# Human Detection Using Depth Information by Kinect

Lu Xia, Chia-Chih Chen and J. K. Aggarwal
The University of Texas at Austin
Department of Electrical and Computer Engineering
{xialu|ccchen|aggarwaljk}@mail.utexas.edu

## Abstract

*Conventional human detection is mostly done in images taken by visible-light cameras. These methods imitate the detection process that human use. They use features based on gradients, such as histograms of oriented gradients (HOG), or extract interest points in the image, such as scale-invariant feature transform (SIFT), etc. In this paper, we present a novel human detection method using depth information taken by the Kinect for Xbox 360. We propose a model based approach, which detects humans using a 2-D head contour model and a 3-D head surface model. We propose a segmentation scheme to segment the human from his/her surroundings and extract the whole contours of the figure based on our detection point. We also explore the tracking algorithm based on our detection result. The methods are tested on our database taken by the Kinect in our lab and present superior results.*

## 1. Introduction

Detecting human in images or videos is a challenging problem due to variations in pose, clothing, lighting conditions and complexity of the backgrounds. There has been much research in the past few years in human detection and various methods are proposed [1, 2, 6, 13]. Most of the research is based on images taken by visible-light cameras, which is a natural way to do it just as what human eyes perform. Some methods involve statistical training based on local features, e.g. gradient-based features such as HOG [1], EOH [8], and some involve extracting interest points in the image, such as scale-invariant feature transform (SIFT) [9], etc. Although lots of reports showed that these methods can provide highly accurate human detection results, RGB image based methods encounter difficulties in perceiving the shapes of the human subjects with articulated poses or when the background is cluttered. These will result in the drop of accuracy or the increase of computational cost. Depth information is an important cue when human recognize objects because the objects may not have consistent color and texture but must occupy an integrated region in space. There has been research using range image for object recognition or modeling in the past few decades [12, 14]. Range images have several advantages over 2D intensity images: range images are robust to the change in color and illumination. Also, range images are simple representations of 3D information. However, earlier range sensors were expensive and difficult to use in human environments because of lasers. Now, Microsoft has launched the Kinect, which is cheap and very easy to use. Also, it does not have the disadvantages of laser so it can be used in human environment and facilitate the research in human detection, tracking and activity analysis.

In recent years, there is a body of research on the problem of human parts detection, pose estimation and tracking from 3D data. Earlier research used stereo cameras to estimate human poses or perform human tracking [3, 4, 15]. In the past few years, a part of the research has focused on the use of time-of-flight range cameras (TOF). Many algorithms have been proposed to address the problem of pose estimation and motion capture from range images [5, 7, 11, 16]. Ganapathi et al. [5] present a filtering algorithm to track human poses using a stream of depth images captured by a TOF camera. Jain et al. [7] present a model based approach for estimating human poses by fusing depth and RGB color data. Recently, there have been several works on human/parts detection using TOF cameras. Plagemann et al. [10] use a novel interest point detector to solve the problem of detection and identifying body parts in depth images. Ikemura et al. [6] proposed a window-based human detection method using relational depth similarity features based on depth information.

In this paper, we present a novel model based method for human detection from depth images. Our method detects people using depth information obtained by Kinect in indoor environments. We detect people using a 2-stage head detection process, which includes a 2D edge detector and a 3D shape detector to utilize both the edge information and the relational depth change information in the depth image. We also propose a segmentation method to segment the figure from the background objects that attached to it and extract the overall contour of the subject accurately. The method is evaluated on a 3D dataset taken in our lab using the Kinect for Xbox 360 and achieves excellent results.

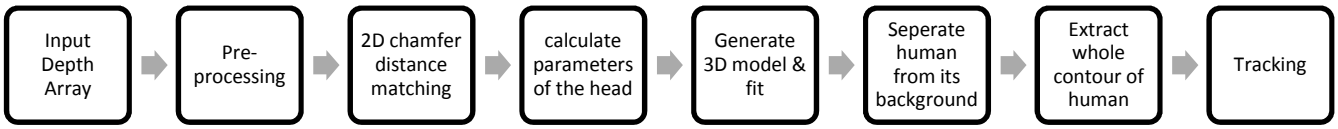



Fig. 1. Overview of our human detection method.

Our algorithm utilizes depth information only. It can also be combined with traditional gradient based approaches to give faster and more accurate detection. The detection algorithm can also serve as an initial step of the research on pose estimation, tracking, or activity recognition using depth information.

