Smart Walker for the Visually Impaired

Christopher Feltner, Jonathan Guilbe, Sharare Zehtabian, Siavash Khodadadeh, Ladislau Bölöni, and Damla Turgut
Department of Computer Science
University of Central Florida
{chris.feltner, jguilbe, sharare.zehtabian, siavash.khodadadeh}@knights.ucf.edu
{lboloni, turgut}@cs.ucf.edu

***Abstract*— Visually impaired individuals often employ canes or guide dogs to help navigate complex environments. Individuals who are both visually and mobility impaired encounter greater difficulty, since conventional aids do not integrate well with walkers or rollators. In this paper we propose a smart walker architecture that is augmented with depth-sensing cameras that detect and recognize obstacles that may endanger the user, as well as obtain their distance from the user. This information is conveyed to the user through a haptic interface on the walker. Our design helps protect the user from colliding with obstacles in his/her path. In comparison with traditional mobility methods, the smart walker allows the user to navigate the environment faster and with less physical and cognitive load. Compared to previous designs, our approach completes this task at a significantly lower cost per unit.**

I. Introduction

Navigating complex, unknown environments is an integral part of human life, but for individuals with disabilities this task is more challenging. Those in particular need of assistance are individuals who are both mobility impaired and visually impaired, a group which includes many elderly people. A report by the World Health Organization [1] noted that 81.7% of blind people worldwide are 50 years or older. In an analysis of the 2004 Health and Retirement Study [2], it was shown that 16% of individuals 65 years or older reported use of a mobility device. Unfortunately, conventional navigation aids for the blind are not designed to be used in conjunction with traditional mobility aids, such as walkers. A common technique for walker users who are blind is to stop periodically and observe their surroundings using a white cane, which is slow, tedious, and reduces their mobility. Some visually impaired individuals need to be guided by a sighted individual in order to get around, which decreases the autonomy of the visually impaired individual.

In this paper, we propose a smart walker for the blind that provides feedback about obstacles in the user's path of travel. This solution attempts to make the user safer, more mobile, and less dependent on caretakers. In our design process, we had several goals:

1) Create a solution that is lightweight, portable, and comfortable for the user. In order to ensure portability, all processing should be completed on the device, since a mobile network connection is not always assured. This precludes networked and cloud-based solutions.
2) Provide feedback to the user about the distance to the obstacles, rather than simply alerting her to the presence of an obstacle.
3) Provide feedback to the user without the use of audio, to avoid impeding the user's sense of hearing [1].
4) Create a solution that is low-cost.

We describe two approaches for recognizing obstacles. The first approach involves an analysis of depth images produced by the Kinect RGB-D camera. The algorithm searches for sudden changes in depth along the user's assumed direction of travel. We show that the low computational requirements of this algorithm allows it to run on low-cost, low-power computing devices, such as the Raspberry Pi. The second approach involves processing a point cloud constructed from a depth image. This approach differentiates between the floor plane and potential obstacles in a point cloud, and segments the obstacles in the cloud. For both approaches, the smart walker is able to recognize obstacles and obtain the distance between those obstacles and the walker.

The rest of this paper is organized as follows. In Section II, we survey previous work in this field. In Section III, we describe the design of our walker. In Section IV, we describe our proposed approaches. In our first approach, we solve the problem of identifying obstacles and finding their distance by processing depth images. In our second approach, we employ point cloud processing. Section V describes the evaluation of the proposed solutions. Finally, in Section VI we conclude and offer possible future improvements.

II. Related Work

Several navigational aids have been proposed for visually impaired individuals, and there has been activity in employing the Microsoft Kinect system in accessibility technologies. Orita et al. [3] proposes a white cane device that uses depth information from a Microsoft Kinect camera to detect obstacles in indoor environments. If the user encounters an obstacle, the device vibrates to alert the user. This reduced navigation time in their study comparing the device's performance to a conventional cane. Since the device is held by the user, a significant portion of their work is devoted to adjusting for the angle at which the user holds the device. It was also noted that users found the device to be heavy and its use to be fatiguing. Since our system is mounted on a walker, we are able to avoid these issues.

