Area Based Alarm System Using 3D Cameras

Daniel Gustafsson, Jonatan Kåhrström

November 13, 2012

Acknowledgments

We would like to thank our advisers at Axis Communications; Willy Sagefalk and Gustav Träff, as well as our adviser Professor Karl Åström from the Faculty of Engineering, Lund University. We would also like to thank the Product Concept Group (PCNI) at Axis for excellent support and for being a friendly and helpful group to work with. A special thanks to the personnel in various jewelry and perfume shops in Lund for allowing us to photograph the premises.

Abstract

Depth map cameras provide new ways of designing surveillance systems. In this thesis we evaluate three different cameras from two different depth sensor techniques, and propose a complete method for detecting thefts over a counter in a retail environment. Our algorithm covers pre-processing with noise reduction and background segmentation using the reflected signals amplitude as a confidence measurement. A plane is fitted both to the 3D points of the top of the retail counter as well as to the 3D points on the side (cashier's side) of the retail counter. The algorithm determines which foreground pixels are on the wrong side of both these planes. By running this result through a few methods to improve rigidity, we show that it is possible to detect thefts with a very high detection rate and low false positive rate. Finally we present the results from our testing of different versions on a database of activities with known ground-truth (theft/no theft).

Contents

Chapter 1

Introduction

1.1 Background

Digital surveillance systems traditionally consists of ordinary 2D video cameras and are thus limited to working with the information that can be obtained from these. There are several alternatives to this technique, of which the field of depth map cameras (commonly referred to as 3D cameras) is one of the most obvious and interesting. These enable for an acquisition of the distance to various objects in a scene, which could provide for more complex image analysis algorithms or merely to simplify existing techniques.

There are three main types of depth map camera techniques; time of flight (TOF) cameras, structured light 3D scanning and stereo vision cameras, of which the former two will be subject to investigation in this thesis.

The main focus of this thesis has been around the concept of Time-of-Flight cameras, and specifically to develop an algorithm which handles their weaknesses well. The algorithm has then been slightly modified in order to work with a structured light camera as well.

1.2 Goals

With regards to the potential advantages of depth sensors, this master thesis was devoted to the following goals:

- To list which types of depth sensors that are available in order to test the sensors.
- To define a use case for retail in more detail.
- To set up test environments which covers different aspects e.g. material, lighting and sensor mounting options.

- To test depth sensors in test environment and to choose two to three for prototypes.
- To develop an algorithm in order to detect shoplifting behavior.
- To build a prototype to test whether it is possible to detect shoplifting behavior.

1.3 Case description

Thefts over the counter in a retail environment is a problem which may occur, e.g. when a salesperson is distracted, inattentive or obscured. This should mainly be a problem for shops selling small, expensive and exclusive items such as jewelry, perfumes and consumer electronics. In order to address such thefts, one can imagine a security system which detects prohibited behaviour and either warns the personnel or gives the perpetrator feedback by for example flashing a red light or emitting a warning sound. Such behaviour can for example be to reach for an item behind the counter. This kind of system should preferably allow for the personnel to reach out from behind the counter and act normally and should thus be able to distinguish them from customers and potential thieves.

1.4 Environment description

The general lighting in the previously mentioned shops is often moderate with spotlights angled towards (see Appendix A), and occasionally inside the counters . For simplicity, the counters are regarded as rectangular with either a white coated or a transparent glass finish. The shape could in a later stage be generalised to an angled or curved disc as required.

1.5 Related works

Real time operating depth map techniques is still a relatively new field. The techniques are mostly limited to indoor environments and rather short distances. With these limitations not much research has been done in applying such cameras for surveillance. However some work has been done. A K Mishra et al. [19] suggests using multiple stereo cameras to create a 3D scene from which a foreground can be extracted and objects identified in a robust way. It points at advantages such as removing hardships on detecting multiple objects and changing occlusion.

High noise ratio is as of today a major setback for many depth sensors, and especially Time-of-Flight cameras. It is therefore crucial to apply efficient noise reduction. In this field a lot more work has been done. In [14]

by J. Mure-Dubois et al. the different noise sources in time of flight cameras are described and suggestions on how to reduce them are made. [8] suggests filters that takes the amplitude of the signal into consideration to smooth the images.

This thesis is as far as we can see the first to challenge this specific problem of automatically detecting theft behaviour in a specific environment.

