Fast Human Detection for Indoor Mobile Robots Using Depth Images

Benjamin Choi[1], Çetin Meriçli[1], Joydeep Biswas[2], and Manuela Veloso[1]

***Abstract*— A human detection algorithm running on an indoor mobile robot has to address challenges including occlusions due to cluttered environments, changing backgrounds due to the robot's motion, and limited on-board computational resources. We introduce a fast human detection algorithm for mobile robots equipped with depth cameras. First, we segment the raw depth image using a graph-based segmentation algorithm. Next, we apply a set of parameterized heuristics to filter and merge the segmented regions to obtain a set of candidates. Finally, we compute a Histogram of Oriented Depth (HOD) descriptor for each candidate, and test for human presence with a linear SVM. We experimentally evaluate our approach on a publicly available dataset of humans in an open area as well as our own dataset of humans in a cluttered cafe environment. Our algorithm performs comparably well on a single CPU core against another HOD-based algorithm that runs on a GPU even when the number of training examples is decreased by half. We discuss the impact of the number of training examples on performance, and demonstrate that our approach is able to detect humans in different postures (*e.g.* standing, walking, sitting) and with occlusions.**

I. Introduction

Human detectors running on autonomous indoor mobile robots encounter several challenges. The perceived motion of the environment due to the motion of the robot itself makes it difficult to isolate moving humans. Humans are observed in a variety of postures depending on whether they are sitting or standing, the type of furniture they are sitting on, and the angle from which they are viewed. Furthermore, humans interacting with the robot may only be partially visible to the robot. Mobile robots possess limited computational resources which are shared among various processes required for autonomous operation, so the human detector must have a small computational footprint.

We contribute a depth image based human detection algorithm that addresses these challenges. The algorithm successfully detects humans under challenging conditions such as occlusion, clutter, and different postures (see Fig. 1). Our algorithm first segments each observed depth image into "regions" corresponding to geometrically distinct objects. These regions are then filtered into a set of "candidates" based on a series of parametric heuristics. The candidates are then used to compute histograms of oriented depth (HOD) [1], and the HOD descriptors are classified as humans or not humans by a trained linear support vector machine (SVM). The contributions of this work include:

[1]B. Choi, Ç. Meriçli, and M. Veloso are with the Computer Science Department, Carnegie Mellon University 5000 Forbes Ave., Pittsburgh, PA 15213, United States {benchoi,cetin,veloso}@cmu.edu

[2]J. Biswas is with the Robotics Institute, Carnegie Mellon University 5000 Forbes Ave., Pittsburgh, PA 15213, United States joydeep@cmu.edu

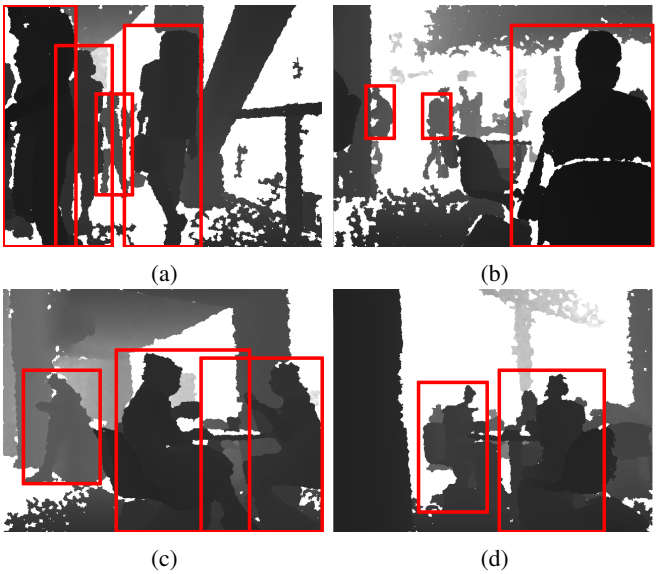


Fig. 1: Our proposed algorithm detecting a) partially occluded people, b) people at different distances, c) people with different postures, and d) people amidst clutter.