The use of RGB-D cameras in wearable navigational aids for the visually impaired has been previously investigated. In Pham et al. [4], a system is devised to deliver a blind user feedback on their environment using depth data produced by

the Kinect. The device is mounted on the abdomen of the user, and a laptop computer backpack is worn. The system is capable of identifying drops, objects, walls, and other potential obstacles. Feedback to the user is provided using a sensory substitution device called a Tongue Display Unit. While this method produces robust detection results, and in this work we employ a similar approach, the feedback method and heavy materials could be considered unwieldy. In Wang et al. [5] a wearable camera system recognizes objects in front of the user and provides information about these objects through a haptic belt and braille display. The use of Kinect to guide users is also surveyed in Elmannai and Elleithy [6].

Several smart walkers for the visually impaired have been proposed in the past. In MacNamera and Lacey [7], a smart walker named PAM-AID is devised and evaluated. This smart walker is capable of alerting the user to obstacles using vocal feedback and taking control of the steering to automatically avoid obstacles in the user's path. In order to detect obstacles, this walker uses an array of sonar sensors and laser ranging devices. Yu et al. [8] propose a smart walker and smart cane for the elderly and evaluate a shared control interface for the device. The device is capable of avoiding obstacles and can determine its location in a care facility by reading signposts mounted on the ceiling of the facility. Paulo et al. [9] proposed ISR-AIWALKER which focuses on safe navigation and allows the user to maneuver the walker safely and avoid obstacles in more complex environments.

The importance of this work is underscored by the fact that there are also efforts to commercialize such a technology. PAM-AID was further developed by Lacey and Diego Rodriguez-Lousada of Haptica into Guido [10], which is a motorized rollator employing sonar and laser ranging devices to avoid obstacles. Guido also employs a simultaneous location and mapping algorithm combined with an extended Kalman filter (SLAM-EKF). SLAM-EKF allows the system to build a map of its environment in real-time, making it capable of guiding a user to a predetermined destination using a previously built map. It is noted by the makers of Guido that an obstacle which must be overcome in order to develop smart walkers further is the reduction of sensor cost. In a joint project with Siemens, a method to track moving objects and achieve simultaneous localization and mapping (SLAM) using Microsoft Kinect is proposed by Panteleris and Argyros [11] for the purpose of the development of their c-Walker. The use of Microsoft Kinect in the c-Walker and the employment of haptic feedback make the c-Walker similar to this project.

Other groups have attempted to reduce sensor cost. In Chaccour et al. [12] a smart walker is proposed which detects obstacles through IR and ultrasonic sensors, and alerts the user to a collision through audio feedback. Since these sensors are much cheaper than laser ranging devices, the total cost was considerably lower than PAM-AID and Guido, but the use of audio feedback can impede navigation by blind users. The choice of sensors also precludes the generation of point clouds, which is necessary to employ mapping algorithms such as those proposed by Lacey and Diego Rodriguez-Lousada [10].

Our work joins others in the e-health field, such as Boudjit and Moungla [13], where two fundamental mechanisms of Wireless Body Area Networks (WBANs)–data dissemination and sensor deployment–are reviewed, as they are intended to provide benefits across various healthcare applications.

This project builds on our lab's previous work in IoT augmented walkers which provide feedback to users. In Zehtabian et al. [14], a smart walker is proposed which provides feedback to users to promote prescription compliance and proper walker usage. In Khodadadeh et al. [15], the collected data stream has been processed by a deep neural network classifier, so it can learn to detect unsafe use patterns of the walker. The dataset has been labeled as unsafe and safe patterns and the classifier can predict the usage type (safe/unsafe) from the input data stream in real time. Viegas et al. [16] presented a system to monitor the usage of walker assistive devices for users to prevent dangerous situations by guiding them to use the device correctly.