Our paper is organized as follows. Section 2 gives an overview of our method. Section 3 describes the 2D chamfer distance match. Section 4 describes 3D model fitting. Section 5 gives detail on extracting the whole contour of people. Section 6 gives our preliminary research on tracking. Section 7 discusses the experimental results. Section 8 concludes the paper and gives possible directions for future research.

## 2. Overview of the method

This section provides an overview of the major steps in our method, which is summarized in Fig. 1. Implementation details are presented in Section 3 through Section 6.

Given an input depth array, we first reduce noise and smooth the array for later process. We use a 2-stage head detection process to locate the people. We first explore the boundary information embedded in the depth array to locate the candidate regions that may indicate the appearance of people. The algorithm used here is 2D chamfer distance matching. It scans across the whole image and gives the possible regions that may contain people. We examine each of these regions using a 3D head model, which utilizes the relational depth information of the array for verification. We extract the parameters of the head from the depth array and use the parameter to build a 3D head model. Then we match the 3D model against all the detected regions to make a final estimation. We also develop a region growing algorithm to find the entire body of the person and extract his/her whole body contour. To do it accurately, we propose a segmentation method to segment the body and the objects that are connected to it. Also, we show some preliminary research on tracking using our detection result.

## 3. 2D chamfer distance matching

### 3.1. Preprocessing

To prepare the data for our processing, some basic pre-processing is needed. In the depth image taken by the Kinect, all the points that the sensor is not able to measure depth are offset to 0 in the output array. We regard it as a kind of noise. To avoid its interference, we want to recover its true depth value. We suppose that the space is continuous, and the missing point is more likely to have a similar depth value to its neighbors. With this assumption, we regard all the 0 pixels as vacant and need to be filled. We use the nearest neighbor interpolation algorithm to fill these pixels and get a depth array that has meaningful values in all the pixels. Then we use median filter with a 4×4 window on the depth array to make the data smooth.

### 3.2. 2D chamfer distance matching

The first stage of the method is to use the edge information embedded in the depth array to locate the possible regions that may indicate the appearance of a person. It is a rough scanning approach in that we need to have a rough detection result with a false negative rate as low as possible but may have a comparatively high false positive rate to provide to the next stage. We use 2D chamfer distance matching in this stage for quick processing. Also, chamfer distance matching is a good 2D shape matching algorithm that is invariant to scale, and it utilizes the edge information in the depth array which means the boundary of all the objects in the scene. We use Canny edge detector to find all edges in the depth array. To reduce calculation and reduce the disturbance from the surrounding irregular objects, we eliminate all the edges whose sizes are smaller than a certain threshold. (Here, the size of the edge is determined by the pixels it contained.) Results of chamfer distance matching are shown in Fig. 2.

We use a binary head template shown in Fig. 2(d) and match the template to the resulted edge image. To increase the efficiency, a distance transform is calculated before the

(a) (b) (c) (d) (e)

Fig. 2. Intermediate results of 2D Chamfer match. (a) shows the depth array after noise reduction. (b) gives the binary edge image calculated using Canny edge detector and then eliminate small edges. (c) shows the distance map generated from edge image (b). Match the binary head template (d) to (c) gives the head detection result (e). Yellow dots indicate the detected locations.

matching process. This results in a distance map of the edge image, where pixels contain the distances to the closest data pixels in the edge image. Matching consists of translating and positioning the template at various locations of the distance map; the matching measure is determined by the pixel values of the distance image which lie under the data pixels of the transformed template. The lower these values are, the better the match between image and template at this location. If the distance value lies below a certain threshold, the target object is considered detected at this place, which means that a head like object is found here. We use the phrase "head like object" here because the object we detected may not be a real head because we used a high threshold here to guarantee a very low false negative rate. Whether this object is actually a head we will decide at the next stage. It is usually the case that the person in the scene is likely to appear at any depth, which means the head size will change according to the depth. To make the algorithm invariant to scale, we generate an image pyramid with the original image at the bottom; each image is subsampled to generate the next image at the higher level. The subsample rate we used here is 3/4, and the number of the level of the pyramid depends on the scene. If the scene contains a larger range of depth, a larger number of levels is needed. This template is able to find head of the person in all poses and views. If the person is in a horizontal position or is upside down, it is easily settled by rotating the template and running the same detection process.