- A novel adaptation of a graph-based segmentation algorithm combined with randomized subsampling for fast depth image segmentation;
- The application of a set of parameterized heuristics to further reduce the number of candidate segments for classification; and
- A performance evaluation of the proposed approach on two different datasets (one a public dataset, and one of our own from a cluttered environment) with an investigation of the impact of the size of the training set on performance.

In the remainder of this paper, we first discuss related work, then detail our approach, and finally present empirical results of our algorithm running on a standard dataset as well as our custom dataset collected from a very crowded area.

II. Related Work

Human detection is an extensively studied subject that was confined mostly to color images. With the advent of affordable RGB-D cameras such as the Microsoft Kinect and Asus Xtion, depth image based human detection has recently attracted attention in the robotics community.

Two widely adopted approaches to human detection are detection of the whole body (e.g. [2]) and detection by parts (e.g. [3]). In this section we focus on approaches based on whole-body detection, and briefly discuss the related work in the literature.

The common method of whole-body human detection consists of determining regions in the image that may contain humans, and then testing these candidate regions for human presence using a classifier trained in a supervised manner. A naive way of determining candidate regions in an image is to slide a fixed-size window over the entire image, and treat every region of that window size as a candidate. This approach was proposed to detect pedestrians in RGB images [2]. While covering the entire image and testing every possible candidate, performing a brute-force search over the entire image is computationally expensive. In scenarios where humans may be seen at a range of distances, it is necessary to search candidate windows of different sizes, further increasing the computational cost.

Various methods have been proposed to improve speed over the brute-force sliding window search for extracting candidate regions. Zhu *et al.* utilize salient features in the image and only test the regions containing such features [4], allowing significant speed increases. Spinello and Arras test only candidate windows at depths corresponding to proportions likely to fit a human in metric space [1]. Their algorithm still needs to run on a GPU to process the incoming depth image stream at full frame rate. Spinello and Siegwart utilize clustered laser scan readings to test only the relevant areas in an RGB image, allowing them to greatly reduce the number of candidates tested [5]. Xu and Fujimura use a segmentation based on connected components of depth images to identify foreground objects [6]. Zhou and Hoang use background subtraction to identify regions which can be tested against a codebook to classify them as human or non-human [7]. Shi and Malik present an algorithm that builds weighted graphs for different color channels, and then uses normalized-cuts to segment a given image [8]. Felzenszwalb and Huttenlocher present a similar segmentation algorithm that has a better asymptotic running time [9]. Pulli and Pietikäinen present a method for segmenting depth images using normals [10]. We utilize a graph-based segmentation algorithm that uses a combination of the depth values and surface normals to identify the candidate regions in the image.

Once a candidate region is extracted, the common approach is to compute a feature descriptor for the candidate, and then classify the computed descriptor to decide whether the region contains a human or not. The most widely used descriptor for human detecion is the Histogram of Oriented Gradients (HOG) descriptor, proposed by Dalal and Triggs [2]. The HOG descriptor divides a given image region into a number of cells, and then computes a histogram of edge orientations for each cell using gradients of the pixels in that cell. Recently, Spinello and Arras proposed a variant of the original HOG descriptor adapted to depth images called Histograms of Oriented Depth (HOD) [1], where they compute gradients using the depth values instead of color information. We also use an HOD descriptor, combined with our graph-based segmentation algorithm to effectively process images captured by a moving camera on a mobile robot traversing multiple types of surfaces imposing different perturbations on the captured images.

III. Approach

We use the Microsoft Kinect depth camera for human detection. Although the Kinect provides both color as well as depth images, we only use depth images for human detection. The Kinect provides depth images of resolution 640×480 at a rate of 30Hz. Our algorithm processes each depth image D individually to detect the humans in it. The raw pixel values in the depth images ("raw depth") obtained from the Kinect sensor do not correspond proportionally to physical units ("metric depth"). The metric depth $D'_{x,y}$ of a particular raw depth $D_{x,y}$ (where x, y denote the pixel location in the image) is computed with

$$D'_{x,y} = \frac{1}{a + b \cdot D_{x,y}} \tag{1}$$

where a and b are intrinsic parameters of the Kinect sensor which are determined through calibration. Due to noise and the hardware limitations of the Kinect sensor, some raw pixels may have invalid values that do not translate to a positive metric depth.