In this project, we seek a solution which is affordable, portable, which can provide feedback without use of audio, and which can alert the user to distance from obstacles, not simply the presence of an obstacle.

III. Design

As shown in Figure 1, our walker consists of a standard off-the-shelf four-wheeled rollator with a Microsoft Kinect camera mounted on the basket and angled towards the floor. The Kinect power supply was modified to be connected to a 12V DC rechargeable battery. Data processing is performed by a Raspberry Pi computer in the first approach, and in the second by a laptop computer. Vibration motors are attached to the handles of the rollator, allowing the walker to provide feedback to the user.

The use of the Microsoft Kinect sensor contributes towards a low-cost design. Whereas in most previous approaches using point clouds, the cost of sensors such as laser ranging devices is on the order of thousands of dollars, while in this approach the cost is on the order of tens of dollars.

The decision to use tactile over audio feedback is based on the fact that for visually impaired people, the sense of hearing is used to supplement the lack of eyesight while navigating. Requiring the user to wear earphones could impede navigation since it reduces their ability to hear ambient sounds. The haptic feedback system categorizes obstacles into three groups: close, mid-range, and far. Each category is assigned a different vibration intensity, with far being the least intense and close being the most intense. This allows visually impaired users to determine the distance to an obstacle and judge the level of danger an obstacle presents.

IV. Proposed Approaches

A. Background and Preliminaries

The Microsoft Kinect camera is an RGB-D camera, which is able to produce both conventional color images (RGB) and

Fig. 1. The smart walker configured for the point cloud processing approach.

depth images. A depth image is a two-dimensional array of values, where each value represents the distance in millimeters from an object. This can also be represented as a grayscale image, as in Figure 2, with darker colors representing closer distances and lighter colors representing farther distances. Error state values are represented by black regions. These pixels have a value of zero, since no depth information was received by the Kinect in these regions.

In a point cloud, each point is represented by its x, y, and z values. These values represent the position of the point in space. Using OpenNI and the Point Cloud Library (PCL), we can construct a point cloud from the depth data produced by the Kinect sensor. We can also use PCL to represent and process point clouds [17].

A Raspberry Pi is a small, energy efficient Linux computer. It can be powered via a MicroUSB connection from a battery onboard the walker. Unlike a traditional personal computer, a Raspberry Pi has General Purpose Input/Output (GPIO) pins, making it capable of interfacing with devices such as LED lights, motors, or sensors.

B. Approach 1: Depth Images

The use of depth maps to detect obstacles was based on the work done by Ortigosa et al. [18]. In this approach, we assume that in depth images with no obstacles, the depth will increase at a steady rate with respect to the index from the bottom of the image to the top of the image. In order to identify obstacles, we first obtain a depth image from the Kinect sensor. Since we are concerned with obstacles in the user's path of travel, and we assume that the user will be travelling forward in a straight line, we concern ourselves with the vertical center line of the image. In order to reduce noise and capture obstacles close to, but not exactly in the center of the image, we then average depth values in each row of a region in the center of the image, as shown in Figure 3. In pixels where the Kinect cannot determine a depth, an error state value of 0 is reported. We remove these error state values from the image. Instead of using linear regression, we simply iterate through the resulting array of values, calculating the slope between each pair of points. An obstacle is identified when a slope is negative, or is greater than a set threshold. This indicates the existence of an obstacle, or a drop in floor level, respectively.

Decreases or plateaus in depth values are indicative of obstacles. Once we detect a negative slope, we iterate through the array of averaged values beyond the detected anomaly to find the minimum distance value. This ensures that we identify the distance of the obstacle closest to the user while also segmenting the region of the array that represents the floor.

Sharp increases in depth values are signs of a dangerous decrease in elevation. In testing, this occurred when the Kinect was placed directly in front of stairs going downwards. In this case, we report the distance just before the jump as the distance to the missing floor. Less extreme cases of elevation decrease, such as a curb dropping off, also trigger increases in depth values that need to be detected and reported to the user.