Chapter 2

Theory

2.1 Cameras

2.1.1 Time-of-Flight cameras

Time of flight cameras are a type of depth sensors, built on the principle of time of flight for electromagnetic waves, i.e. a technique for measuring distances using the time it takes for transmitted light to be reflected back to the camera.

One of the essential parts of the camera is the illumination device which actively illuminates the scene. This is typically done with a sinusoidal intensity modulated signal in the near infrared light spectrum called the reference signal, r. The light is reflected by objects in the scene and this reflected signal, s, is detected by the depth sensor after a time of flight, τ. The sensor consists of a matrix of distance measuring devices called smart pixels, consisting of two light collectors each. The light collectors alternately detects the reflected light in accordance with the frequency of the reference signal (see Figure 2.1), by charging an integrational capacitor each [11].

By computing the phase shift, ϕ_d, between the reference signal and the reflected signal, the camera can evaluate the distance

$$d = \frac{c}{2f_{mod}} \frac{\phi_d}{2\pi} \ , \tag{2.1}$$

where c is the speed of light and f_{mod} the frequency with which the reference signal is modulated. Because of the periodicity of the cosine signal, however, there exists a maximum distance where an object can be uniquely determined. This is called the wrapping distance and is dependent on the modulation frequency f_{mod} as

$$d_w = \frac{c}{2f_{mod}} \ . \tag{2.2}$$

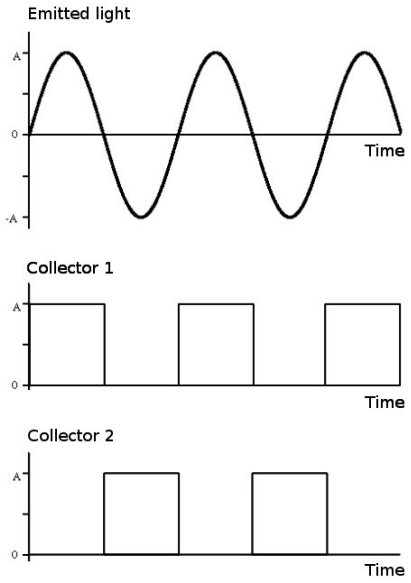

Figure 2.1: A schematic Figure of the pixel light collectors.

Note that this corresponds to the same expression as (2.1), with $\phi_d = 2\pi$. With a modulation frequency of for example $20MHz$, this equates to roughly $7.5m$.

The phase shift can now be determined from the correlation between the reference signal and the reflected optical signal [7]. This is done by sampling several phase images at phase shifts of $\alpha_n = \frac{2\pi n}{N}$

$$I_n = \frac{1}{t_1 - t_0} \int_{t_0}^{t_1} r(t) \cdot s(t + \frac{\alpha_n}{2\pi f_{mod}} + \tau) dt \ , \tag{2.3}$$

where r and s so far are considered arbitrarily shaped signals and $T = t_1 - t_0$ is the integration time.

Since both the reference signal and the reflected signal are modulated with the same frequency, they can be rewritten using Fourier series as

$$r(t) = \sum_{j=-\infty}^{\infty} r_j e^{ij\omega t}, \quad s(t) = \sum_{k=-\infty}^{\infty} s_k e^{ik\omega t} \ , \tag{2.4}$$

which leads to the correlation formula

$$I_n = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} r_j s_k e^{ik(\alpha_n + \omega\tau)} \underbrace{\frac{1}{t_1 - t_0} \int_{t_0}^{t_1} e^{ik\omega t} e^{ij\omega t} dt}_{=\eta} \ . \tag{2.5}$$

For $j = -k$, the integral equates to $t_1 - t_0$, which leads to $\eta = 1$. For $j \neq -k$, η has a maximum magnitude of $\frac{2}{(t_1-t_0)(j+k)\omega}$, which becomes negligible as long as the integral times are much larger than the modulation time $\frac{2\pi}{\omega}$. This leads to the final expression for the correlation

$$I_n \approx \sum_{k=-\infty}^{\infty} r_{-k} s_k e^{ik(\alpha_n + \omega\tau)} \quad . \tag{2.6}$$

