A COMPARISON BETWEEN BACKGROUND SUBTRACTION ALGORITHMS USING A CONSUMER DEPTH CAMERA

Klaus Greff[1,3], André Brandão[1,2,3], Stephan Krauß[1], Didier Stricker[1,3] and Esteban Clua[2]

[1]*German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany*

[2]*Federal Fluminense University (UFF), Niterói, Brazil*

[3]*Technical University of Kaiserslautern, Kaiserslautern, Germany*

Keywords: Foreground Segmentation, Background Subtraction, Depth Camera, Kinect.

Abstract: Background subtraction is an important preprocessing step in many modern Computer Vision systems. Much work has been done especially in the field of color image based foreground segmentation. But the task is not an easy one so, state of the art background subtraction algorithms are complex both in programming logic and in run time. Depth cameras might offer a compelling alternative to those approaches, because depth information seems to be better suited for the task. But this topic has not been studied much yet, even though the release of Microsoft's Kinect has brought depth cameras to the public attention. In this paper we strive to fill this gap, by examining some well known background subtraction algorithms for the use with depth images. We propose some necessary adaptions and evaluate them on three different video sequences using ground truth data. The best choice turns out to be a very simple and fast method that we call minimum background.

1 INTRODUCTION

The release of Microsoft's Kinect had a huge impact on computer vision. This device changed the face of many problems such as gesture recognition (Tang, 2011), activity monitoring, 3D reconstruction (Cui and Stricker, 2011) and SLAM (Henry et al., 2010; Sturm et al., 2011).

The fusion of different sensors, combined with a very low price, makes the Kinect an excellent choice for many applications. Its certainly most interesting sensor is the depth camera, that Microsoft used to ship a product quality gesture control for the Xbox 360. Since then, the Kinect has been used for a wide variety of problems including skeleton tracking (Kar, 2010), gesture recognition (Tang, 2011), activity monitoring, collision detection (Pan et al., 2011), 3D reconstruction (Cui and Stricker, 2011) and robotics.

Many applications, especially those from the field of human computer interaction, utilize a static camera to track moving persons or objects. Those applications greatly benefit from background subtraction algorithms, which separate the foreground (objects of interest) from the potentially disturbing background. This preprocessing step is well known in computer vision (for an overview see (Cannons, 2008)) and helps to reduce the complexity of further analysis and can even increase the quality of the overall result.

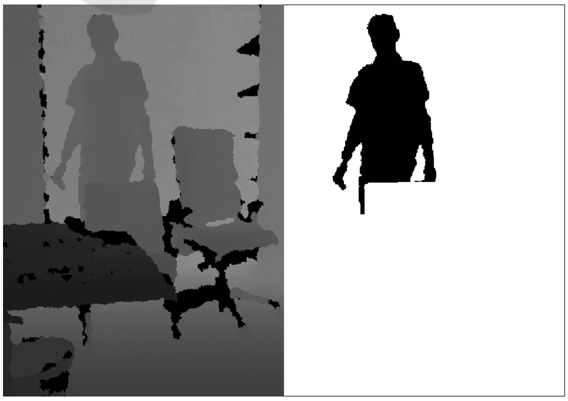

Figure 1: Foreground objects (right) are detected in depth images (left) taken by a static depth camera.

Also, the task of background subtraction appears to be easier with a depth image at hand. It is therefore quite surprising to see that only little work on the subject can be found. Early publications deal with background subtraction based on the use of stereoscopic cameras (Gordon et al., 1999; Ivanov et al., 2000). There are papers dealing with the Kinect, which mention the use of background subtraction (Kar, 2010;

Greff K., Brandão A., Krauß S., Stricker D. and Clua E..
A COMPARISON BETWEEN BACKGROUND SUBTRACTION ALGORITHMS USING A CONSUMER DEPTH CAMERA.
DOI: 10.5220/0003849104310436
In *Proceedings of the International Conference on Computer Vision Theory and Applications* (VISAPP-2012), pages 431-436
ISBN: 978-989-8565-03-7

Stone and Skubic, 2011; Tang, 2011; Xia et al., 2011), however, there is no publication dealing with the topic directly.

This paper tries to fill this gap and provide a starting point for further research. We start by analyzing the characteristics of the Kinect depth camera (Section 2), and their impact on the problem of background subtraction problem (Section 3). After that, we we choose four background subtraction algorithms (Section 4), and adapt them to the domain of depth images (Section 5). Finally we evaluate them using three different depth videos along with their ground truth segmentation (Sections 6 and 7).

