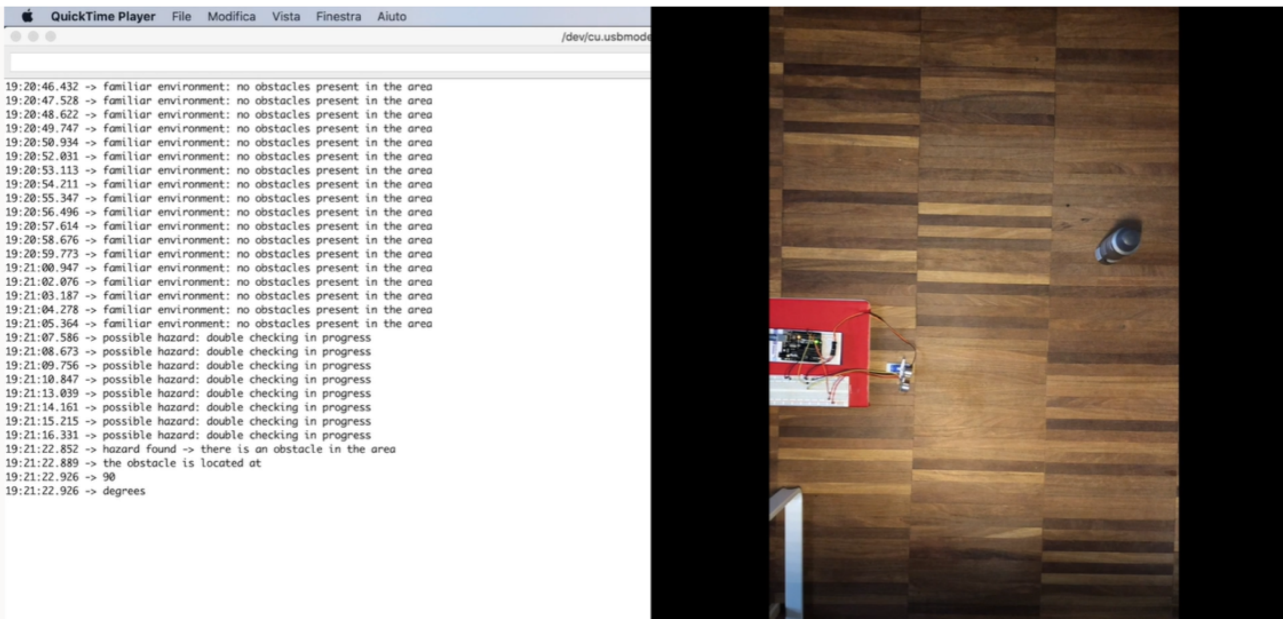
When the prototype was tested the servo motor was programmed to rotate four times, resulting in five directions of measurement (4 + 1 of set-up) and a covered area of 150 degrees in angle. The threshold of maximum harmless distance variation was set to 10 cm and the limit of the counter before entering the emergency protocol was set to 3. The code was written in such a way that when the system sensed a change in the familiar environment but the counter – relative to the direction where the change was sensed - didn't exceed the limit value, a notification would be printed on the screen to warn about the possible presence of an obstacle and that further checking was being carried out.
When one of the counter exceeds the limit value the emergency protocol starts and the system notifies the presence of the obstacle specifying its position in polar coordinates.
To test the prototype, different objects were placed in the working area of the sensor and it was checked whether the system could detect them and return their correct

location. These are the objects used to test the device:

- Deodorant case;
- Shower gel case;
- Tablet charger;
- Wireless headphones case;
- Towel.

The choice of the objects was made considering their dimension and the material they're made of. In particular, the objects had to be different in size and texture, so to test the system under a variety of situations.