In practice, most cameras use four samples at a time and a sinusoidal reference signal

$$s(t) = c' + A' \cos(\omega t) \quad , \tag{2.7}$$

which leads to the raw intensity images

$$\begin{array}{rcl} I_n & = & c + \frac{A}{2}(e^{2\pi i \frac{n}{N}} e^{i\phi_d} + e^{-2\pi \frac{n}{N}} e^{-i\phi_d}) \\ & = & c + A\cos(\alpha_n + \phi_d) \end{array} \quad , \tag{2.8}$$

where $c = r_0 c'$ and $A = |r_1| A'$, [8]. Inserting $n = 0 \ldots 3$ leads to

$$\begin{array}{rcl} I_0 & = & c + A\cos(\phi_d) \quad , \\ I_1 & = & c - A\sin(\phi_d) \quad , \\ I_2 & = & c - A\cos(\phi_d) \quad , \\ I_3 & = & c + A\sin(\phi_d) \quad . \end{array} \tag{2.9}$$

These four expressions finally yield the formulas to compute the phase shifts

$$\phi_d = arg\left(\sum_{n=0}^{3} I_n e^{-(\pi n/2)}\right) = \arctan 2(I_{3-1}, I_{2-0}) + \pi \quad , \tag{2.10}$$

amplitudes

$$A = \frac{1}{2}\left|\sum_{n=0}^{3} I_n e^{-(\pi n/2)}\right| \quad , \tag{2.11}$$

and the incident light intensities

$$c = \frac{1}{4}\sum_{n=0}^{3} I_n. \tag{2.12}$$

2.1.2 Structured-light 3D scanner

The most simple kind of structured light sensor sends out a single laser beam, which is reflected in the scene, back to the camera sensor. If it is reflected from an object far away from the camera, the reflected beam will travel a longer distance compared to if it had been reflected on an object close by. It will thus hit a slightly different spot on the sensor, which is illustrated in Figure 2.2.

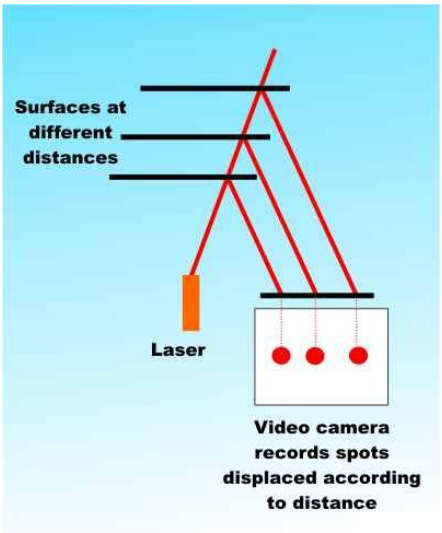

Figure 2.2: Schematic image of the principle of structured light 3D scanning.

The laser and the sensor are calibrated so that it is known what place on the sensor the beam hits when reflected from an object at a known distance. The displacement from this position on the sensor can, together with triangulation, is used to calculate the distance from the camera to the object. As this method only scans one point in the scene, the laser has to sweep in two dimensions. It is more efficient to scan using a line instead of a single dot, in this case the scanner only has to be sweep in one direction in order to capture the entire scene. In real time application both those methods are too time consuming, so a three dimensional scanning method has to be used. There are three ways to do this: multi-stripe patterns, a grid, or using multiple dots. These are called bi-dimensional patterns, and their patterns are created by splitting the laser. One method to do this is with a hologram. In the case of a dot pattern each dot reflects light back to the sensor. To identify the dots on the sensor is a correspondence problem. This can be solved using a pattern which gives a unique signature for each pixel. Using this signature it is possible to identify where on the sensor each pixel was detected. The Microsoft Kinect device which is used in this thesis combines three different patterns each with its maximum accuracy within a specific range, making for precise measurements over a longer range possible.

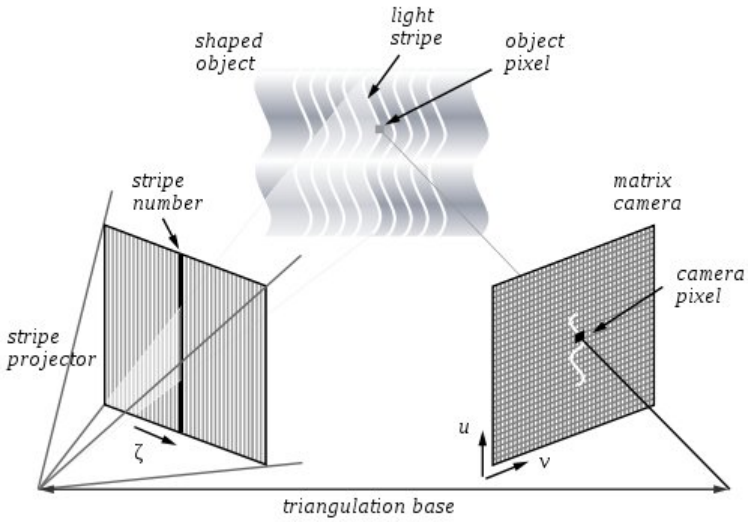

Figure 2.3: Image depicting a stripe pattern and the corresponding detection and triangulation for one of these stripes.