The result of all the steps in this stage is shown in Fig. 2.

## 4. 3D model fitting

Now we are going to examine all the regions that are detected by the 2D chamfer matching algorithm.

### 4.1. Compute the parameters of the head

In order to generate the 3D model to fit on the depth array, we must know the true parameter of the head that appears in the detected region. To estimate this, we conduct an experiment and get the regression result for the depth of the head and its height, shown in Fig.3.

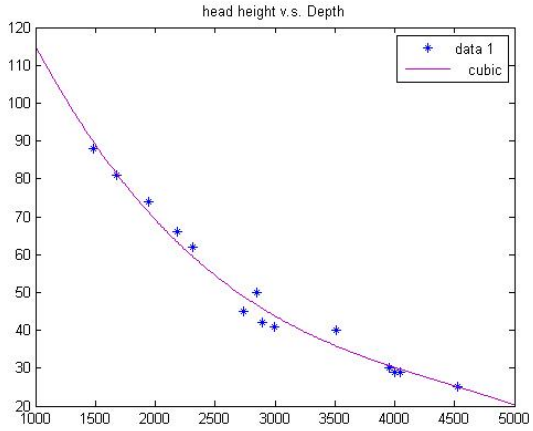



Fig. 3 Regression result of head height and depth.

The cubic equation we get is:

$$-y = p_1 \cdot x^3 + p_2 \cdot x^2 + p_3 \cdot x + p_4 \qquad (1)$$

Here,

$$\begin{cases} p_1 = -1.3835 \times 10^{-9} \\ p_2 = 1.8435 \times 10^{-5} \\ p_3 = -0.091403 \\ p_4 = 189.38 \end{cases}$$

From the detection result of 2D chamfer matching, we can get the depth of the head like object from the original depth array. By equation (1), we calculate the standard height of the head in this depth. Then we search for the head

within a certain range that is defined by the standard height of the head:

$$R = 1.33h / 2. \tag{2}$$

Here, h is the height of the head calculated from equation (1), R is the search radius.

Next, we search for the head within a circular region defined by radius R in the edge image. If there is a circular edge in this region that satisfied all the constraints, e.g. size pass a certain threshold, it is decided that a head is detected. The next thing to do is to find the true radius of the head. It happened to be that the distance map we calculated at 2D chamfer matching stage can be used to estimate the radius of the head. Recall that the pixels in the distance map contain the distances from this pixel to the closest data pixels in the edge image, considering the head is a circular like shape, the value of the center of the head on the distance map is just an approximation of the radius of the head. So we can take this directly as our estimation of the true radius of the head $R_t$.

### 4.2. Generate 3D model

Considering the calculation complexity of 3D model fitting is comparatively high, we want the model to be view invariant so that we don't have to use several different models or rotate the model and run several times. The model should generalize the characteristics of the head from all views: front, back, side and also higher and lower views when the sensor is placed higher or lower or when the person is higher or lower. To meet these constraints and make it the simplest, we use a hemisphere as the 3D head model.

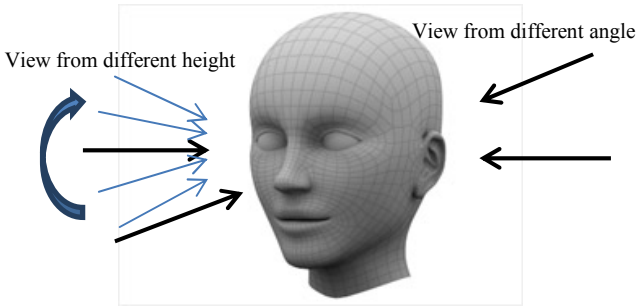
### 4.3. Fitting

Next, we fit the model onto the regions detected from previous steps. We extract a circular region CR with radius $R_t$ around the detect center and normalize its depth:

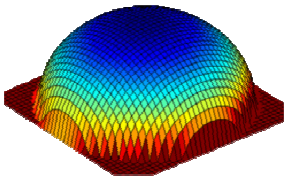
$$depth\_n(i,j) = depth(i,j) - \min_{i,j}(depth(i,j))$$
$$i, j \in CR \tag{3}$$