To study the performance of the system an efficiency parameter was used: it is defined as the ratio between the number of correct detection and localization tests (both were considered for the correct functioning of the system) and the total number of tests. Each of these tests starts with a modification of the environment, whether it is the presence of a new object (new with respect to familiar conditions) or the absence of the same it doesn't matter, as long as the environment is modified in that moment. So, for example, when the deodorant case is placed in the area (as shown in the previous picture) a test starts and when the same case is removed another test starts.
This efficiency parameter was calculated for 25 tests and the result was 21/25 correct tests. This means that in 84% of tests the device performed well. It's important to notice that the system failed to detect and localize mostly the smallest objects when they were positioned far from the sensor. For example, the towel and the headphones case which are both smaller than 2 cm in height were not detected at all when placed at 1 meter distance from the sensor. The explanation is to be found in the tension of the wires that connect the sensor to the control board: when the sensor rotates the wires create some resistance slightly inclining the sensor; in this way the ultrasonic beam misses the target and the system cannot detect it. In support of this theory, it was demonstrated that the same towel, in the same position is detected if folded two times (so to triplicate its vertical dimension). This being understood, it is reasonable to calculate the same efficiency parameter but without considering the measurements of the two small objects at big distance: the result this time is 23/25 correct detection and localization tests, 92% of all tests.



```
19:20:46.432 -> familiar environment: no obstacles present in the area
19:20:47.528 -> familiar environment: no obstacles present in the area
19:20:48.622 -> familiar environment: no obstacles present in the area
19:20:49.747 -> familiar environment: no obstacles present in the area
19:20:50.934 -> familiar environment: no obstacles present in the area
19:20:52.031 -> familiar environment: no obstacles present in the area
19:20:53.113 -> familiar environment: no obstacles present in the area
19:20:54.211 -> familiar environment: no obstacles present in the area
19:20:55.347 -> familiar environment: no obstacles present in the area
19:20:56.496 -> familiar environment: no obstacles present in the area
19:20:57.614 -> familiar environment: no obstacles present in the area
19:20:58.676 -> familiar environment: no obstacles present in the area
19:20:59.773 -> familiar environment: no obstacles present in the area
19:21:00.947 -> familiar environment: no obstacles present in the area
19:21:02.076 -> familiar environment: no obstacles present in the area
19:21:03.187 -> familiar environment: no obstacles present in the area
19:21:04.278 -> familiar environment: no obstacles present in the area
19:21:05.364 -> familiar environment: no obstacles present in the area
19:21:07.586 -> possible hazard: double checking in progress
19:21:08.673 -> possible hazard: double checking in progress
19:21:09.756 -> possible hazard: double checking in progress
19:21:10.847 -> possible hazard: double checking in progress
19:21:13.039 -> possible hazard: double checking in progress
19:21:14.161 -> possible hazard: double checking in progress
19:21:15.215 -> possible hazard: double checking in progress
19:21:16.331 -> possible hazard: double checking in progress
19:21:22.852 -> hazard found -> there is an obstacle in the area
19:21:22.889 -> the obstacle is located at
19:21:22.926 -> 90
19:21:22.926 -> degrees
```

OTHER POSSIBLE IMPLEMENTATIONS

Our device could be easily integrated with indoor navigation systems. Let us imagine an indoor navigation app for smartphone which uses vocal feedback to guide visually impaired around the house. Such an application of virtual reality can provide the location of the customer to our device. The latter, then, scans the environment in search of obstacles and performs a risk assessment to finally communicate the result back to the smartphone.
The two solutions are highly compatible and provide a complete and robust product when joined together.

