Entity Counting and Tracking Using Microsoft Kinect Depth Maps

Jared Meade, Sheree Enlow, Brian Thomas, Aaron Crandall
School of Electrical Engineering and Computer Science, Washington State University

Introduction

The Palouse Discovery Science Center (PDSC) in Pullman has requested a head counting/tracking system to track the number of patrons coming in and out of their building as well as their corresponding heights (adult or child). Gathering accurate headcount data through traditional surveillance can be intrusive and is not ideal for a family friendly facility or in a (smart) home environment. Our solution utilized an unsupervised learning method and entity tracking algorithms coupled with depth maps from a ceiling mounted Microsoft Kinect V1 above entry ways.

Hardware/Software

Two major components for our research was a Microsoft Kinect V1 infrared camera and an open source driver that allowed us to pull depth map frames in C++ on a Linux machine called libfreenect. Instead of color value being contained in a pixel, depth maps contain a pixel value equal to the distance away from an object in a traditional RGB image.

a. Microsoft Kinect V1

b. Depth image

c. RGB image

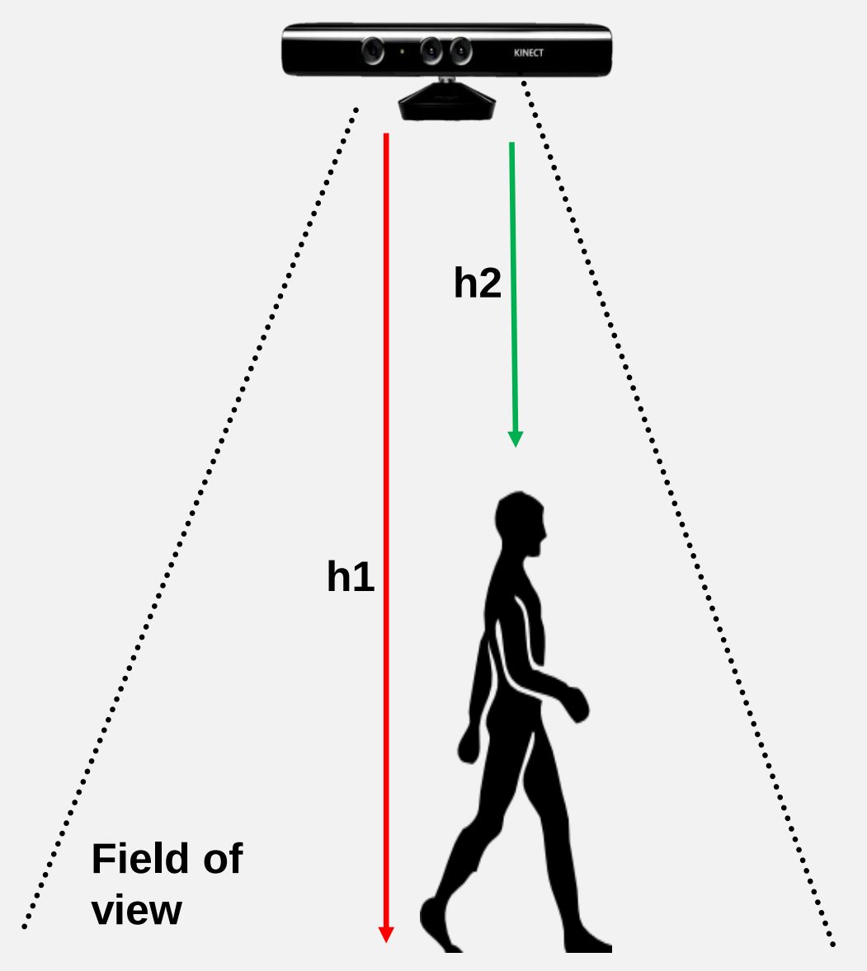
The data received by our Kinect system is shipped to the middleware servers called SHiB's (Smart Home in a Box) through the utilization of the XMPP protocol and the Gloox C++ library. This protocol allows for sensor agents to subscribe and publish to various channels hosted by the SHiB servers.

Robustness Tests

These various cases were tested to determine the robustness of our algorithm and are implemented in our test dataset:

- Individuals from a wide range of heights walking in view
- Many entities in view at once
- Child carried on person's shoulders or back
- Entities wearing hats, backpacks, bags, or purses