Here, depth(i,j) is the depth value of pixel (i, j) in the depth array. depth_n(i,j) is the normalized depth value of pixel (i,j). Then we calculate the square error between the circular region and the 3D model:

$$Er = \sum_{i,j \in CR} |depth\_n(i,j) - template(i,j)|^2 \tag{4}$$




(a)



(b)

Fig.4. 3D head model. (a) illustrates the demands of the head model: the model should invariant to different views. (b) shows the hemisphere model we used as the 3D head model

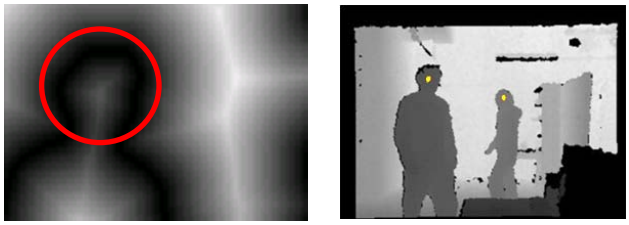
We use a threshold to decide whether the region is actually a head. Fig. 5. illustrates some of the steps in this stage, and shows the result of the 3D matching.

## 5. Extract contours

We extract the overall contour of the person so that we may track his/her hands and feet and recognize the activity. In an RGB image, despite the person is standing on the ground, it is less a problem to detect the boundary between the feet and the ground plane using gradient feature. However, in a depth array, the values at the person's feet and the local ground plane are the same. Therefore, it is not feasible to compute humans' whole body contours from a depth array using regular edge detectors. The same applies when the person touches any other object that is partially in the same depth with the person. To resolve this issue, we take advantages of the fact that persons' feet generally appear upright in a depth array regardless of the posture. We use the filter response of

$$F = [1,1,1,-1,-1,-1]^T \tag{5}$$

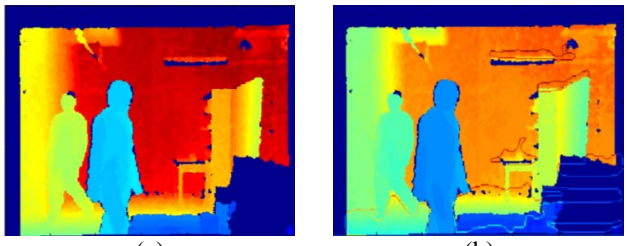
to extract the boundary between the persons and the ground.



(a) (b)

Fig. 5 (a) illustrates the process of estimating the true parameter of the head from the distance map. Input of the 3D model fitting is the output of the 2D chamfer matching in Fig. 2(e). Output of 3D model fitting is shown in (b). Yellow dots indicate the center of the head detected.

The thresholded filter response delineates the planar areas that are parallel to the floor. The edges extracted by F filter response together with the original depth array are added together as the input to our region growing algorithm. Fig. 6 shows an example of the filter response. (The color distributions of in both images are a little different because of we scale the array for display, the corresponding values are the same.)



(a) (b)

Fig. 6. (a) Original depth array. Some parts of the body are merged with the ground plane and wall. (b) The input depth array to the region growing algorithm. The ground plane is delineated by the thresholded *F* filter response. The edges along the feet well separate the persons from the floor.

We develop a region growing algorithm to extract the whole body contours from the processed depth array. It is assumed that the depth values on the surface of a human object are continuous and vary only within a specific range. The algorithm starts with a seed location, which is the centroid of the region detected by 3-D model fitting. The rule for growing a region is based on the similarity between the region and its neighboring pixels. The similarity between two pixels *x* and *y* in the depth array is defined as:

$$S(x,y) = |depth(x) - depth(y)| \quad (6)$$

| Start region growing until similarity between the region and neighboring pixels is higher than a threshold |
|---|
| i. Initialize: region = seed<br>ii.<br>(1) Find all neighboring pixels of the region<br>(2) Measure the similarity of the pixels and the region (Eq.7) s1, s2, … and sort the pixels according to the similarity.<br>(3) If smin < threshold<br>(3.1). Add the pixel with the highest similarity to the region.<br>(3.2). Calculate the new mean depth of the region.<br>(3.3). Repeat (1)-(3)<br>else<br>algorithm terminate<br>iii. Return the region |