Algorithm 1: Finding Distance to Obstacles in a Depth Image

```
1  image = getDepthImage();
2  removeErrorValues(image);
3  averageDepths[] = averageAcrossCenterColumn(image,
     widthOfColumn);
4  for i = 0; i < averageDepths.length; i++ do
5      slope = averageDepths.differenceQuotient(i, difference);
6      if slope < negativeThreshold or slope >
         positiveThreshold then
7          minimumIndex = 0;
8          maximumIndex = i;
9          return getMinimumDepthInRange(averageDepths,
             minimumIndex, maximumIndex);
10                          ▷ Obstacle found at returned distance.
11 return null;
12                                          ▷ No obstacle found.
```

C. Approach 2: Point Clouds

In order to recognize obstacles, we employ an approach similar to that of Pham et al. [4]. We first use OpenNI and PCL to generate a point cloud from distance data captured by the Kinect camera. Since the user must be notified of obstacles which might be in their path of travel, we remove points which are outside of this path. Since the width of the walker is about

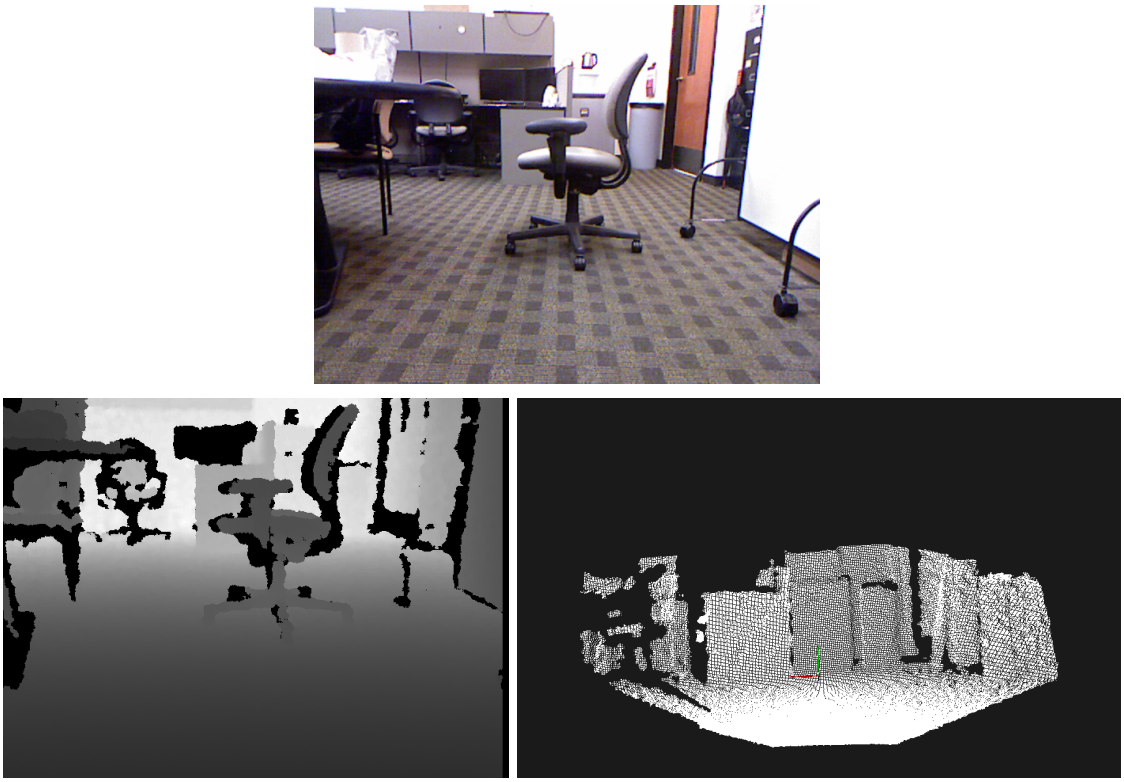

Fig. 2. Example of a color image (top), depth image (bottom left), and point cloud (bottom right).