The structured light method has a problem with so called "dead areas". These are patches in the image where no depth data can be obtained due to the light pattern being obstructed by an object in the captured scene. This can be seen in Figure 2.4.

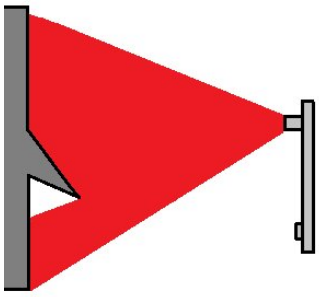

Figure 2.4: Shadowing problem due to an obstructing object in the captured scene.

2.2 TOF noise sources

The images from time of flight cameras are corrupted by substantial noise. This noise arises from different sources which can be divided into two different categories, *systematic* and *non-systematic* errors. The noise from the systematic sources can be eliminated using camera calibration and the non-systematic by filtering.

2.2.1 Systematic errors

According to [13], there are five types of known systematic errors in time of flight cameras. The first is *depth distortion*. This error arises when modeling the emitted infrared signal. The theoretical model used can not in practice be generated perfectly and will thus contaminate the depth values with an offset in each pixel dependent on the measured depth value.

Integration-time related errors is an error that can be observed when changing the integration time of the camera. Dependent on what integration time is used, the depth values of the entire scene is changed. The source of this error is as of march 2011 still an object of investigation, [13]

Another type of systematic error is the *built-in pixel-related error*, an error which gives a unique constant offset in each pixel. This corruption arises mainly from two hardware properties in the camera, differences in the CMOS-gate materials and capacitor charge time delay during the signal correlation process.

Amplitude-related errors are related to the hardship of obtaining a good amplitude value for each pixel in the image. The error appears when a pixel either has a too low amplitude value or the amplitude is over saturated. There are three main reasons why this happens. Pixels in the outskirt of the image or far away receives less illumination due to non-uniform LED illumination. This results in a weaker reflected signal, giving pixels in those areas a lower amplitude. If the object instead is too near the camera, the pixel is instead over-saturated resulting in invalid pixel values. The amplitude value can be tweaked by changing the integration time. This is a trade off between long integration time, giving far away pixels good amplitude, but also making the near pixels over-saturated and a shorter one, making near pixels measurable but pixels far away get a low amplitude. The third source of amplitude-related errors is reflections. When a non-specular area is illuminated it retains energy and thereby changes the phase of the reflected light, resulting in an incorrect value.

The last kind of systematic error are *heat related errors*. This error derives from the fact that the semiconductor material is affected by temperature changes, causing the depth values to drift until the temperature is stabilised.

2.2.2 Non-systematic errors

Aside from the systematic errors there are also non-systematic noise sources. There are four major sources they arise from.

Amplitude related errors can occur for other reasons than the previously mentioned systematic ones. If, for example, a close object causes another object far away to be less illuminated by occluding some of the modulated light, the signal to noise ratio can be very low, which can cause inaccurate readings. There are various ways to reduce this type of error. It can either be removed by filtering the low amplitude pixels or by applying more sophisticated filters taking more parameters into account such as the importance of the accuracy in that pixel.

Multiple light reception is an error that mainly arises from surface edges and concave areas. When surfaces are present in the scene, the low resolution of the camera can cause one single pixel to be illuminated from both the close and far end part of the edge [13]. In scenes where concave surfaces are present, the light reflections may cause a single pixel sensor to capture multiple light reflections due to the way a concave area reflect lights, this is illustrated in Figure 2.5.

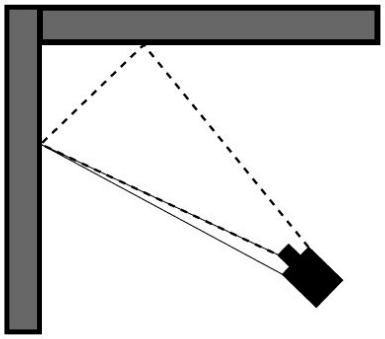

Figure 2.5: Illustration of multiple reflections caused by concave surface. Image taken from [13].