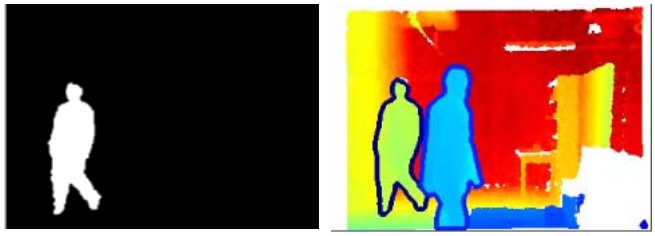
Table 1. Region Growing Algorithm

Here, *S* is similarity and *depth*() returns the depth value of the pixel. The depth of a region is defined by the mean depth of all the pixels in that region:

$$depth(R) = \frac{1}{N}\sum_{i \in R}(depth(i)) \quad (7)$$

The pseudocode of our algorithm is summarized in Table.1

The results of our region growing algorithm are shown in Fig. 7.



(a) (b)

Fig. 7. (a) Result of our region growing algorithm. (b) The extracted whole body contours are superimposed on the depth map.

## 6. Tracking

Finally, we give preliminary results on tracking using depth information based on our detection result. Tracking in RGB image is usually based on color, the assumption is that the color of the same object in different time frames should be similar. But in depth images we don't have such color information. What we have is the 3D space information of the objects, so that we can measure the

movements of the objects in a 3D space. Our tracking algorithm is based on the movements of the objects. We assume that the coordinates and speed of the same objects in neighboring frames change smoothly, i.e. there should not be big jumps in coordinates or speed. First, we find the center of the detected blob. Then we calculate the 3D coordinates and speed of the persons in each frame. The coordinates are given in the depth array directly; the speed is calculated from the coordinates of neighboring frames. We define a energy score of the changes in space and speed:

$$E = (c - c_0)^2 + (v - v_0)^2 \quad (8)$$

Here, E is the energy score, c is the coordinates of the person in the current frame and $c_0$ is the coordinates of the person in the last frame. v is the speed of the person in the current frame and $v_0$ is the speed of the person in the last frame.

In the first frame, we label the person in turn according to the detection order. For the subsequent frames, we try all the possible matches of those people and take the one with the smallest energy score to be the solution. Special cases need to be handled when the total number of people in the frame changes, like when there are people get out of the scene or new persons join in.

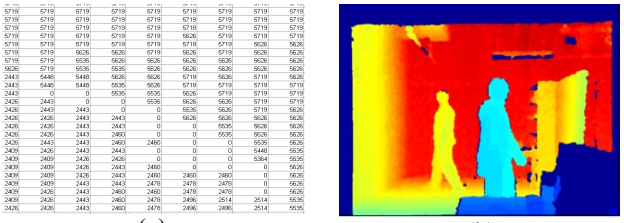
## 7. Experimental results

In this section we describe the experiments performed to evaluate our method. We show both qualitative and quantitative results on our datasets and compare our approach with a window-based human detection algorithm [6].

### 7.1. Dataset

We evaluate our method using a sequence of depth arrays taken by the Kinect for XBOX 360 in indoor environment. We took the sequence in our lab with at most two persons presented in the scene. There are tables, chairs, shelves, computers, an overhead lamp and so on presented in the scene. The people have a variety of poses, and they have interaction with others or the surrounding objects. There are 98 frames in the test set and the frame rate is 0.4s/frame. The size of the depth array is 1200 ×900 and the resolution is about 10mm.

To better illustrate our image frames, we scale the depth array and plot using color map 'JET' as in Fig. 8. The depth is measured in millimeters and the points that failed to be measured are offset to 0 (which usually happen in the irregular edge of objects or surfaces that do not reflect infrared rays well, e.g. polyporous materials and when the



(a) (b)

Fig.8. (a) a patch of the depth array (b) the depth array showed using color map JET

objects are moving fast). 0 value in depth array corresponds to the dark blue in the image in (b).

### 7.2. Experimental results

Our detection method performs well on our indoor dataset. Fig. 9 shows some of the results of our algorithm.