To calculate the 3D point $p = \langle p_x, p_y, p_z \rangle$ (in the reference frame of the Kinect sensor) corresponding to the metric depth $D'_{x,y}$ we use

$$p_x = D'_{x,y}\Big(\frac{y}{w-1} - 0.5\Big)\tan\Big(\frac{f_h}{2}\Big) \tag{2}$$

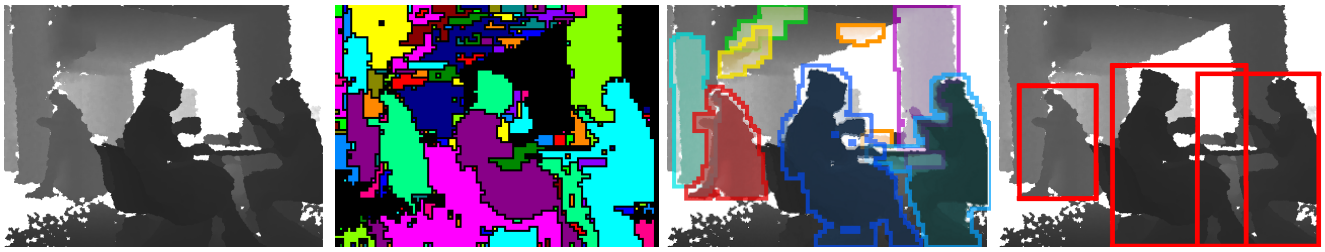
$$p_y = D'_{x,y}\Big(\frac{x}{h-1} - 0.5\Big)\tan\Big(\frac{f_v}{2}\Big) \tag{3}$$

$$p_z = D'_{x,y} \tag{4}$$

where f_h is the horizontal field of view and f_v is the vertical field of view of the camera, and w and h are the image width and height in pixels (for the experiments in this paper, we used $w = 640$ and $h = 480$).

Our human detection algorithm takes a depth image D and outputs a set of detected humans H in three steps:

1) **Depth image segmentation**: This step accepts a depth image D and returns R, a set of regions corresponding to distinct areas in the depth image. This is achieved through sampling points on D, calculating their 3D position, and constructing the graphs G_{depth} and G_{normal}. In G_{depth}, the edge weights represent differences in depth between neighboring points while in G_{normal}, they represent the angle between surface normals of neighboring points. G_{depth} and G_{normal} are then segmented to group the points into regions.
2) **Region filtering and merging**: This step takes a set of regions R and returns a set of candidates C, where each candidate is a smaller depth image constructed from the parts of D corresponding to a region or several merged regions. This step uses parametric heuristic rules to reject regions of infeasible dimensions. The remaining regions are merged based on their location. The parts of D corresponding to these combined regions are then extracted to form the candidates.
3) **Candidate classification**: This step takes a set of candidates C and returns H, the set of humans ($H \subseteq$



(a) The raw depth image (b) Regions after segmentation (c) Candidates after filtering and merging (d) Detected humans

Fig. 2: The steps of the detection process in our approach. The image (b) shows the regions separated by the segmentation step in different colors (best viewed in color).

C). For each candidate, first the HOD descriptor is computed. The computed descriptors are then classified using an SVM. Finally, the set of candidates that are classified as humans is returned.

Fig. 2 depicts the detection steps in our algorithm. In the remainder of the section, we describe each step in detail.

A. Depth Image Segmentation

The purpose of this step is to obtain R, a set of regions corresponding to distinct objects or parts of objects in a depth image D. The resolution of depth images deteriorates rapidly with distance. Therefore, it is not necessary to segment these regions perfectly at larger distances. We subsample the depth image to significantly reduce the amount of computation needed for the segmentation and region filtering steps.