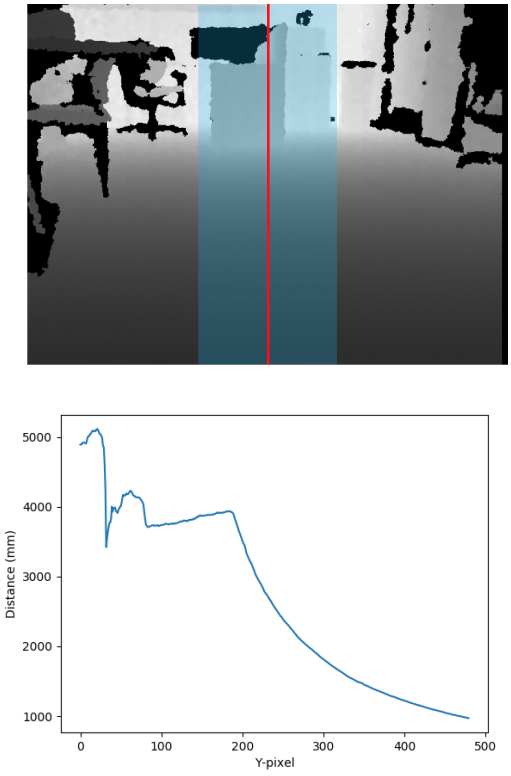

Fig. 3. A depth image with an averaged column and resulting data displayed as a Distance over Y-pixel graph.

70 cm, we remove points that lie beyond this width with a 5 cm tolerance on each side. As the camera is mounted in the center of the walker, we use a Pass-Through filter to remove points that have x-values greater than 0.4 or less than -0.4; 40 centimeters on each side of the camera. We also use a Pass-Through filter to remove points with values of greater than 4.0 or less than 0.4 on the z-axis. This removes points farther than 4 meters or closers than 40 centimeters from the camera, as these points lie beyond the Kinect's specified range of accuracy.

Since the size of a point cloud can be on the order of hundreds of thousands of points, we must downsample the cloud in order to reduce runtime. In order to downsample the cloud, we employ a Voxel Grid filter with a leaf size of 1 cm.

We now need to differentiate between points which belong to the floor and those which belong to obstacles. We assume that the floor plane will be the largest plane parallel to the z-axis. We use Random Sample Consensus (RANSAC) to determine coefficients of a model of a plane parallel to the z-axis that fits the largest such plane in the point cloud, and points within a threshold distance from the plane model are considered inliers. We then extract the points not included in the plane, which we consider to be obstacles. If no points match this model of a plane, this indicates that either a sudden drop or an obstacle which blocks the majority of the camera's field of view exists. In this case, the user is warned of an obstacle at close range. The segmentation process is shown in Figure 4.

