A Low-Cost Time-Critical Obstacle Avoidance System for the Visually Impaired

D. Bernabei, F. Ganovelli, M. Di Benedetto, M. Dellepiane and R. Scopigno
Visual Computing Lab (CNR), Pisa, ITALY. Email: name.surname@isti.cnr.it

***Abstract*—We present a low cost system for unassisted mobility of blind people built with off-the-shelf technology. Our system takes as input the depth maps produced by the Kinect©device coupled with the data from its accelerometer to provide a registered point based 3D representation of the scene in front of the user. We developed a time-critical framework to analyze the scene and classify the ground and still or moving obstacles and provide the user with a constant and reliable feedback.**

I. Introduction

Exploiting state-of-the-art technology for improving people's everyday life is always a compelling challenge, especially when the people in question are impaired in some way. Over the recent years, the rapid evolution of color acquisition devices and computing hardware and their affordability has spawned a number of solutions to assist blind people in indoor and outdoor mobility. Most of the edge-breaking technologies have been applied to assist or train them: RFID [1][2][3], haptic devices [4], ultrasonic systems [5], virtual reality [6][7]. One of the latest steps of this evolution is the availability of low cost 3D scanners which are able to supply a three dimensional description of the scene in front of the device. It is straightforward to connect this novelty to the design of yet another system that would analyze the context and allow a blind person to move freely. This has some points in common with the robot guidance immense literature (see [8] for an application for impaired people), but the kind of feedback needed by the user is different. More specifically, given the 3D description of the scene, the algorithm identifies obstacles in the path, but there's no real way to constrain the movement of the user. Hence, the system should assist her/him during the exploration. Several ad-hoc sensors have been developed in the past [9][10][11], even able to recognize known features of inhabited environments, such as stairs or sidewalks. However, when considering 3D scanning technologies, we must confront ourselves with the fact that, while a color camera provides information that the blind person has no way to know, i.e. the colors, a 3D scanner-based system has two strong competitors against with it should be compared: the *white cane* and the *guide dog*. The white cane allows a person to explore the terrain in front of his/her foot in a range of approximately one meter (it depends on the length of the cane and the stature of the person), it is done with light weight materials, it is foldable and easily stored, it does not need any non-human power source and therefore is extremely fault-tolerant and finally it is very reliable, since the human brain does an excellent job in elaborating the haptic information returned by the cane. For this reason, some of the proposed solutions are integrated in it [3][11].

A trained guide dog does not provide a punctual information like the cane, but in this case it is not necessary since it does the brain work and leads the way. Obviously owning a dog is much more committing that owning a white cane and it may not even be possible for some people. In this paper we present an early implementation of a assistive navigation system for blind people based on a 3D scanner. We use a recent device, called Kinect©, released in the context of videogames and entertainment. The Kinect is essentially a low-cost short-range 3D scanner that is able to acquire na 640×480 range map of the scene at the pace of 20 times per second. Since the Kinect works in a range between 0.5 to 4 meters, it is perfect for the guidance of the movement of a person, and the quality data it provides is sufficient for the task of analyzing and detecting obstacles. The concept of our system is quite the same as [9] and similar works but there are substantial differences. Firstly, the prototype of our system is built with off-the-shelf technologies and it costs less than 500 Euros while the solutions proposed in the past were essentially costly prototypes. We consider this aspect of primary importance since, in our opinion, the market for blind people is not big enough to scale down the cost of ad-hoc technologies [10]. Secondly, we aim at exploiting the availability of solutions in the field of computational geometry to analyze 3D data and provide accurate understanding of the scene in front of the user. As we will see later on, these solutions are not directly appliable as they are, since almost all of them are not concerned with time-critical interruptible computation but only with a general concept of efficiency. We designed a simple framework, shared in a open source repository, for time-critical computation that is able to incorporate new algorithms in the system in a collaborative fashion.