We consider the depth image D as a grid of cells. Each cell contains $\alpha \times \alpha$ depth pixels from D. For each cell $E_{i,j}$ at row i and column j (where $0 \leq i < \frac{h}{\alpha}$ and $0 \leq j < \frac{w}{\alpha}$), s pixels are sampled at random from within the cell and the pixel with the median depth value is selected. If its depth value is valid, its corresponding point in 3D space $\langle p_x, p_y, p_z \rangle$ is calculated using Eq. 2-4 and stored as $p_{i,j}$. Each point $p_{i,j}$ corresponds to a distinct cell $E_{i,j}$, and so there are at most $\frac{wh}{\alpha^2}$ points in the final subsampled image $P = \{p_{i,j} : 0 \leq i < h, 0 \leq j < w\}$.

In the next step, we use a variation of an image segmentation algorithm [9] initially designed for color images. The original algorithm performs multiple segmentations on separate graphs whose edges encode the similarity between neighboring pixels in each color channel, and combines the results. Each segmentation uses a single parameter k, which sets the preferred size of connected components. We adapt the algorithm to work on depth images, where we perform two separate segmentations on graphs whose edges encode similarity of depth and normals between neighboring points, using parameters k_{depth} and k_{normal} respectively.

First, we construct the graph G_{depth} where the vertices are the points in P, and each vertex is connected to its neighbors (if they exist) in the 4 cardinal directions. The weight of an edge between points of metric depth z_1 and z_2 is $|z_1 - z_2|$. We then perform graph segmentation on G_{depth}, grouping the vertices into connected components based on the similarities of the depth values.

For each point $p_{i,j}$, we compute the normal $n_{i,j}$ by performing least-squares regression on a list of samples consisting of the point itself and those of its 8 neighbors which lie on the same connected component in the segmented of G_{depth}. Including only neighbors on the same connected component removes outliers which are far away in 3D space.

In the next step, we construct the graph G_{normal} similarly to G_{depth}, except, here the weights of the edges are the angular differences between the normals at the vertices they connect. The weight of an edge between vertices with normals u and v is thus computed by $\cos^{-1}(u \cdot v)$. We then perform graph segmentation on G_{normal}.

The final output of the segmentation step is a set of regions R. Each region $r_i \in R$ is a distinct set of points from P that are in the same connected component in both G_{depth} and G_{normal}. R is sorted in decreasing order of size, so $|r_i| \geq |r_{i+1}|$ is strictly true, and only regions with at least β points are considered. This filtering helps removing a number of regions that result from noisy data and are unlikely to correspond to a human part.

B. Region Filtering and Merging

This step takes a set of regions R as input, discards those a heuristic deems unlikely to be humans, and merges the remaining regions, returning C, a set of candidates likely to correspond to distinct objects.

For each region $r_i \in R$ its mean depth, height and width in real space, center point $\mu(r_i)$, and the proportion of points that fit a plane are calculated. The mean depth of a region is the mean depth of all the points in the region.

The top, bottom, left and right points of the region are calculated using Eq. 2-4 on its mean depth and the midpoints of each side of the bounding box that contains the region. The metric height of a region is the difference in y between its top and bottom midpoints, while the metric width of a region is the difference in x between its left and right midpoints. The center point $\langle \mu_x, \mu_y, \mu_z \rangle$ of a region is defined by μ_x (the midpoint between its left and right points), μ_y (the midpoint between its top and bottom points) and μ_z (the mean depth of the region). We denote the position of the region on the XZ plane with $\mu_{xz} = \langle \mu_x, \mu_z \rangle$.

We use RANSAC [11] to fit planes to the region in a manner similar to Fast Sampling Plane Filtering [12]. At each of n iterations, 3 points in the region are randomly

picked and used to model a hypothesis plane π_k (where k is the iteration number). The proportion of points that fit a plane is the maximum over all n iterations:

$$\max_{k=1}^{n} \left(\frac{|\{p \in r_i : \text{distance of } p \text{ to } \pi_k < \epsilon\}|}{|r_i|} \right)$$

A list of parametric heuristic rules specifying valid ranges for mean depth, height and width, and the minimum inlier fraction (proportion of points that can fit a plane) is found by measuring the ranges of positive examples in the training set. These rules are then used to eliminate candidate regions that could not contain humans.