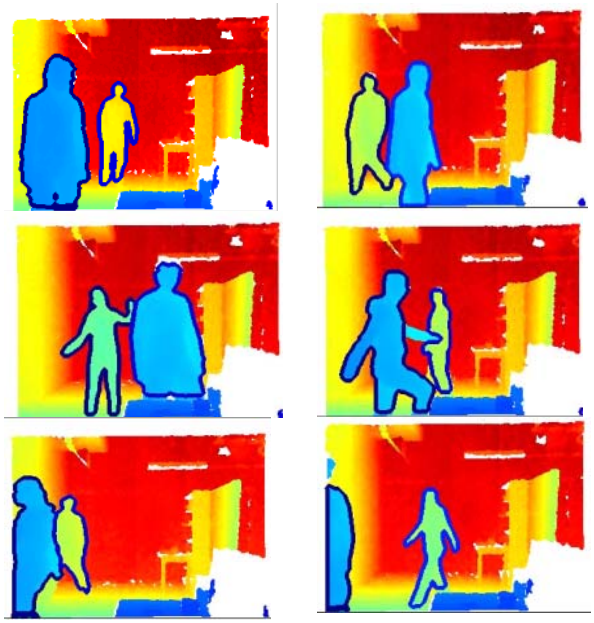


Fig. 9. Examples of the human detection result.

Fig. 10 shows the preliminary tracking results based on our detection result. 15 consecutive frames are shown, which includes two people walking past each other, one person gets occluded and appears again. The detection method performs well in most cases. We do not have any FP instances but only a few FN detections. It happened when the person's head is occluded by another person or half of the body is out of the frame, as shown in Fig. 11.

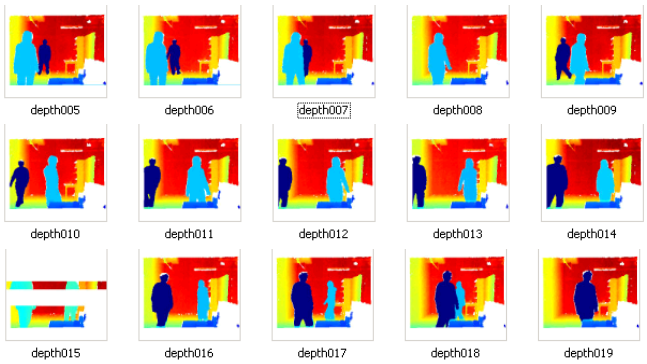
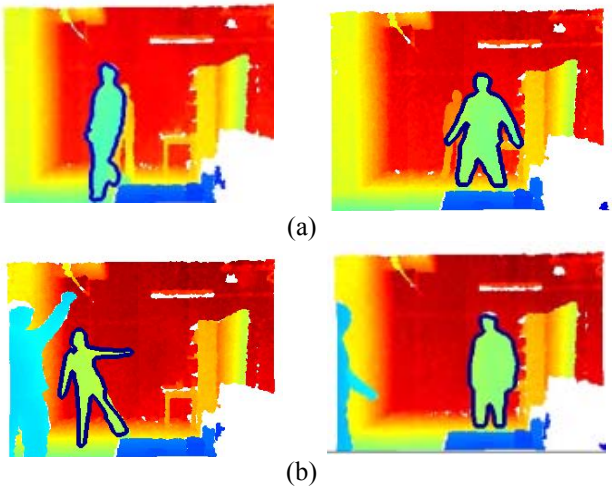



Fig. 10 Tracking result.



(a)

(b)

Fig. 11. FN examples. (a) The person behind is not detected. (b) The person on the left edge of the image is not detected.

We evaluate our method with different accuracy metrics, as shown in Table 2. The precision, recall and accuracy are defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

We compare our algorithm with a recent window-based algorithm which uses the relational depth similarity features for classification [6].

| TP | TN | FP | FN |
|---|---|---|---|
| 169 | 266 | 0 | 7 |
| **Precision** | | **Recall** | |
| 100% | | 96.0% | |
| **Accuracy** | | | |
| 98.4% | | | |

Table 2. Accuracy of our algorithm.

| | Precision | Recall | Accuracy |
|---|---|---|---|
| **Our** | 100% | 96.0% | 98.4% |
| **Ikemura** | 90.0% | 32.9% | 85.8% |

Table 3. Comparison of performance

The original method uses TOF data. We perform an additive preprocessing step and implement the scheme on our Kinect data. Table 3 shows the comparison of performances of both methods. There are about 0 to 500 windows extracted from each frame and we subsample them and use the odd number of frames for training and even number of frames for testing. There are 770 positive examples and 2922 negative examples in the training set and 738 positive examples and 2930 negative examples in the test set. Note that the unit here is window not frame.