As in Li et al. [19], if the number of remaining points

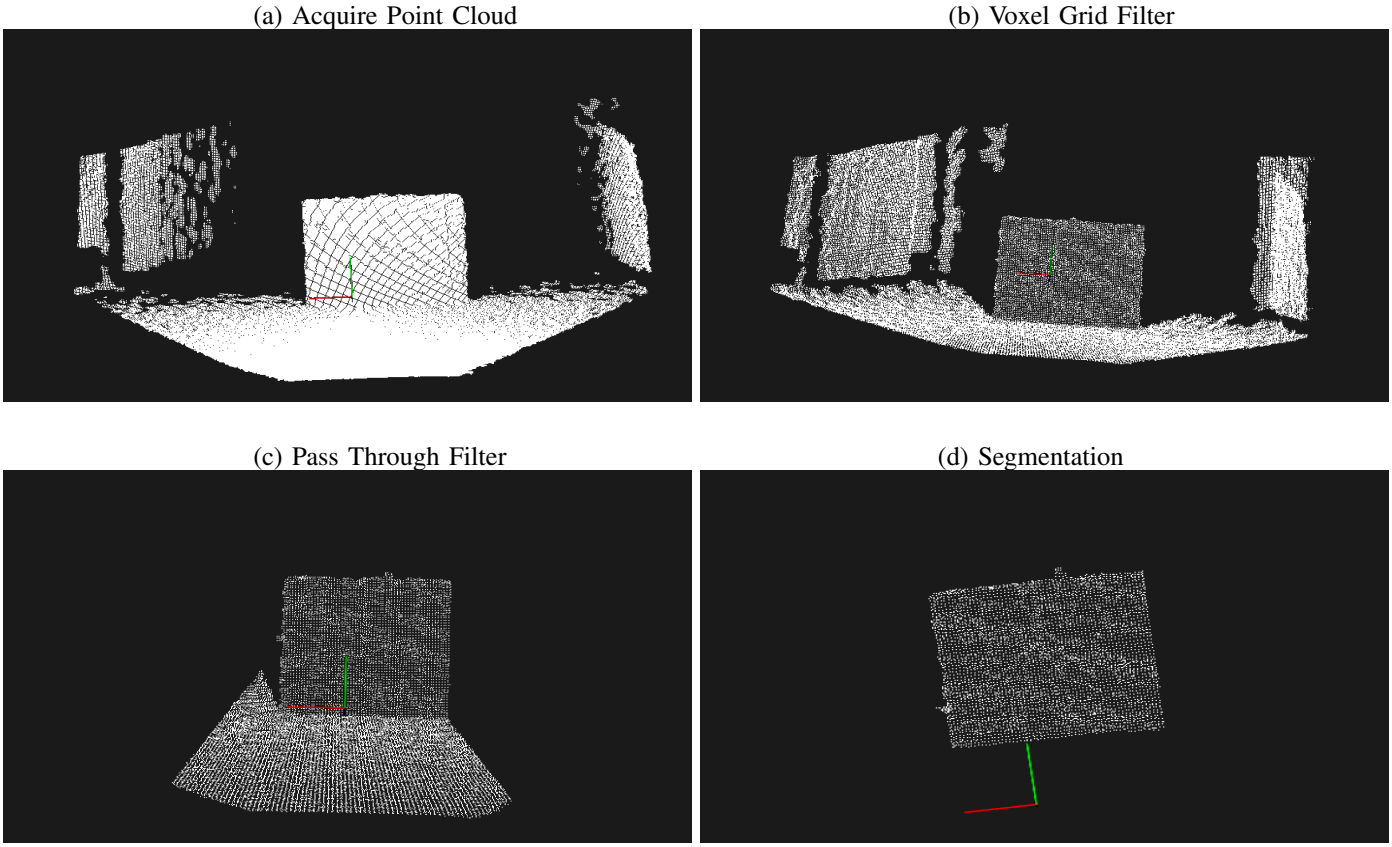

Fig. 4. Demonstrating the segmentation process: (a) A point cloud is acquired from the Microsoft Kinect sensor using OpenNi and PCL. (b) A voxel grid filter is applied to the image to downsample, improving run-time of the algorithm. (c) A pass-through filter is applied to remove points which are not in front of the user or are outside of the specified accuracy of the sensor. (d) RANSAC is used to identify the coefficients of a plane equation for the largest plane parallel to the z-axis. Points within a threshold distance from this plane are assumed to be part of the floor and are removed from the cloud.

are above a threshold value, we iterate through all points and identify the minimum z value, which we consider to be the distance from the obstacle. If the number of remaining points is below a threshold value, the area is determined to be free of obstacles.

V. Results

The walker was evaluated on its ability to detect four types of obstacles the walker users are likely to encounter on a daily basis (see Table I). It was also evaluated on its ability to recognize a path free of obstacles. The actual distance from the obstacles to the walker were measured using a tape measure and were compared to the distances reported by the walker software.