2 KINECT DEPTH IMAGE CHARACTERISTICS

We start this section by delivering an overview of the distinct characteristics of depth images provided by Kinect. They will provide the basis to analyze the problems associated with the task of foreground detection. The functional principles of the Kinect will not be discussed in this paper (Refer to (Khoshelham, 2011) instead).

Although depth image resolution is 640×480 pixels but the effective resolution is much lower since the depth calculation depends on small pixel clusters. The detection range is between 50 cm and about 5 m with a field of view of approximately 58 °. Depth information is encoded using 11 bit for the depth information and 1 bit indicating an undefined value.

But the most important property is obviously the usage of distance information instead of color intensities. This which makes the image independent of illumination, texture and color. Direct sunlight, however, can outshine the projected pattern, turning many pixels to undefined. Certain kinds of material properties can also hinder a stable depth recognition, including high reflectiveness and transparency or dark colors.

The depth image contains different types of disturbances and noise. We characterize the pixels according to those errors as follows:

- **Stable:** A fixed depth value with only a small variance increasing quadratically with range (see (Khoshelham, 2011)).
- **Undefined:** A special value meaning that no depth information is available. This is typical for object shadows, direct sunlight, and objects below the minimum range of 50cm.
- **Uncertain:** Switching in a random manner between the undefined and stable state. This is often the case for boundaries of undefined regions, reflections, transparencies, very dark objects, and fine-structured objects (e.g. hair).
- **Alternating:** Switching between two different stable values.

Occasionally, there are pixels with “uncertain” and “alternating” characteristics, i.e. they switch between two different stable values and the undefined state. It is also important to note that alternation and uncertainty do not usually occur pixel-wise but cluster-wise, therefore contours may differ substantially from frame to frame.

3 FOREGROUND DETECTION CHALLENGES

In the following we give a summary of challenges faced by background subtraction algorithms that work on depth images. The list is based upon the more detailed summary of (Toyama et al., 1999). We recite only the challenges related to depth images, and also modified the descriptions to better reflect the characteristics of depth images as provided by the Kinect sensor.

Moved Objects: The method should be able to adapt to changes in the background such as a moved chair or a closed door.

Time of Day: Direct sunlight can outshine the infrared patterns used for depth estimation, resulting in undefined pixels in the according regions. If the illumination changes, the state of the pixels in the affected regions might also change (to stable or undefined), which results in the pixel class “uncertain” (see Section 2). *This is similar to the moved object problem.*

Dynamic Background This problem, originally referred to as *waving trees* in (Toyama et al., 1999), can be caused by any constantly moving background object e.g. slowly pivoting fans.

Bootstrapping: In some environments it is necessary to learn a background model in the presence of foreground objects.

Foreground Aperture: When a homogeneous background object moves, changes in the inner part might not be detected by a frame to frame difference algorithm. *This is especially true for depth images, because there is no color and texture.*

Shadows and Uncertainty: The system has to cope with undefined and uncertain pixels (see Section 2) both in the fore- and background. Additionally, foreground objects often cast shadows, which should not be considered to be foreground. *This problem behaves differently with the Kinect because only the inherent shadow casting of the sensor is relevant. Also these shadows always result in an undefined value making it easy to rule them out as foreground.*

We omitted the point “Light Switch” as artificial lighting does not affect the Kinect. Furthermore, the challenges “Sleeping Person” and “Waking Person” were dropped, because we believe this task is better solved at a higher level that includes semantic knowledge[1]. Finally, the “Camouflage” problem was also omitted, since depth images lack both, color and texture.

4 BASIC METHODS

Many background subtraction and foreground detection algorithms have been proposed. Cannons (Cannons, 2008) provides an overview of the subject. Most of those algorithms were created having color images in mind. In our work we chose four standard methods and adapted to the segmentation of depth images, achieving three suitable and high quality possible solutions.

First Frame Subtraction: In this method the first frame of the sequence is subtracted from every other frame. Absolute values that exceed a threshold are marked as foreground.

Single Gaussian: In this method, the scene is modeled as a texture and each pixel of this model is associated to a Gaussian distribution. During a training phase pixel-wise mean and variance values are calculated. Later on pixel values that differ more than a constant times the standard deviation from its mean are considered foreground. This method was used in Pfinder (Wren et al., 1997).