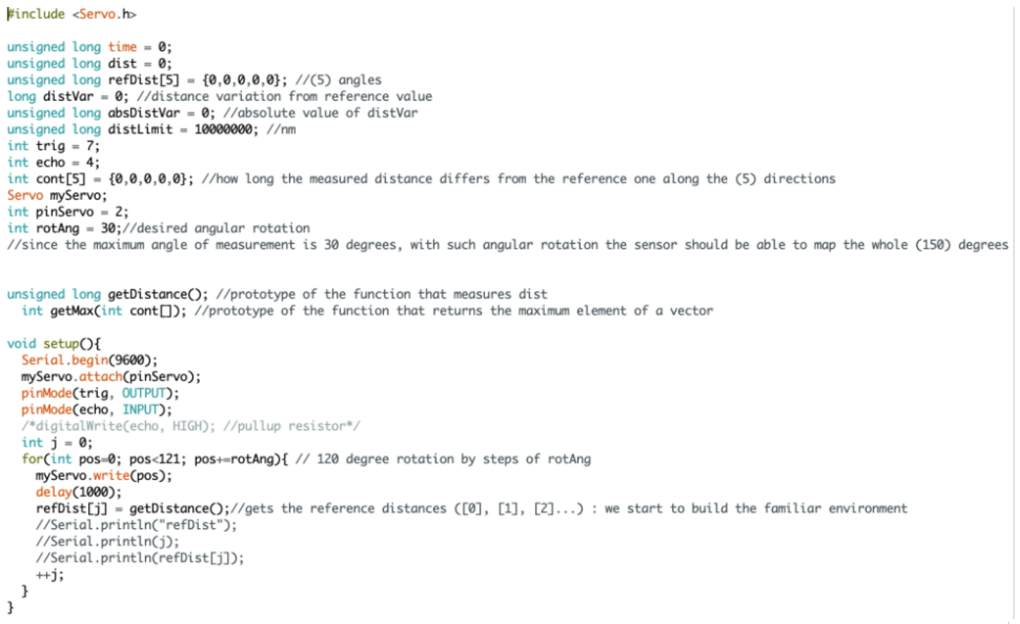
CONCLUSION

In conclusion, our product represents a cheap and simple solution to the issue of indoor safe navigation for visually impaired people. Nevertheless, its performance strongly depends on the performances of its single components. As a consequence, the choice of these components has an important role in the configuration of the product. Ideally, given the modularity of the device, we could configure it so to meet any customer’s preference in terms of accuracy and cost.

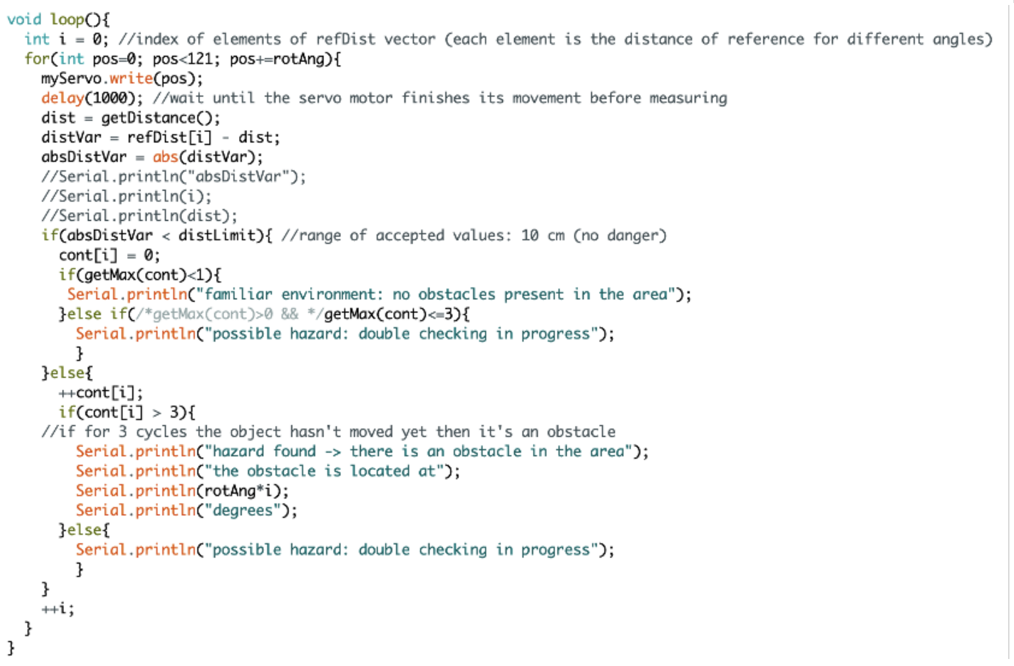
BIBLIOGRAPHY

[1] The causes of falls: views of older people with visual impairment, Wiley Online Library, 2015.
[2] https://www.safewalk.it/it/
[3] Urban Informatics, Shi, W., Goodchild, M., Batty, M., Kwan, M.-P., Zhang, A. , 2021