Determining Head Height

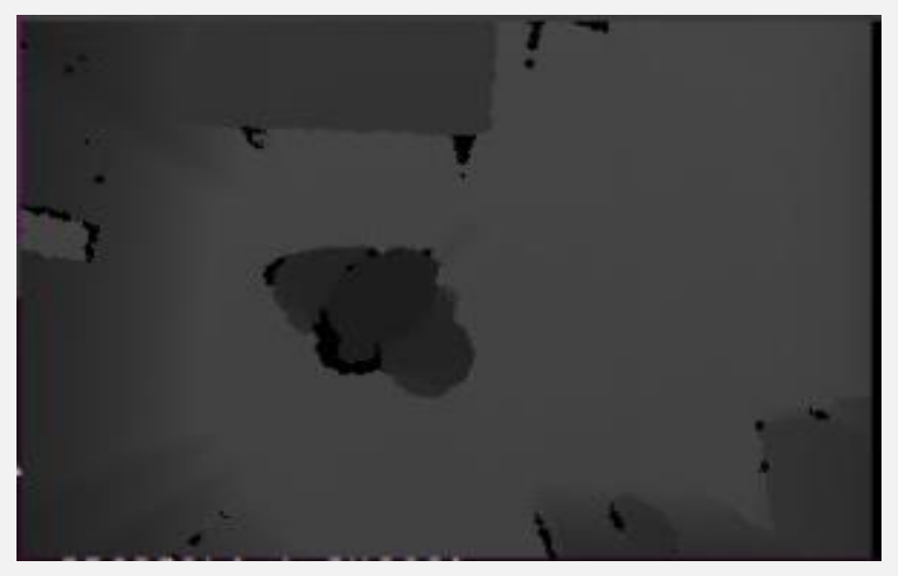


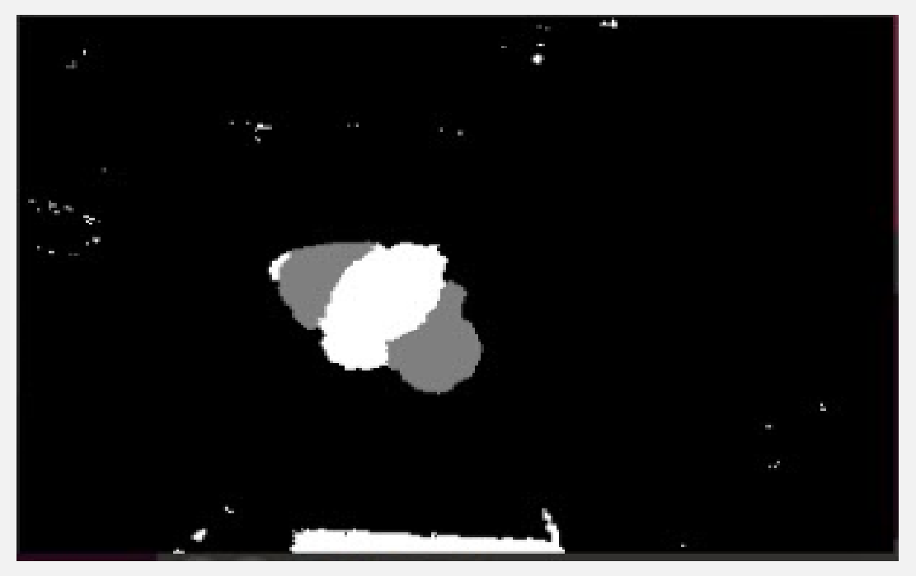
h1-h2 = height. For a more robust solution we created a method of determining the floor distance from a sample of initial images. The first 30 depth images (1 sec worth) are taken and a 5x6 portion of the images centered in the middle is assumed to be the floor and each value is stored. Next, each pixel in 30 more images is iterated through and if the pixel is within one standard deviation from the mean of our stored floor values it is added to the list of floor values. After all 60 initialization images have been processed we find the mean of our floor list and use this value as the constant floor to Kinect distance.

Methods and Algorithms

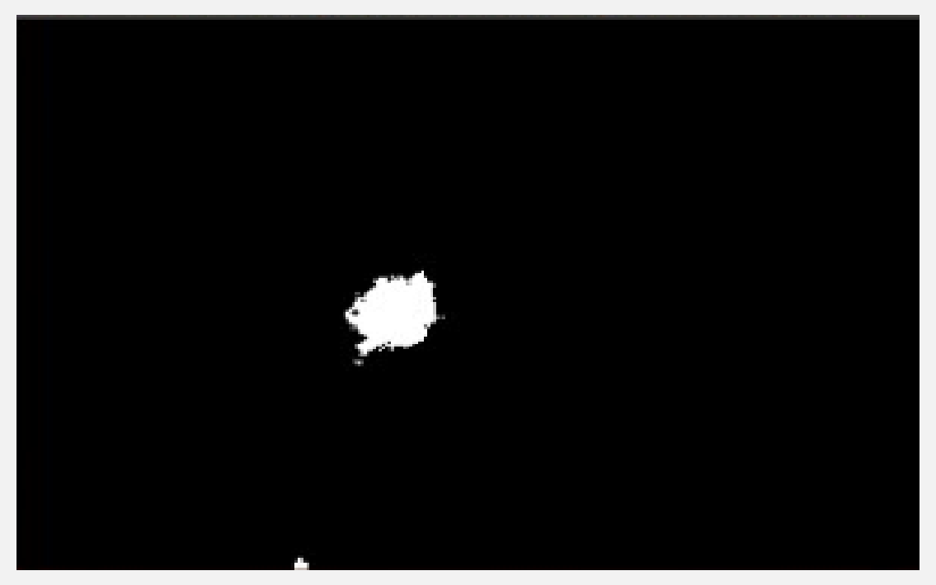
Head Count/WaterFill Algorithm[1]:

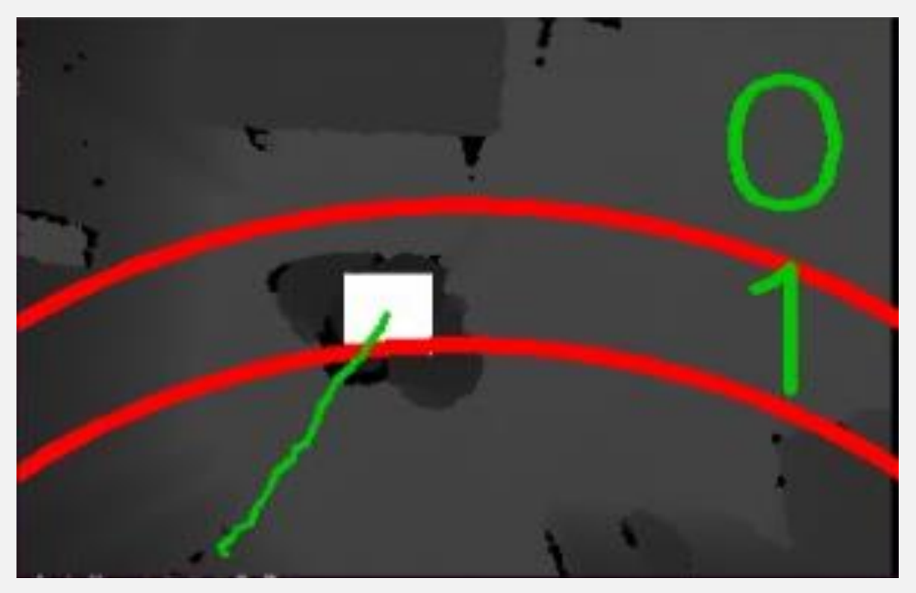
The WaterFill algorithm works as follows, people/heads are identified as the foreground of the image by using a Gaussian mixture model background subtraction method. The statistical nature of this subtractor allows for the WaterFilll algorithm to adapt to a dynamically changing background.


d. Depth image greyscale


e. Foreground/subtracted background

After filtering out objects in the foreground image that have contours (boxed outline over head) with area's smaller then some threshold value, the initial depth map's pixels are all turned to one except for the pixels corresponding to white regions on the filtered foreground region. Now conceptually 'water drops' are dropped and randomly distributed over the filtered depth map in all regions that aren't equal to one and travel in the direction of descent. All of these drops are kept track of in a separate blank image representing areas where 'water' collected (f). This water image is then filtered of noise by eliminating regions that did not reach the threshold of depth. The contour boxes of the remaining water filled regions now correspond to heads and can be seen drawn on image (d) in the final image (g).


f. Greyscale representation of water filled regions



g. Contour box/head

Head Tracking Algorithm:

Our head tracking algorithm determines which contour boxes are the same between frames by producing a predicted path for each head in the current frame. This is path is produced by weighting each of the head's previous movements (x and y's difference) and dividing the sum of these products by the sum of weights in order determine a mean and then adding the mean to the center of the head's current x,y coordinates. Each current head is then matched to the next frame's contour boxes by assigning each one to a current contour box with the shortest Euclidean distance to it's projected path. Other threshold limiting the distance a head can move between frames also helps prevent occlusion errors. With this head tracking algorithm we created a head count system in which a head had to cross two lines in order to increment the in/out counter (seen in figure g).

Results

Test Dataset	In Count	Out Count	Total
Head Tracking Algorithm	64	55	119
Actual Count	66	63	129
Accuracy	96.97%	87.30%	92.25%

Table 1: Accuracy of Head Tracking Algorithm

Entity	1	2	3	4	5	6	7	Average
Actual Height (ft)	3.88	4.60	5.02	5.54	5.33	5.79	6.06	5.18
Measured Height (ft)	4.41	4.58	4.98	5.43	5.63	5.72	5.93	5.24
Difference	0.53	0.03	0.04	0.11	0.30	0.08	0.14	0.17

Table 2: Actual vs. Measured Height of Entities

It should be noted that the test dataset is not representative of an average dataset/day at PDSC and was designed to test the robustness of our system.

Conclusion

Our head tracking system struggled in various edge cases:

- Small children/heads (parameters can be tuned)
- When a person was carried on another's shoulder/back
- When arms were raised above one's head
- Person walked out of left or right edge of view before crossing line

Although, this test set was meant to see what will break our algorithm we will have to later refine it to better handle these various edge cases because they could still occur at PDSC.

Moving Forward

- Deploy a visually appealing secure setup of the Kinect system at PDSC
- Publish headcount and height data on CASAS servers
- Create a visual display of the Kinect data at PDSC
- Annotate test dataset
- Optimize code to better handle child patrons at PDSC

References

[1]Zhang, X., Yan, J., Feng, S., Lei, Z., Yi, D., and Li1, S. Z., 2012, Water Filling: Unsupervised People Counting via Vertical Kinect Sensor, *IEEE*.

Acknowledgments

We would like to thank the National Institutes of Health & the National Institute of Aging and their generous support of the GSUR program under grant 1R25AG046114.