Codebook Model: This more elaborate model (Kim et al., 2005) aggregates the sample values for each pixel into codewords. The Codebook model considers background values over a long time. This allows to account for dynamic backgrounds and is also used to bootstrap the system in the presence of foreground objects.

Minimum Background: This is one of the first models completely developed depth images (Stone and Skubic, 2011). During training stage, the minimum depth value for each pixel is stored. Afterwards every pixel closer to the camera (depth value smaller than stored value) is considered foreground. This works well for range based data because the foreground usually is *in front of* the background.

5 ADAPTATIONS

The presented methods need to be adapted in order to work for depth images. So we developed and included different improvements: Uncertainty Treatment, Filling the Gaps and Post-Processing.

Uncertainty Treatment: Treating the undefined value (zero) as a normal depth information leads to problems with almost every model (e.g. turning most shadows into foreground). So the question arises how to treat undefined values. We certainly do not want a shadow of an object to be considered foreground. But sometimes the shadow of some object falls onto the foreground, for example a hand in front of the torso. Or the foreground contains undefined regions, as caused by glass for example. These problems illustrates that on a pixel level the question, whether some undefined value belongs to the foreground or not, is impossible to decide. This decision clearly requires additional knowledge (other sensory input, the region around the pixel). But it is not the task of a foreground detection algorithm to do complex reasoning. It should merely be a preprocessing step (see Principle 1 in (Toyama et al., 1999)). Thus, we decided to treat all undefined values as background for all the presented methods.

Filling the Gaps: Undefined pixel values can lead to gaps within the background model learned by each presented algorithm. Those gaps can lead to errors because every "defined" pixel value differs from an undefined background. So depending on the chosen policy they will either lead to false positives or to false negatives. In order to close these gaps, an image reconstruction algorithm (like (Telea, 2004)) that tries to estimate the correct values for the undefined regions can be used. This can obviously only reduce the errors induced by those gaps, and not completely eliminate it. According to our experiments, the method from (Telea, 2004) works quite well in practice.

Post-processing: As discussed earlier, the depth images as generated by Kinect contain lots of noise. This leads to a large amount of false positives in form of very small blobs and thin edges around objects. The desired foreground (i.e. humans) on the other hand appears always quite large because of the range

[1] For more in-depth discussion please refer to (Toyama et al., 1999) Section 4.

constraints of the Kinect sensor. Therefore, morphological filters are an easy way of improving the final result. We experimented with the erode-dilate-operation and the median filter, but both of them change the contour of the desired foreground. A connected component analysis, on the other hand, combined with an area threshold is suitable to remove the false positive regions while keeping the foreground intact. This threshold can be quite high for most applications (1000 pixels in our case). This filtering is applied as a post-processing step to all of the presented methods. However, we also evaluate each of them without any filtering.

6 EXPERIMENTS

In order to evaluate the different approaches we recorded a set of three typical sequences for the application of human body tracking. All of them are recorded indoors at 30 fps and with VGA resolution. Every sequence contains at least 100 training frames of pure background.

Gesturing 1: The camera shows a wall in a distance of approximately 3 meters for a few seconds. Then a person enters and stands in front of the sensor performing some gestures (641 frames).

Gesturing 2: The same as in the first sequence, but the background contains a lot of edges (643 frames).

Occlusion: This sequence shows an office with some chairs, then a person enters and walks in between those chairs. The ideal foreground for this sequence is marked manually in every frame (567 frames).

The Kinect depth sensor produces data with high noise at the edges of objects. For this kind of noise a single frame evaluation would not be representative. Therefore, we created ground truth videos containing the ideal foreground segmentation for each sequence. The first two sequences were recorded in a way that simple distance truncation cleanly separates the foreground. For the third sequence the foreground was marked manually in each frame.

7 RESULTS AND DISCUSSION

We measured the error of every algorithm using the absolute amount of *false positives* N_{e_+} (background that was marked as foreground) and *false negatives* N_{e_-} (foreground that was marked as background). To establish some comparability we also measure an error ratio for every sequence, that is

$$e_+ = \frac{N_{e_+}}{N_{BG}} \text{ and } e_- = \frac{N_{e_-}}{N_{FG}} \tag{1}$$

respectively, where N_{BG} and N_{FG} are the total number of background and foreground pixels in the ground truth sequence. The results can be found in Table 1 and some selected frames for every video and method are shown in Figure 2.