Light scattering is an error which originates in when objects in a scene appear to be close together in the images, but have different depth values. It is the effect of multiple light reflections between the lens and the sensor. When the illuminating light is reflected back from a close object, the signal has a high amplitude relative to the amplitude from objects further away, as the amplitude decrease is dependent on the distance as $A \propto 1/d^2$, [14]. This makes the signals reflected in the lens from the close object strong enough to compete with the directly reflected signals from the objects further away. This is illustrated in Figure 2.6 below.

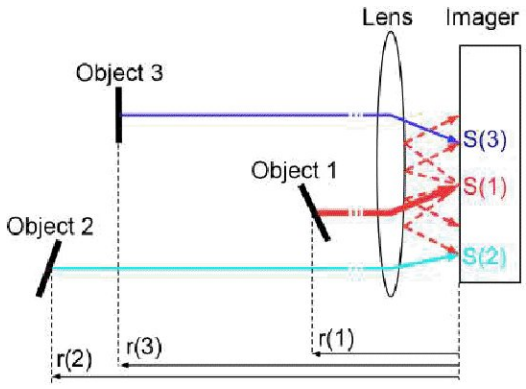

Figure 2.6: Illustration of light scattering. Image taken from [13].

The pixel corruption from light scattering can be reduces by first choosing an optimal integration time that reduces the saturation problem. This integration time can be chosen using a mix of the amplitude and intensity values, where the sum of these are compared to a maximum allowed threshold [12].

The last type of noise that appear with time of flight cameras is the motion blur. This artifact derives from motion during the integration time, causing the edges of a moving object to receive corrupted values. The reason for this is that the camera will take four images, each for a duration of the integration time. This will make moving objects appear in different places in each raw image.

2.3 Create plane

One simple and robust way to create a plane $ax + by + cz + d = 0$ from three interest points p_1, p_2 and p_3 with coordinates $(p_{1,x}, p_{1,y}, p_{1,z})$, $(p_{2,x}, p_{2,y}, p_{2,z})$ and $(p_{3,x}, p_{3,y}, p_{3,z})$ respectively, firstly by computing the two non-parallel vectors

$$\mathbf{v}_1 = (x_1, y_1, z_1) = (p_{1,x} - p_{2,x}, p_{1,y} - p_{2,y}, p_{1,z} - p_{2,z}) \ , \tag{2.13}$$

$$\mathbf{v}_2 = (x_1, y_1, z_1) = (p_{3,x} - p_{2,x}, p_{3,y} - p_{2,y}, p_{3,z} - p_{2,z}) \ . \tag{2.14}$$

As both these vectors lie in the plane, we can find the normal vector $\mathbf{n}$ using the cross product $\mathbf{v}_1 \times \mathbf{v}_2$. As the plane parameters a, b, c are the components of the normal vector we get

$$a = y_1 z_2 - z_1 y_2 \ , \tag{2.15}$$

$$b = z_1 x_2 - x_1 z_2 \ , \tag{2.16}$$

$$c = x_1 y_2 - y_1 x_2 \ . \tag{2.17}$$

Finally we can rewrite the scalar equation of the plane

$$a(x - p_x) + b(y - p_y) + c(z - p_z) = 0 \tag{2.18}$$

where $p(x, y, z)$ is one of the reference points p_1, p_2 or p_3 as

$$ax + by + cz + d = 0 \tag{2.19}$$

by setting the parameter d to

$$d = -(ap_x + bp_y + cp_z) \tag{2.20}$$

The next step is to determine if the keypoints are on the correct side of the plane. This can be done by evaluating the distance from the point to the plane. This distance is easily calculated as the projection of a vector $\mathbf{w}$ from a point (x, y, z) in the plane to the keypoint (x_0, y_0, z_0), onto the normal vector $\mathbf{n}$.

$$\begin{aligned} D &= \frac{|\mathbf{n} \cdot \mathbf{w}|}{|\mathbf{v}|} \\ &= \frac{|a(x-x_0)+b(y-y_0)+c(z-z_0)|}{\sqrt{a^2+b^2+c^2}} \\ &= \frac{|ax+by+cz-ax_0-by_0-cz_0|}{\sqrt{a^2+b^2+c^2}} \\ &= \frac{|-d-ax_0-by_0-cz_0|}{\sqrt{a^2+b^2+c^2}} \\ &= \frac{|d+ax_0+by_0+cz_0|}{\sqrt{a^2+b^2+c^2}} \ . \end{aligned} \tag{2.21}$$

By dropping the absolute value of the nominator, D becomes the signed distance to the plane. The sign of D then represents on what side of the plane the point (x_0, y_0, z_0) is located.