The regions are processed in order from the largest one to the smallest. If a region r_i satisfies all of the heuristic rules except that it is too small (its height or width are below the minimums) and appears to be planar (proportion of points fitting plane is large), then we check whether a larger region r_j exists such that

$$\|\mu_{xz}(r_i) - \mu_{xz}(r_j)\| < \delta_{xz} \quad \text{and} \quad |\mu_y(r_i) - \mu_y(r_j)| < \delta_y$$

If such r_j is found, then r_i is merged into r_j, and the new position of the merged region is the mean position of all regions incorporated into it, weighted by their sizes. This step is important, especially for humans further from the camera and subject to distortion due to resolution loss: limbs and other parts are sometimes detached from the torso by the segmentation step, but their close proximity in 3D space allows them to be merged again with the torso.

Our final classification step requires candidates of a fixed size, so the pixels corresponding to each region must be scaled and copied into a $w_c \times h_c$ array called a candidate image. They are positioned such that the image of the region is centered in the candidate image, and the entire bounding box of the merged region fits inside the candidate image. The set of pixels to be copied from a region r_i (after scaling to fit inside the candidate image) is

$$\{d \in E_{j,k} : p_{j+a,k+b} \in r_i \text{ where } a, b \in \{-1, 0, 1\}\}$$

The candidate image is thus composed of $w_c \times h_c$ values that are either raw depth pixels (if a pixel was copied to that position from a depth pixel in the above set) or undefined otherwise. Each candidate $c \in C$ consists of the candidate image along with the bounding box of the merged region the candidate was derived from. This step returns the set C

C. Candidate Classification

The last step of our approach takes the set C and returns H, which is the subset of C classified as human. We use the HOD descriptor [1]. First, we compute the metric depth of each pixel in the candidate window using Eq. 1. Then we compute the HOD descriptor, where the gradient between any undefined value and some other value is always zero. We use signed gradients since the sign of gradients between metric depths is significant (objects, if not occluded, are always closer to the camera than their background).

In the final classification step, we use a Support Vector Machine (SVM) with linear kernel to classify the computed descriptor for a region and decide whether the region contains a human or not.

D. Training the Algorithm

To train the classifier, the algorithm is run on depth image frames from a training set, the candidates saved, and subsequently labeled as positive or negative. The heuristic rules used in the region filtering step can also be derived by choosing ranges that include all positive examples while including as few negative examples as possible.

IV. Experimental Results

We evaluate our human detector with two sets of experiments. In the first set of experiments, we evaluate the accuracy of our approach, while in the second set we investigate the impact of varying the number of training examples on the accuracy of the human detector. We used two datasets with different characteristics for our experiments. The first dataset consists of depth images from a publicly available RGB-D dataset provided by Spinello and Luber [1], [13]. The dataset contains images of people passing by three vertically mounted Kinect sensors and is collected in a university hallway. We call this dataset "the Hallway dataset". The environment is not cluttered, and there are humans at different distances with varying occlusion levels, but all humans in this dataset are in the upright position. We collected a second dataset which we call "the Café dataset" from a crowded café area in our university by driving our service robot around. There is substantial environmental clutter including plants, counters, pillars, and various types of tables and chairs. Standing, sitting, and walking humans are mixed and are often occluded by other humans, by furniture or plants, or by backpacks and other items they are carrying.

A. Accuracy

We evaluate the performance of our algorithm on both the Hallway and Café datasets (Fig. 3). On the Hallway dataset, we compare our algorithm against the HOD11 approach [1] using the annotated depth data from the same Hallway dataset. Our HOD calculation is similar to the HOD descriptor calculation in [1] (gradients are computed between metric depths). The candidates extracted by our algorithm also only contain pixels from the relevant regions, reducing background clutter in the HOD window. Since the original implementation of the HOD11 was not available to us, we were not able to compare the performance of our approach against HOD11 on our Café dataset.