The walker was able to identify the distances from obstacles to within 10 centimeters of the actual distance. Because we will report this distance to the user at a granularity much greater than 10 centimeters using haptic feedback, this is an acceptable range of error for our purposes. It was also able to identify a path free from obstacles, and a potentially dangerous drop.

The walker was not effective at identifying obstacles in brightly sunlit environments. This is due to the fact that distance is computed using an infrared sensor and emitter, and the infrared rays from the sun corrupt the images produced by this sensor. Therefore, the current design is most suited to indoor environments, and additional work is needed to make the walker perform well in outdoor environments.

TABLE I

Measured distances

Obstacle Type	Approach 1	Approach 2	Actual
None	*	*	*
Wall	130cm	120cm	128cm
Drop	+	+	+
Bottom of Stairwell	129cm	119cm	125cm
Item in Path	187cm	172cm	182cm
"*" denotes that the path is free of obstacles			
"+" denotes a drop or close obstacle was identified			

An evaluation of the effectiveness of the haptic feedback system was also performed (see Table II). The test compared performance with a conventional rollator/cane combination with our design in two different environments. The first sample environment was an empty hallway, where the user had to avoid collisions with the walls. The second one was a hallway with obstacles placed in the path of the user, requiring the user to use the aid to avoid the obstacles. The test subjects were sighted individuals without mobility impairments. The eyesight of these

TABLE II
HAPTIC FEEDBACK EVALUATION RESULTS

Aid Type	Environment	Avg. Time	Avg. Obstacles Hit
Smart Walker	Empty Hallway	2:47	5
Smart Walker	Hallway w/ Obstacles	3:37	2
Walker/Cane	Empty Hallway	3:02	3
Walker/Cane	Hallway w/ Obstacles	1:53	0

subjects was blocked during the test. The time taken by these individuals to traverse the hallway and number of obstacles hit were measured.

The walker is most effective when navigating clear spaces. Based on observation during the trials, it was determined that the walker was not as effective in situations with many obstacles since the walker does not provide information to the user about the direction of the obstacle, or which way to go in order to avoid the obstacle. A possible improvement could be to add a "turn-signal" like system to convey an obstacle free direction of travel to the user.

Furthermore, it was observed during the evaluation that the presence of false positives in obstacle recognition slowed down the users. While a cautious approach was taken to avoid the presence of false negatives in obstacle reporting, false positive results proved to be more confusing to the walker user than originally anticipated. Increasing the threshold for false positives makes it more difficult for the walker to operate in real-time, so careful testing will be necessary to further improve the performance.

VI. CONCLUSION

In this research, we have designed a smart walker capable of identifying obstacles that a visually impaired user might collide with, and obtain the distance from the obstacle to the user. We found that the walker is capable of identifying this distance with a sufficient accuracy to warn the user of the obstacle. We have developed two methods of identifying obstacles. The first approach analyzed depth images produced by the Kinect RGB-D camera while the second approach involved processing a point cloud constructed from a depth image.

Possible further research on this topic includes improving the accuracy of the walker by creating a method of identifying obstacles in sunlit environments, improving detection of downwards drops, reducing false positive results, and improving detections of obstacles parallel to the user. Improvements to the usability of the walker include identifying the location of obstacles relative to the user and signaling to the user a new direction of travel free of obstacles, and eliminating the need for a full-size laptop computer in order to run the approach involving point clouds.

ACKNOWLEDGEMENT

The support for this work was provided by the National Science Foundation REU program under Award No. 1560302.

Any opinions, findings, and conclusions and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] A. Wachaja, P. Agarwal, M. Zink, M. R. Adame, K. Möller, and W. Burgard, "Navigating blind people with a smart walker," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 6014–6019.

[2] N. M. Gell, R. B. Wallace, A. Z. Lacroix, T. M. Mroz, and K. V. Patel, "Mobility device use in older adults and incidence of falls and worry about falling: Findings from the 2011–2012 national health and aging trends study," *Journal of the American Geriatrics Society*, vol. 63, no. 5, pp. 853–859, 2015.