2.4 Edge detection

A crucial part of the plane fitting is to detect suitable 3D-points from which to base the model estimation. An appropriate and convenient way of obtaining these points is to choose edge points on the counter.

2.4.1 Sobel filtering

One of the most common ways to detect edges in images is to use the Sobel operator in one or two directions. This is essentially a discrete differentiation operator, which result is an estimation of the gradient in each pixel of the image. Normally this is done using the gray scale image, but with the data obtained from the TOF camera, the depth map is more suitable.

Formally, the Sobel operator consists of two 3×3 kernels which are convolved with the image in order to obtain the approximate derivates in the

x and y directions. As we are only interested in depth changes across the counter in the y direction (from the outside to the inside), the depth images are only convolved with the kernel

$$\mathbf{S}_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \ , \tag{2.22}$$

leading to the image

$$\mathbf{G}_y = \mathbf{S}_y * \mathbf{D} \ . \tag{2.23}$$

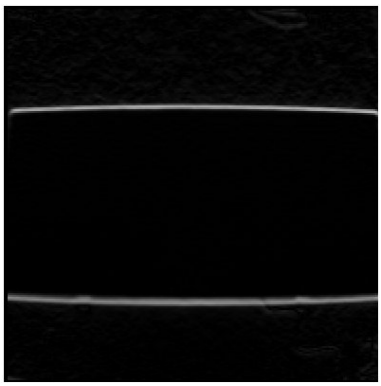

Figure 2.7: Resulting image $\mathbf{G}_y$ after convolving the background with $\mathbf{S}_y$.

2.5 RANSAC

RANSAC (the RANdom SAmple Consensus), is a model parameter estimation algorithm which, unlike many other techniques was developed specifically for use in imaging and graphics implementations. The largest benefit of RANSAC is its ability to estimate a model to a set of observations with a large number of outliers, i.e. erroneous data points.

RANSAC is an iterative, deterministic algorithm and thus only produces a reasonable result with a certain probability based on the number of iterations. The algorithm can be divided into the following base steps:

- Randomly select the minimum number of data points from the observations needed to estimate the model parameters.
- Construct a model from the selected data points.
- Determine how many of the observations fit the model with a predetermined tolerance, ϵ.

- If this number exceeds a threshold, compare the total error of these inliers to the previously best error.
- If this error is smaller than the previously smallest error, save the model parameters and inliers as the best set, along with their corresponding total error.
- Repeat from step one.

This is repeated N times, where N is chosen high enough to achieve at least one model without outliers with a reasonable probability, p. Let w denote the probability of randomly selecting an inlier from all of the observations and thus that $1 - w^m$ is the probability to select at least one outlier from m selected points. The probability to observe an outlier after N iterations is therefore

$$1 - p = (1 - w^m)^N. \tag{2.24}$$

Assuming $w = 0.2$ and $p = 0.99$, this leads to a minimum of 573 iterations.

2.6 Background models

Working only with the foreground provides many advantages over working with the entire image, such as less pixels to process and easier object segmentation.

2.6.1 Threshold model

The basic concept behind a threshold background model is to generate a foreground mask by computing the difference between the current pixels and a reference background model and to make the classifying decision by comparing this difference to a pre-defined threshold value.

The first step is to create a background model. To do this, a background image is generated from the mean of a series of initialization frames. From a depth map of size $m \times n$

$$\mathbf{D} = \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ d_{m,1} & d_{m,2} & \cdots & d_{m,n} \end{pmatrix}, \tag{2.25}$$

a background model $\mathbf{B}$ is generated

$$\mathbf{B} = \begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,n} \\ b_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ b_{m,1} & b_{m,2} & \cdots & b_{m,n} \end{pmatrix}. \tag{2.26}$$

Here each element is calculated as

$$b_{y,x} = \frac{\sum_{t=t_0}^{t_n} d_{y,x}^{(t)}}{n} \ , \tag{2.27}$$

where n is the number of frames from which the initial background model is created.