II. System Description

Figure 1 illustrates the setup of the system. We fix the Kinect sensor to the belt so it can have an ideal coverage of both floor and object at human height. The Kinect is connected to a $12V - 3mAh$ battery pack and to a computing device such as a smartphone, which provides audio feedback to the user. The use of audio as interface is a temporary solution and it will be replaced by an haptic device currently being designed. Although the design of most systems for visually impaired people include audio as primary interface to provide

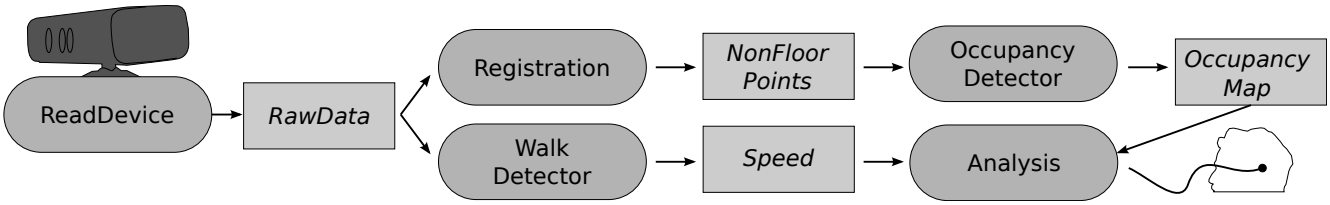

Fig. 2. A scheme of our time-critical framework.

feedback to the user, there is a fairly strong argument against this solution: we are essentially depriving the person of the use of hearing, which for a blind person is more than one sense.

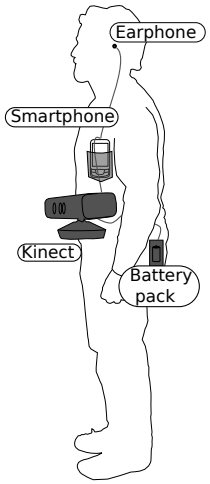

Fig. 1. Illustration of the proposed system.

A. A Time-Critical Framework

The feedback from data analysis must be responsive. Unlike with a robot, we cannot make a person *freeze* while waiting for system response, therefore the computation must be not only as fast as fast possible but also *interruptible*, i.e. the control must return in the time assigned with a meaningful, although conservative, answer. This constraint gives rise to a number of new problems in the field of computational geometry. At the present time there are a large number of techniques to analyze 3D data in order to discover regularities, symmetries, known shapes etcetera. While of the corresponding algorithms are designed to be fast, very few of them are also interruptible. In order to make a practical example, none of the several algorithms for surface curvature computation, which is often used as a building block for shape recognition algorithms, may be run with a time constraint or interrupted in the middle of the computation.

Our control software is organized as a series of interruptible tasks that run concurrently (the ellipses in Figure 2) and a series of data (the rectangles). Each task has its input data and produces its output data, and it is assigned with a time lapse by which it must complete its computation. In the current implementation we use the following tasks:

- **ReadDevice**: reads the data from the device and does the necessary conversion to express it as 3D points;
- **Registration**: detects the floor and registers the data in a reference system centered at the user's feet;
- **OccupancyDetector**: detects the occupancy of the volume in front of the user;
- **WalkDetector**: analyzes the output of the accelerometer to establish if the user is walking and at what speed;
- **Analysis**: provides the feedback to the users on the base of the result of the other tasks.

In the following we provide more details on how the single tasks are carried out with time constraint.

B. Registration

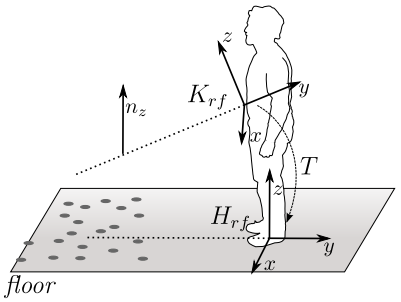

Fig. 3. Coordinate frames used by the *Registration* task.