1) Hallway Dataset: For training, our algorithm was run on the training set (700 frames from each of the 3 Kinects in the dataset), and extracted candidates were checked against the annotated bounding boxes. Positive training examples were taken from candidates intersecting with bounding boxes denoting unobstructed people, while negative examples were taken from candidates which did not intersect with the annotated bounding boxes. We performed our test under similar conditions to the experimental evaluation in [1]: we used 1000 positive examples where each example was

TABLE I: Parameters used for evaluation on the Hallway and Café datasets, determined by parameter sweeps.

	Parameter	Hallway	Café
α	Height/width of cell	8	8
s	Samples per cell	13	16
k_{depth}	Depth segmentation parameter	0.4	0.22
k_{normal}	Normal segmentation parameter	0.04	0.024
β	Min region size	5	13
ϵ	RANSAC plane distance threshold	0.025	0.03
	RANSAC min inlier fraction	0.975	0.97
n	RANSAC iterations	20	20
	Region height (meters)	[1.0, 2.4]	[1.0, 3.0]
	Region width (meters)	[0.3, 1.2]	[0.3, 1.5]
δ_{xz}	Max merge distance (meters)	0.5	0.5
δ_y	Max vertical merge distance (meters)	1.5	1.5
	Candidate size (pixels)	64x128	192x192
	HOD cell size (pixels)	16x16	16x16
	HOD number of bins	18	18
	HOD normalization block size	2x2	2x2

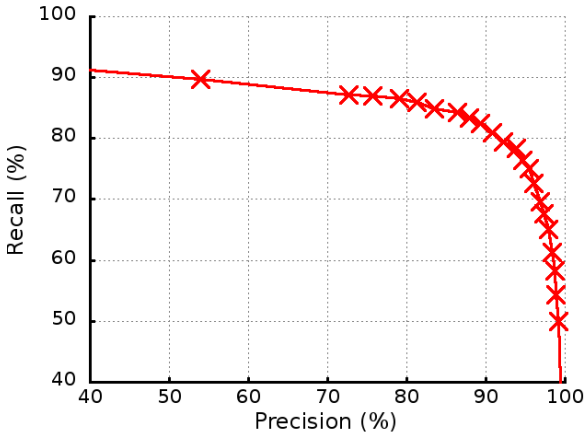
mirrored horizontally for a total of 2000 positive examples, and we randomly selected 5000 negative examples. We used 300 frames from another sequence on each of the 3 Kinects for the testing. Similarly to [1], humans identified by the algorithm were counted as detections if the bounding boxes of the regions overlapped by 40% or more with the annotated boxes, and a no-reward-no-penalty policy was applied to detections of partially obstructed people.

Our algorithm achieved an Equal Error Rate (EER) of 84% using the parameters in Table I, which is comparable to the 83% of HOD11. For applications where recall is less important than precision, our algorithm achieves 96% precision at 70% recall(Fig. 3a), compared to the 85% precision of HOD11. Each point on the precision-recall graph was obtained by taking the mean precision and recall over 20 runs. Our algorithm processed each frame in an average of 25ms on a single core of an Intel Core i5 processor.

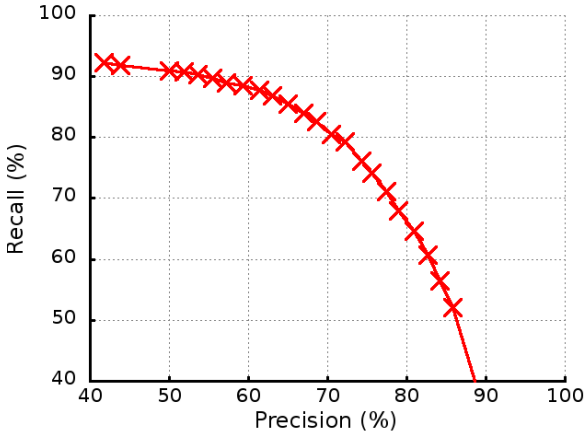
2) Café Dataset: For the purposes of this test, the algorithm was only required to detect people at up to 5 meters away. Detections were counted where bounding boxes overlapped by at least 40% vertically and the detected bounding boxes were horizontally at least 70% inside and occupying at least 30% the width of the annotated box (the modified width overlap requirement allows the algorithm to detect multiple overlapping humans at close proximity to one another as a single human). However, it was required to identify people in any pose (standing, walking, or sitting) even if partially obstructed by up to 50%.