During run time, the depth value of each pixel $d_{y,x}^{(t)}$ at a time $t > t_n$ is compared to the background

$$\Delta d_{y,x}^{(t)} = |d_{y,x}^{(t)} - b_{y,x}^{(t)}| \ . \tag{2.28}$$

If this value is lower then a set threshold value ϵ, the pixel p is considered background. If it instead is larger than the threshold value, it is classified as foreground. If a pixel is considered to be part of the background we update the background value as

$$b_{y,x} = b_{y,x} \cdot (1 - \alpha) + d_{y,x}^{(t)} \cdot \alpha \ . \tag{2.29}$$

Where α is a small weight. This ensures that the background will adapt to small changes, increasing the accuracy of the model.

A pixel which is classified as part of the foreground is also weighted into the new background according to the same principle, but with a much smaller weight. This is to gradually include objects which have been static for a long time span.

2.6.2 Gaussian Mixture Model

A Mixture Model is a probabilistic model to represent a distribution consisting of several sub-distributions. In a Gaussian Mixture Model (GMM), the distribution is, as the name suggests, a superposition of M gaussian distributions or components, each with unique mean μ_m, standard deviation σ_m and mixture weights π_m. The weights have the properties of being non-negative and that they sum up to one.

For our pixel based background model, the decision R if a pixel at a time t belongs to the background (BG) or foreground (FG) is [9], [10]

$$R = \frac{p(BG|x_t)}{p(FG|x_t)} = \frac{p(x_t|BG)p(BG)}{p(x_t|FG)p(FG)} \ . \tag{2.30}$$

If we assume that nothing is known about the foreground objects, the priors $p(BG)$ and $p(FG)$ are set to be equal and the distribution of foreground objects is considered uniform $p(x_t|FG) = c_{FG}$. This leads to the threshold based formula

$$p(x_t|FG) > R \cdot c_{FG} = c_{thr} \ , \tag{2.31}$$

with the threshold c_{thr}.

This algorithm can be viewed as an on-line clustering algorithm, where foreground objects will be represented by the components with the smallest mixture weights $\hat{\pi}_m$. This means that by choosing the components with the B largest weights, we can approximate the background to consist of pixels belonging to these clusters [10]

$$p(x|X_T, BG) \sim \sum_{m=1}^{B} \hat{\pi}_m \mathcal{N}(x; \hat{\mu}_m, \sigma_m^2 I) \ , \tag{2.32}$$

for a training set X_T. If we keep the components sorted by their respective weights $\hat{\pi}_1 \geq \ldots \geq \hat{\pi}_M$, this corresponds to solving the equation

$$B = \arg\min_b \left(\sum_{m=1}^{b} \hat{\pi}_m > (1 - c_f) \right) \ , \tag{2.33}$$

where c_f is the maximal proportion of the image, which is allowed to belong to the foreground.

2.7 Noise reduction

The raw depth map obtained from a TOF camera is contaminated with a considerable amount of noise. In [7], Frank et al. argues that the noise can be modelled as an offset normal distribution of the phase shift of the signal. They later show that this is a good estimate and derive that the variance of the depth distribution in a pixel is proportionate to the inverse of the amplitude squared:

$$\sigma_d^2 \propto \frac{1}{A^2} \ . \tag{2.34}$$

This would suggest that the amplitude is a good measure of the confidence of the collected depth data. Another advantage of using the amplitude for confidence measurements, is the fact that it is provided with little more effort from the camera.

2.7.1 Weighted Gaussian filter

A simple way of implementing the amplitude as a confidence smoothing of the depth map is to combine it with a spatial filter, e.g. convolving with a Gaussian kernel [8]. This provides a filter which takes into account the spatial relationship of pixels, weighted with their respective confidence.

The resulting smoothed depth value $d_{h;(x,y)}$ of a pixel (x, y) can thus be calculated from the raw depth value d as

$$d_{h;(x,y)} = \frac{\sum_{k_x=-\frac{n-1}{2}}^{\frac{n-1}{2}} \sum_{k_y=-\frac{n-1}{2}}^{\frac{n-1}{2}} d_{x+k_x,y+k_y} \cdot G^h_{k_x,k_y} \cdot A^2_{x+k_x,y+k_y}}{\sum_{k_x=-\frac{n-1}{2}}^{\frac{n-1}{2}} \sum_{k_y=-\frac{n-1}{2}}^{\frac{n-1}{2}} G^h_{k_x,k_y} \cdot A^2_{x+k_x,y+k_y}} \quad , \tag{2.35}$$

where G^h is the Gaussian kernel with bandwidth h and n is an odd mask size of three or larger.