This task consists of applying a geometric transformation to bring the 3D points from the reference system of the Kinect, indicated with K_{rf} in Figure 3, to the reference system centered at the user's feet with the z axis passing trough the barycenter and the head of the user, indicated with H_{rf}. Since the Kinect is mounted on the waist, it goes under ample movements when the user walks, which means that the transformation between these two reference systems, a 4×4 matrix called T from now on, must be recomputed many times per seconds. We do this by identifying the points belonging to the floor in the data input, i.e. in the frame K_{rf}, fitting a plane to those points and computing T as the roto-translation that express such plane in the frame H_{rf}. The transformation T is computed by assuming that the normal to the plane in world coordinates coincides with the z axis of H_{rf}. The translation part is simply the difference between the origin of H_{rf} and the origin of K_{rf} while the rotational part is given by:

$$R_{\vec{X}}(acos(n_z))) \cdot R_{\vec{Y}}(acos(\sqrt{({n_x}^2 + {n_z}^2)}))$$

where n is expressed in the K_{rf} frame, $\vec{X}$ and $\vec{Y}$ are canonical axes and $R_{ax}(r)$ indicates the rotation matrix of r radians around the ax axis.

We identify the points belonging to the floor as those closer than a threshold to the plane fitted at the previous step. The plane at step 0 is computed in a *calibration phase* where the user stays motionless without obstacles in front of him/her, so that it is easy to identify points belonging to the floor.
This simple technique relies on frame-to-frame coherency, therefore it is crucial that the time lapse between consecutive plane fitting operations is minimal, which means that the task, like any other, must complete within the assigned time interval.

Completing within the assigned time The time required for computation is directly proportional to the size of the input depth map, i.e. the number of points, therefore the algorithm may subsample the input data or the assigned time interval is otherwise insufficient. The subsampling procedure consists of a *reduce* operation, in which an input depth map is reduced to a smaller one using a *min* operator. That is, to guarantee conservative results, i.e. that no obstacle goes undetected, each used sample is assigned with the minimum depth among the original samples it represents.

C. OccupancyDetector

This tasks must tell, for each direction the user could walk, the distance to the closest obstacle. Because the user needs a free space corresponding to his/her height and width, the task first computes a low-resolution, one-dimensional $D \times 1$ depth map using a conservative process as in the Registration task. The resulting map is therefore a quantization of the space in front of the user into D values, each one corresponding to a vertical volume spanning a range of possible directions. The produced depth map is then used to find, for each direction, the farthest point reachable by the user considering his/her width. Figure 4 shows a depth map of size 8, in which the most right region is not reachable although *visible* from the device. Note that we have no use of a high resolution depth map since ultimately the user makes imprecise movements. In our tests we empirically found that a reasonable size of the depth map is 32.

Completing within the assigned time Like for the Registration task, the time for the OccupancyDetector is proportional to the size of the depth-map that can be reduced to fulfill the time constraint. However, note that this task is very simple and fast to complete, so that its interruption was never necessary in practical cases.

D. WalkDetector

This task uses the accelerometer mounted with the Kinect to detect if the user is walking and at, approximately, at what speed. Since the device is fixed at the height of the pelvis, when the user walks it moves from left to right and vice versa. Therefore, just like any step counter does, we track the values provided by the accelerometer and count the frequency of changes of sign of derivatives. To avoid errors introduced by high-frequency noise, the input signal is first smoothed using a simple *finite impulse response* filter.