The *First Frame Subtraction*, performs surprisingly well. Unfiltered, it produces the least false negative ratio among all considered algorithms. But it is sensitive to all sorts of noise, so depending on the background this can lead to many false positives.

The statistical approach used by the *Single Gaussian* method is affected by the high variances of alternating pixels on the one hand and the low variance of stable pixels on the other hand. If the constant multiplied with the standard deviation is high, this will lead to false negatives when a foreground object occludes the high variance region. If the constant is small, stable pixels will emit a lot of noise. Consequently, we concluded that the depth values provided by Kinect cannot be modeled effectively by a single Gaussian distribution.

The best overall results are achieved by the *Codebook Model* and the simple *Minimum Background* method. Both methods manage to eliminate the errors of uncertain and alternating regions without missing the desired foreground. Since the Minimum Background method is faster and simpler, we found it to be the best choice among the algorithms we have considered. This result might not come as a surprise, since the Minimum Background method is the only one that takes advantage of the depth information.

8 CONCLUSIONS

In this paper we have adapted four different approaches of background subtraction to depth images. They were evaluated on three different test sequences using ground truth data. We have identified a simple and fast algorithm, the *Minimum Background* algorithm, that gives close to perfect results. So for the scenario of a static Kinect and a static background the problem of background subtraction can be considered solved. This clearly shows the task of background subtraction to be much easier for depth images than for color images.

Nevertheless, there are still some open questions for future work. Scenarios with a moving Kinect, or

Table 1: The results for the algorithms run on the test sequences. The rows with e_+ and e_- represent false positives and false negatives respectively. The values are specified with respect to the total number of background pixels in the ground truth data. The lowest positive and negative errors are highlighted for each test sequence.

		Gesturing 1		**Gesturing 2**		**Occlusion**	
		$N_{\text{BG}} = 179,896,685$		$N_{\text{BG}} = 160,009,777$		$N_{\text{BG}} = 164,507,583$	
		$N_{\text{FG}} = 17,018,515$		$N_{\text{FG}} = 37,519,823$		$N_{\text{FG}} = 9,674,817$	
		plain, in %	*filtered, in %*	*plain, in %*	*filtered, in %*	*plain, in %*	*filtered, in %*
First Frame	e_+	8.83	2.67	0.71	0.02	1.75	0.09
Subtraction	e_-	**0.00**	0.05	**0.00**	0.37	**0.91**	1.58
Single	e_+	0.52	0.19	1.91	0.07	2.40	0.85
Gaussian	e_-	8.33	9.15	9.98	13.55	6.42	9.02
Codebook	e_+	0.06	0.01	0.04	0.01	0.24	**0.07**
Model	e_-	**0.00**	0.03	**0.00**	0.30	1.32	1.84
Minimum	e_+	0.06	**0.00**	0.04	**0.00**	0.19	0.07
Background	e_-	**0.00**	0.02	0.00	0.37	1.20	1.94