We performed the evaluation using three separate depth image sequences containing about 1000 frames each. We used two sequences for training. We selected a total of 1300 positive and 1350 negative training examples, which were candidates extracted from the training sequences.

We then tested the trained system on 500 randomly selected frames from the third sequence. The no-reward-no-penalty policy was applied to indistinct humans, humans obstructed by more than 50% or less than 50% visible, humans closer than 0.5 meters, and humans further than 5 meters from the robot. Our algorithm achieved an EER of 75% on the Café dataset (Fig. 3b) using the parameters in Table I. Each point on the precision-recall graph was obtained by taking the mean precision and recall over 20 runs. On average, our algorithm processed each frame in about 33ms on a single core of an Intel Core i5.



(a)



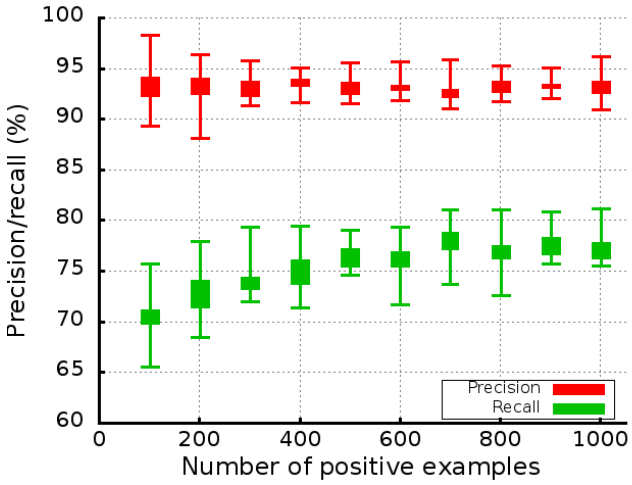
(b)

Fig. 3: Performance evaluation of our algorithm on a) the Hallway dataset, and b) the Café dataset.

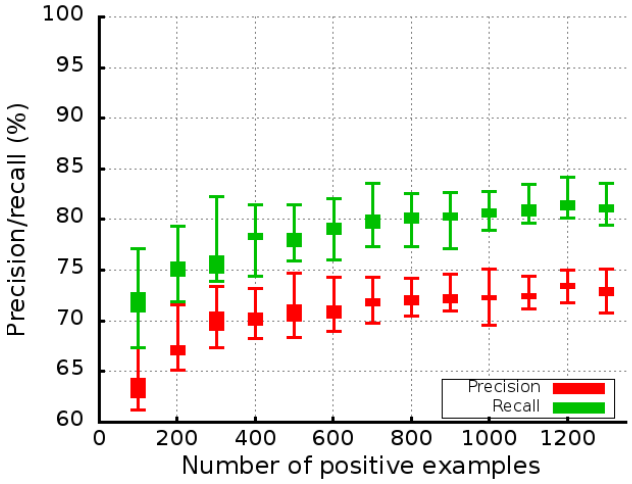
B. Impact of the Number of Training Examples on the Performance

We investigate how the number of training examples affects the performance of the algorithm by conducting several tests using a different number of training examples. We performed the tests on both the Hallway and Café datasets. For each dataset, we ran experiments with 100 to 1000 (for the Hallway dataset) or 100 to 1300 (for the Café dataset) randomly selected positive training examples and a fixed SVM margin threshold θ (we used $\theta = 0$). Each experiment was performed 20 times, each time with the training examples being randomly reselected.