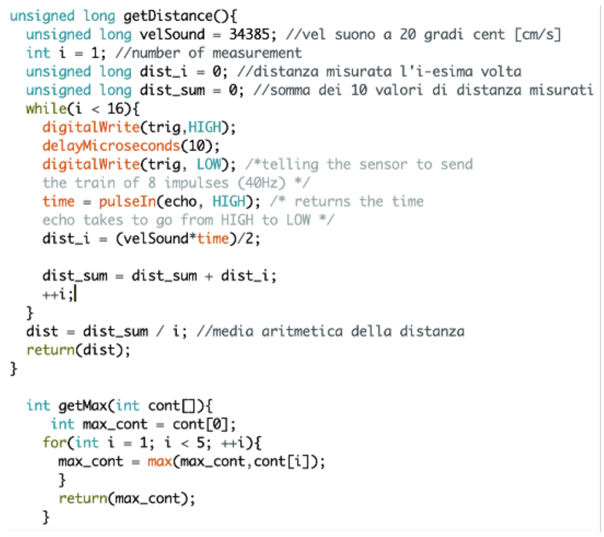
APPENDIX: CODE



```
#include <Servo.h>

unsigned long time = 0;
unsigned long dist = 0;
unsigned long refDist[5] = {0,0,0,0,0}; //(5) angles
long distVar = 0; //distance variation from reference value
unsigned long absDistVar = 0; //absolute value of distVar
unsigned long distLimit = 10000000; //nm
int trig = 7;
int echo = 4;
int cont[5] = {0,0,0,0,0}; //how long the measured distance differs from the reference one along the (5) directions
Servo myServo;
int pinServo = 2;
int rotAng = 30;//desired angular rotation
//since the maximum angle of measurement is 30 degrees, with such angular rotation the sensor should be able to map the whole (150) degrees


unsigned long getDistance(); //prototype of the function that measures dist
  int getMax(int cont[]); //prototype of the function that returns the maximum element of a vector

void setup(){
  Serial.begin(9600);
  myServo.attach(pinServo);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  /*digitalWrite(echo, HIGH); //pullup resistor*/
  int j = 0;
  for(int pos=0; pos<121; pos+=rotAng){ // 120 degree rotation by steps of rotAng
    myServo.write(pos);
    delay(1000);
    refDist[j] = getDistance();//gets the reference distances ([0], [1], [2]...) : we start to build the familiar environment
    //Serial.println("refDist");
    //Serial.println(j);
    //Serial.println(refDist[j]);
    ++j;
  }
}
```



```
void loop(){
  int i = 0; //index of elements of refDist vector (each element is the distance of reference for different angles)
  for(int pos=0; pos<121; pos+=rotAng){
    myServo.write(pos);
    delay(1000); //wait until the servo motor finishes its movement before measuring
    dist = getDistance();
    distVar = refDist[i] - dist;
    absDistVar = abs(distVar);
    //Serial.println("absDistVar");
    //Serial.println(i);
    //Serial.println(dist);
    if(absDistVar < distLimit){ //range of accepted values: 10 cm (no danger)
      cont[i] = 0;
      if(getMax(cont)<1){
       Serial.println("familiar environment: no obstacles present in the area");
      }else if(/*getMax(cont)>0 && */getMax(cont)<=3){
        Serial.println("possible hazard: double checking in progress");
        }
    }else{
      ++cont[i];
      if(cont[i] > 3){
    //if for 3 cycles the object hasn't moved yet then it's an obstacle
        Serial.println("hazard found -> there is an obstacle in the area");
        Serial.println("the obstacle is located at");
        Serial.println(rotAng*i);
        Serial.println("degrees");
      }else{
        Serial.println("possible hazard: double checking in progress");
        }
    }
    ++i;
  }
}
```



```
unsigned long getDistance(){
  unsigned long velSound = 34385; //vel suono a 20 gradi cent [cm/s]
  int i = 1; //number of measurement
  unsigned long dist_i = 0; //distanza misurata l'i-esima volta
  unsigned long dist_sum = 0; //somma dei 10 valori di distanza misurati
  while(i < 16){
    digitalWrite(trig,HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW); /*telling the sensor to send
    the train of 8 impulses (40Hz) */
    time = pulseIn(echo, HIGH); /* returns the time
    echo takes to go from HIGH to LOW */
    dist_i = (velSound*time)/2;

    dist_sum = dist_sum + dist_i;
    ++i;
  }
  dist = dist_sum / i; //media aritmetica della distanza
  return(dist);
}

  int getMax(int cont[]){
     int max_cont = cont[0];
    for(int i = 1; i < 5; ++i){
      max_cont = max(max_cont,cont[i]);
      }
      return(max_cont);
    }
```