[3] K. Orita, H. Takizawa, M. Aoyagi, N. Ezaki, and M. Shinji, "Obstacle detection by the kinect cane system for the visually impaired," in *IEEE/SICE International Symposium on System Integration (SII)*, 2013, pp. 115–118.

[4] H.-H. Pham, T.-L. Le, and N. Vuillerme, "Real-time obstacle detection system in indoor environment for the visually impaired using microsoft kinect sensor," *Journal of Sensors*, vol. 2016, 2016.

[5] H. Wang, R. K. Katzschmann, S. Teng, B. Araki, L. Giarr, and D. Rus, "Enabling independent navigation for visually impaired people through a wearable vision-based feedback system," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 6533–6540.

[6] W. Elmannai and K. Elleithy, "Sensor-based assistive devices for visually-impaired people: Current status, challenges, and future directions," *Sensors*, vol. 17, no. 3, p. 565, 2017.

[7] S. MacNamara and G. Lacey, "A smart walker for the frail visually impaired," in *IEEE International Conference on Robotics and Automation*, vol. 2, 2000, pp. 1354–1359.

[8] H. Yu, M. Spenko, and S. Dubowsky, "An adaptive shared control system for an intelligent mobility aid for the elderly," *Autonomous Robots*, vol. 15, no. 1, pp. 53–66, 2003.

[9] J. Paulo, P. Peixoto, and U. J. Nunes, "ISR-AIWALKER: Robotic Walker for Intuitive and Safe Mobility Assistance and Gait Analysis," *IEEE Transactions on Human-Machine Systems*, 2017.

[10] G. J. Lacey and D. Rodriguez-Losada, "The evolution of Guido," *IEEE robotics & automation magazine*, vol. 15, no. 4, pp. 75–83, 2008.

[11] P. Panteleris and A. A. Argyros, "Vision-based SLAM and moving objects tracking for the perceptual support of a smart walker platform," in *European Conference on Computer Vision*, 2014, pp. 407–423.

[12] K. Chaccour, J. Eid, R. Darazi, A. H. el Hassani, and E. Andres, "Multisensor guided walker for visually impaired elderly people," in *International Conference on Advances in Biomedical Engineering (ICABME)*, 2015, pp. 158–161.

[13] S. Boudjit and H. Moungla, "mHealth: WBANs Issues and Challenges," in *Mobile Health*, 2015, pp. 771–790.

[14] S. Zehtabian, S. Khodadadeh, R. Pearlman, B. Willenberg, B. Kim, D. Turgut, L. Bölöni, and E. A. Ross, "Supporting rehabilitation prescription compliance with an IoT-augmented four-legged walker," in *2nd Workshop on AI for Aging, Rehabilitation and Independent Assisted Living (ARIAL'18)*, July 2018.

[15] S. Khodadadeh, S. Zehtabian, J. Guilbe, R. Pearlman, B. Willenberg, B. Kim, E. A. Ross, L. Bölöni, and D. Turgut, "Detecting unsafe use of a four-legged walker using IoT and deep learning," in *IEEE International Conference on Communications (ICC)*, May 2019.

[16] V. Viegas, J. Dias Pereira, O. Postolache, and P. S. Girão, "Monitoring walker assistive devices: a novel approach based on load cells and optical distance measurements," *Sensors*, vol. 18, no. 2, p. 540, 2018.

[17] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1–4.

[18] N. Ortigosa, S. Morillas, and G. Peris-Fajarnés, "Obstacle-free pathway detection by means of depth maps," *Journal of Intelligent & Robotic Systems*, vol. 63, no. 1, pp. 115–129, 2011.

[19] B. Li, X. Zhang, J. P. Muñoz, J. Xiao, X. Rong, and Y. Tian, "Assisting blind people to avoid obstacles: an wearable obstacle stereo feedback system based on 3D detection," in *IEEE IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2015, pp. 2307–2311.