For the Hallway dataset, in each experiment with n positive training examples, each was mirrored horizontally to give a total of $2n$ positive examples, and $5n$ negative examples were randomly selected from the set of negative examples. This follows the ratios used for the evaluation against HOD11. For the Café dataset, in each experiment



(a)



(b)

Fig. 4: Variation of our algorithm's performance with the number of training examples on a) the Hallway dataset, and b) the Café dataset. The whiskers denote the minimum and maximum, and the error bars denotes the middle quartiles.

with n positive training examples, n negative training examples were randomly selected and used.

For the Hallway dataset, the results show that precision levels remain similar even with fewer training examples, and recall levels only began to decrease at below 500 training examples (Fig. 4), therefore, our algorithm achieved precision and recall levels comparable to the HOD11 algorithm even with half as many training examples.

Our algorithm tests considerably fewer candidates due to segmentation and filtering as compared to a brute-force search. This accounts for the improved precision compared to HOD11 and similar overall accuracy even with smaller training sets. Also, the number of false positives is reduced since the windows which may appear human-shaped under the HOD descriptor but are actually implausible (not occupying a contiguous region in 3D space) are filtered out.

For the Café dataset, both precision and recall decreased with fewer training examples. This can be accounted for by the high levels of clutter in the dataset, which resulted in a large number of candidates being tested. This made it more important for the classifier to be better trained so that it would filter out these non-human candidates.

V. Conclusion and Future Work

We presented an algorithm to detect people in indoor environments using depth images provided by an RGB-D camera. Our approach uses a graph-based segmentation algorithm on a depth image to determine regions of interest and tests considerably fewer candidates using the HOD descriptor than previous approaches. We demonstrated better precision, comparable Equal Error Rate, and higher speed than an informed scale-space search on a publicly available dataset of depth images. Our algorithm runs at 30Hz on a mobile robot using a single CPU core, allowing computational headroom for the possibility of algorithmic extensions to further improve the detection performance.

Possible future work includes the use of SVMs with non-linear kernels to classify the descriptors, concatenating the feature vectors resulting from running HOD with different block sizes, utilizing RGB data along with depth to improve candidate classification accuracy (especially of people at farther distances), and employing a parts-based approach to better handle partially occluded people in a variety of postures in cluttered environments.

References

[1] L. Spinello and K. Arras, "People Detection in RGB-D Data," in *Proceedings of IROS 2011*, pp. 3838–3843.

[2] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Proceedings of CVPR 2005*, pp. 886–893.

[3] K. Mikolajczyk, C. Schmid, and A. Zisserman, "Human Detection Based on a Probabilistic Assembly of Robust Part Detectors," in *Proceedings of ECCV 2004*, pp. 69–82.

[4] Q. Zhu, M. Yeh, K. Cheng, and S. Avidan, "Fast Human Detection using a Cascade of Histograms of Oriented Gradients," in *Proceedings of CVPR 2006*, pp. 1491–1498.

[5] L. Spinello and R. Siegwart, "Human Detection using Multimodal and Multidimensional Features," in *Proceedings of ICRA 2008*, pp. 3264–3269.

[6] F. Xu and K. Fujimura, "Human Detection using Depth and Gray images," in *Proceedings of AVSS 2003*, pp. 115–121.

[7] J. Zhou and J. Hoang, "Real Time Robust Human Detection and Tracking System," in *CVPR 2005 Workshops*, pp. 149–149.

[8] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 888–905, 2000.

[9] P. Felzenszwalb and D. Huttenlocher, "Efficient Graph-based Image Segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[10] K. Pulli and M. Pietikäinen, "Range Image Segmentation based on Decomposition of Surface Normals," in *Proceedings of the Scandinavian conference on image analysis*, vol. 2, 1993, pp. 893–893.

[11] M. Fischler and R. Bolles, "Random Sample Consensus: a Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[12] J. Biswas and M. Veloso, "Depth Camera Based Indoor Mobile Robot Localization and Navigation," in *Proceedings of ICRA 2012*, pp. 1697–1702.

[13] M. Luber, L. Spinello, and K. Arras, "People Tracking in RGB-D Data with On-line Boosted Target Models," in *Proceedings of IROS 2011*, pp. 3844–3849.