From Table 3 we can see that our algorithm outperforms Ikumura's algorithm on this dataset. The main reason is that Ikumura's window-based algorithm is better at handling the instances when the people in the frame are in an upright position. However, people in this dataset are presented in all kinds of postures and rotations. The recall of Ikemura's algorithm is low because it is a window-based method which has a large false negative rate. The high false negative rate actually does not deteriorate his performance because the same person would appear in a lot of scanning windows. The algorithm will produce true positive when the person is well centered in the window, and it will classify the rest of the windows which the persons are not in the center as negative frames. And that is the cause of the high false negative rate. But the person in the image is successfully detected in this case.

## 8. Conclusions

In this paper, we propose a human detection method that uses the depth images obtained from the Kinect for Xbox360. The experimental results show that our algorithm can effectively detect the persons in all poses and appearances from the depth array, and it provides an accurate estimation of the whole body contour of the person. In addition, we explore a tracking algorithm based on our detection results. The approach can be applied in multiple tasks such as object segmentation, human detection,

tracking, and activity analysis. And the algorithm is generally applicable to depth images acquired by other types of range sensors.

The advantages of our method are briefly described in the followings. First, the method can easily adjust to new datasets, no training is needed. Second, the algorithm uses a two layer detection process with 2D chamfer matching in the first layer which largely reduces computational cost. Third, we do not assume person's pose for accurate detection. The limitation is that this algorithm has high dependency on accurate head detection, which implies that if the head is occluded or if the person is wearing a strange shape hat, it probably will not be detected, but this problem can be handled when we extend the head detector to other parts of the body, e.g. combine with hand detector or central body detector.

In the future, we plan to develop body parts detectors, continue the research on tracking and perform activity analysis based on the contour we extracted.

## 9. References


[1] N. Dalal and B. Trigges.: Histograms of oriented gradients for human detection. CVPR, 1 (2005) 886-893.

[2] N. Dalal, B. Triggs, C. Schmid.: Human detection using oriented histograms of flow and appearance, in: European Conference on Computer Vision, Graz, Austria, May 7–13, 2006

[3] T. Darrell, G. Gordon, J. Woodfill, and M. Harville, "Integrated Person Tracking using Stereo, Color, and Pattern Detection," Proceedings IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, June 1998

[4] D. Demirdjian, T. Darrell: 3-D Articulated Pose Tracking for Untethered Diectic Reference. ICMI 2002: 267-272

[5] V. Ganapathi, C. Plagemann, D. Koller, S. Thrun. Real time motion capture using a single time-of-flight camera. Proceedings of CVPR 2010. pp.755~762

[6] S. Ikemura, H. Fujiyoshi.: Real-Time Human Detection using Relational Depth Similarity Features. ACCV 2010, Lecture Notes in Computer Science, 2011, Volume 6495/2011, 25-38

[7] HP. Jain and A. Subramanian. Real-time upper-body human pose estimation using a depth camera. In HP Technical Reports, HPL-2010-190, 2010

[8] K. Levi and Y. Weiss. Learning object detection from a small number of examples: the importance of good features. CVPR 2(2004) 53-60

[9] DG. Lowe.: Object Recognition from Local Scale-Invariant Features. Proceedings of the International Conference on Computer Vision. 2 (1999). pp.?1150–1157

[10] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. Realtime identification and localization of body parts from depth images. In IEEE Int. Conference on Robotics and Automation (ICRA), Anchorage, Alaska, USA, 2010

[11] J. Rodgers, D. Anguelov, H.-C. Pang, and D. Koller. Object pose detection in range scan data. In Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2006

[12] B. Sabata, F. Arman, J. K. Aggarwal.: Segmentation of 3D Range Images Using Pyramidal Data Structures, CVGIP: Image Understanding, Vol. 57, No. 3, 1993, pp. 373-387

[13] W. Schwartz, A. Kembhavi, D. Harwood, L. Davis,: Human detection using partial least squares analysis. In: ICCV (2009)

[14] BC. Vemuri, A. Mitiche, J. K. Aggarwal.: Curvature-based Representation of Objects from Range Data, Image and Vision Computing, Vol. 4, No. 2, pp. 107-114, May 1986

[15] HD. Yang, S.W. Lee: Reconstruction of 3D human body pose from stereo image sequences based on top-down learning. Pattern Recognition 40(11): 3120-3131 (2007)

[16] Y. Zhu, B. Dariush, and K. Fujimura. Controlled human pose estimation from depth image streams. Proc. CVPRWorkshop on TOF Computer Vision, June 2008