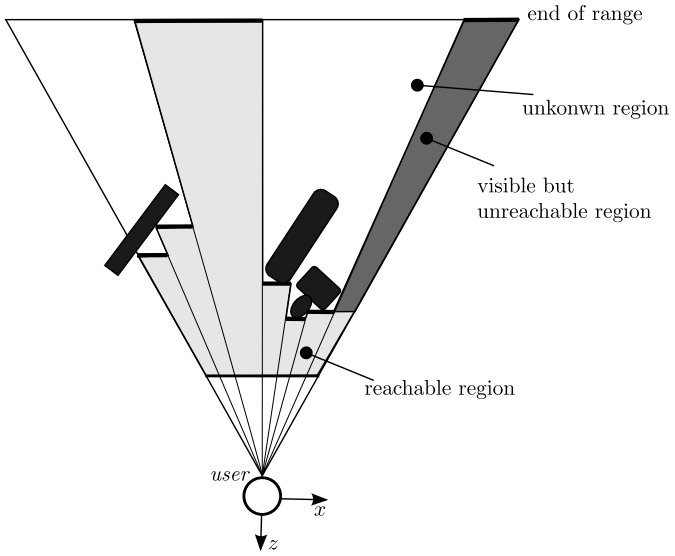

Fig. 4. Linear depth map of size 8.

E. Analysis

This task reads the input provided by the OccupancyDetector and the WalkDetector and decides what feedback to give to the user. The algorithm consists of tracking the result of the OccupancyDetector to detect if the user should be told to change direction or to stop. Note that the occupancy of a region of space is an obstacle only if the user is moving towards said region. If, for example, the user is walking along a corridor, the walls will occupy the space on left and right, but they are not obstacles. Another example is represented by someone walking in front of the user in the same direction. In this case, the distance to the occupied region will be roughly constant. However, roughly means that we must filter off oscillations due to human walking which means, in turn, that we should know if the person is walking, and this is where WalkDetector is comes into play.
In the current implementation, the Analysis task monitors the occupancy map and tests if an obstacle is approaching the user. If this is the case it suggests the closest direction to take to avoid the obstacle. If there is no such direction, it notifies the user the he/she is approaching to a dead end. The speed estimated by the WalkDetector is used to assign the dynamically set time constraints to the other tasks. If the user is walking slowly or is standing still, we need a lower refresh rate than if the user is walking fast. In other words, we may say we express the update frequency per meter. As a conservative policy, if the OccupancyData is not updated on time the Analysis task tells the user to stop.

F. Audio feedback

We tried two alternative audio feedback responses. The first consisted in using few simple messages ("proceed","keep left","keep right","stop") that were given according to the simple goal of walking towards the farthest reachable place visible from the scanner. The second was a continuous tone used to indicate the direction to the farthest reachable place by mapping the direction on the stereo balance, and the distance to the place in question to the tone pitch.

III. Experimental Results

We implemented our platform-independent framework in C++, using Qt for multithread/multiprocess foundations and OpenAL for audio feedback and run the system on a Eee PC 1015PD 1.66 GHz, 1 GB RAM. Note that the Eee PC is not a smart phone but it is less powerful of many currently available smarphone such as the Motorola Atrix (dual core with an NVidia Tegra 2 graphics card). During development and test we used OpenGL to have a three-dimensional overview of the acquired data.

We tested the system on a blindfolded, non visually impaired person. This choice was made for the twofold reason that the system is not yet in its final release and that a born blind person is used to walk without seeing and it would be harder to evaluate the actual contribution of our system. The subject was brought to an unfamiliar location and asked to reach a point identified as the source of a sound. The location was filled with obstacles such as chairs, desks and books on the ground and there were other people moving in the room. As expected, the user was able to safely reach the target place avoiding the obstacles and taking the shortest way. It is clear that this is still a controlled environment where, for example, there are no stairs, therefore the tasks essentially work as proximity detectors. However, even at this early stage we could observe the effects of the time-critical system. After the first test was completed, we asked the user to do another one by walking as fast as he felt like. In this case the system asked the user to stop when the first obstacle was approaching. This happened not because the OccupancyDetector could not compute the direction to move to, but simply because it could not do it fast enough.

We observed that, after few minutes, the use of the continuous tone for providing feedback was more profitable that single messages. This did not come as a surprise since the tone gives the essentially same information processed by the Analysis task, but lets the brain do the analysis.