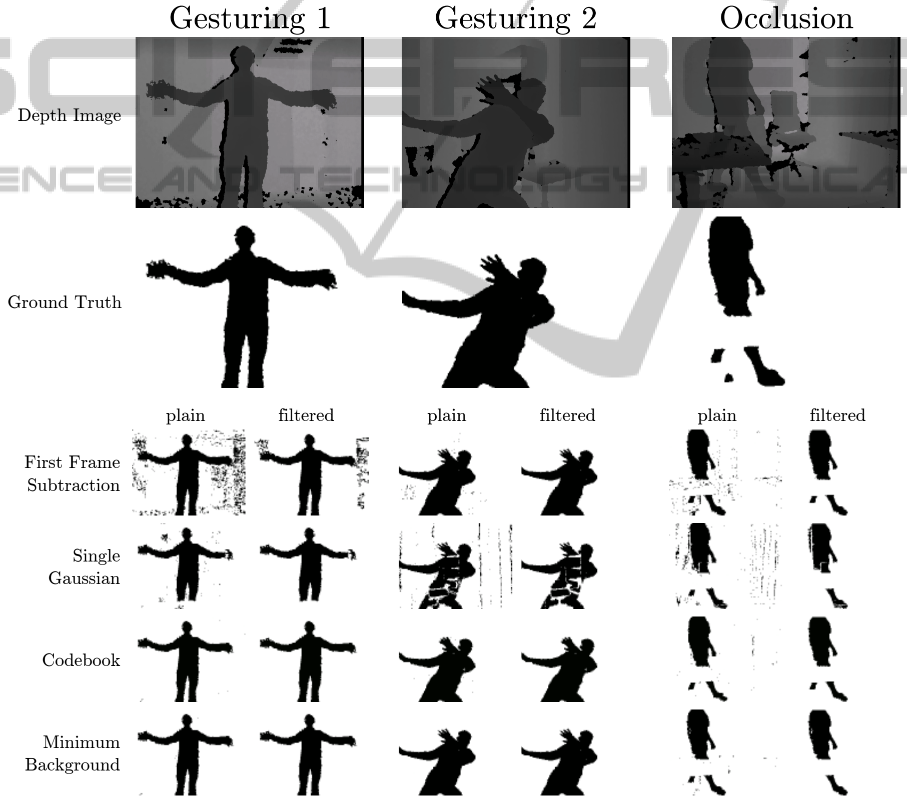

Figure 2: Sample images from the segmentation for all methods and all sequences. Every image is presented with and without filtering (see Post-Processing in Section 5).

a dynamic Background require more sophisticated algorithms. For the problem of bootstrapping we suggest testing more complex background subtraction techniques, e.g. optical-flow, that try to handle foreground clutter in the training phase. Finally, the color camera of the Kinect could complement the depth camera for the task of background subtraction.

ACKNOWLEDGEMENTS

We would like to thank the German Research Center for Artificial Intelligence (DFKI), Federal Fluminense University - Brazil (UFF), CAPES-Brazil, DFG IRTG 1131 and German Academic Exchange Service (DAAD).

REFERENCES

Cannons, K. (2008). A review of visual tracking. Technical Report CSE-2008-07, York University, Department of Computer Science and Engineering.

Cui, Y. and Stricker, D. (2011). 3D shape scanning with a Kinect. In *ACM Transactions on Graphics*.

Gordon, G., Darrell, T., Harville, M., and Woodfill, J. (1999). Background estimation and removal based on range and color. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2.

Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2010). RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *Proc. of the International Symposium on Experimental Robotics (ISER)*, Delhi, India.

Ivanov, Y., Bobick, A., and Liu, J. (2000). Fast lighting independent background subtraction. *International Journal of Computer Vision*, 37(2):199–207.

Kar, A. (2010). Skeletal tracking using Microsoft Kinect. Department of Computer Science and Engineering, IIT Kanpur.

Khoshelham, K. (2011). Accuracy analysis of kinect depth data. In *ISPRS Workshop Laser Scanning*, volume 38.

Kim, K., Chalidabhongse, T. H., Harwood, D., and Davis, L. (2005). Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11(3):172–185.

Pan, J., Chitta, S., and Manocha, D. (2011). Probabilistic collision detection between noisy point clouds using robust classification. In *International Symposium on Robotics Research (ISRR)*.

Stone, E. and Skubic, M. (2011). Evaluation of an inexpensive depth camera for passive in-home fall risk assessment. In *Pervasive Health Conference*, Dublin, Ireland.

Sturm, J., Magnenat, S., Engelhard, N., Pomerleau, F., Colas, F., Burgard, W., Cremers, D., and Siegwart, R. (2011). Towards a benchmark for RGB-D SLAM evaluation. In *Proc. of the RGB-D Workshop on Adv. Reasoning with Depth Cameras at Robotics*, Los Angeles, USA.

Tang, M. (2011). Recognizing hand gestures with Microsoft's Kinect. Department of Electrical Engineering, Stanford University.

Telea, A. (2004). An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 9(1):25–36.

Toyama, K., Krumm, J., Brumitt, B., and Meyers, B. (1999). Wallflower: Principles and practice of background maintenance. In *IEEE International Conference on Computer Vision*, volume 1, pages 255–261, Los Alamitos, CA, USA. IEEE Computer Society.

Wren, C., Azarbayejani, A., Darrell, T., and Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785.

Xia, L., Chen, C. C., and Aggarwal, J. K. (2011). Human detection using depth information by Kinect. In *International Workshop on Human Activity Understanding from 3D Data in conjunction with CVPR (HAU3D)*, Colorado Springs, CO.