IV. Discussion and Future Work

As stated in the introduction, any new technology should be compared with what is already there. At the present development stage, it is clear that the white cane wins in many regards: it is cheaper, it is lighter, it does not need electric power and it is more reliable in detecting obstacles on the floor, even if the floor is a perfectly reflecting surface, in which case laser scanning devices fail. However, even at this early implementation stage, there are advantages in using our system. First of all, the user may walk faster because we have 4 meters view of the 3D space and not just of the floor. Secondly, when there is some obstacle the system tells where to go while with the cane the user must explore the neighborhood looking for a free path. Also, there are already many image based applications that can be seamlessly integrated in our framework, for example to read writings or ad hoc signals, to track or even recognize known people. Concerning the cost and weight, it is easy to predict that the 3D range scanners will be, even in a short while, cheaper and smaller then they are now and comparable to the cane it terms of weight and price. However, in our opinion, there are two aspects that can really make the difference and that are conditio sine qua non for the actual use of these systems. First, how much they are, or will be, safe. Walking without seeing and without touching requires a deep trust on the supporting system. In this regard, we have focused on the interruptibility of algorithms for analysis of 3D data which we believe to be the key to reliability and we will continue to do so in the next steps of this project. Second, from the hardware point of view, we should concentrate on an efficient haptic interaction that leave the hearing free. Although in this first implementation we used audio feedback and implemented some simple artificial intelligence to provide simple indications to the user, we believe that haptic feedback has the potential to provide a sufficient description of the surrounding environment to allow him/her to move safely.

References

[1] J. Faria, S. Lopes, H. Fernandes, P. Martins, and J. Barroso, "Electronic white cane for blind people navigation assistance," in *World Automation Congress (WAC), 2010*, 2010, pp. 1 –7.

[2] E. D'Atri, C. Medaglia, A. Serbanati, and U. Ceipidor, "A system to aid blind people in the mobility: A usability test and its results," in *Systems, 2007. ICONS '07. Second International Conference on*, 2007, p. 35.

[3] S. Chumkamon, P. Tuvaphanthaphiphat, and P. Keeratiwintakorn, "A blind navigation system using rfid for indoor environments," in *ECTI-CON 2008*, vol. 2, May 2008, pp. 765 –768.

[4] "Haptic virtual environments for blind people: Exploratory experiments with two devices," *International Journal of Virtual Reality*, vol. 3, pp. 10–20.

[5] P. S. Blenkhom P. and E. D. G., "An ultrasonic mobility device with minimal audio feedback," in *12 Annual International Conference on Technology and Persons with Disabilities*, 1997.

[6] M. A. Torres-Gil, O. Casanova-Gonzalez, and J. L. Gonzalez-Mora, "Applications of virtual reality for visually impaired people," *W. Trans. on Comp.*, vol. 9, pp. 184–193, February 2010.

[7] A. Hub and J. Diepstraten, "Augmented indoor modeling for navigation support for the blind," *: CPSN'05-The International Conference on*, 2005.

[8] R. Bostelman, P. Russo, J. Albus, T. Hong, and R. Madhavan, "Applications of a 3D Range Camera Towards Healthcare Mobility Aids," *2006 IEEE International Conference on Networking, Sensing and Control*, pp. 416–421, 2006.

[9] T. Ueda, H. Kawata, T. Tomizawa, A. Ohya, and S. Yuta, "Visual Information Assist System Using 3D SOKUIKI Sensor for Blind People, System Concept and Object Detecting Experiments," *IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*, pp. 3058–3063, Nov. 2006.

[10] F. G. Peris, L. Dunai, P. V. Santiago, and I. Dunai, "CASBliP - a new cognitive object detection and orientation aid system for blind people," *CogSys2010 Conference*, 2010.

[11] S. Shoval, I. Ulrich, and J. Borenstein, *Computerized obstacle avoidance systems for the blind and visually impaired*. Boca Raton, FL, USA: CRC Press, Inc., 2001, pp. 413–448.