2.7.2 Confidence modified background model

One of the major flaws of the threshold background model is the artifacts that occur because of light scattering (see 2.2). This corrupts the measurements for large portions of the images, which manifests in large erroneous foreground patches, see Figure 2.8.

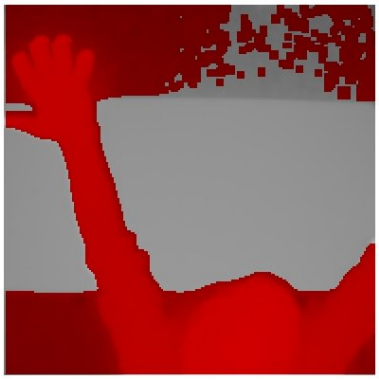

Figure 2.8: Erronous foreground classification due to light scattering. Red areas are foreground.

Utilizing the fact from [8] that the amplitude is a good confidence measurement, we modify the background model to incorporate the standard deviation of the depth map noise. This is done by multiplying the depth difference in equation 2.16 with the pixel's corresponding amplitude value. The amplitudes in the images vary extensively, from very insecure values with an amplitude of around 10, to very accurate measurements with an amplitude of around 40000. To work around these large differences, the logarithm of the amplitude is used instead of its raw value, resulting in the foreground

$$FG = \{(x, y) \in \mathbb{Z}^{m,n} \mid \Delta d_{x,y} \cdot (\log_{10}(A) - w_A) > \epsilon\} \quad , \tag{2.36}$$

where m, n are the dimensions of the images, w_A is a weight to balance the amplitude and distance and ϵ is the set threshold.

2.8 Morphology

Digital images, and specifically, binary images can be manipulated using the morphological operations erosion and dilation in order to shrink or grow objects in the image.

2.8.1 Binary erosion

Binary erosion is one of the two fundamental morphological operations. The basic idea of it is to shrink objects by removing edge pixels, both on outer and inner edges, such as holes in an object. Erosion of the binary set $A \subseteq \mathbb{Z}^2$ by its structuring element $B \subseteq \mathbb{Z}^2$ centered at the point $(i, j) \in A$ is defined by

$$A \ominus B = \{(i, j) \in \mathbb{Z}^2 | B \subset A\} \ . \tag{2.37}$$

The erosion of A by B is the set of all points in $\mathbb{Z}^2$ for which B centered at that point is fully contained within A.

Erosion of a circle can be seen in the Figure below.

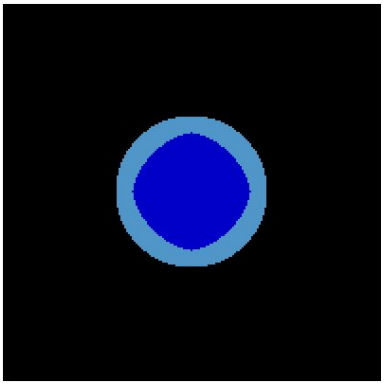

Figure 2.9: Original (light blue), and resulting (blue) Figure after erosion with a square structuring element.

2.8.2 Binary dilation

The other fundamental operation of morphology is dilation. It can be seen as the complement of the erosion operation in the sense that it has somewhat the opposite effect. Dilation can in fact also be defined using the erosion of the complement of the image, A^c as

$$A \oplus B = (A^c \ominus B)^c \ . \tag{2.38}$$

There is an alternative definition of dilation, which for every symmetric structuring element can be simplified as

$$A \oplus B = \bigcup_{b \in B} A_b \ . \tag{2.39}$$

The dilation of A by B is the set of all points covered by B whilst the center of B moves inside A.

The effect of dilation can be seen in Figure 2.10 below.

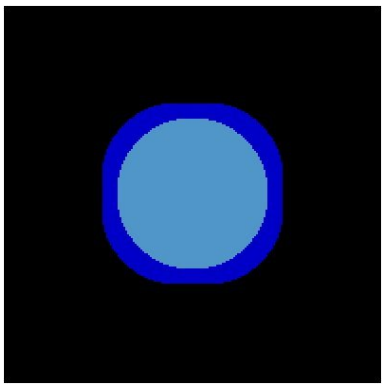

Figure 2.10: Original (light blue), and resulting (blue) figure after dilation with a square structuring element.

2.8.3 Opening and closing

By combining erosion and dilation, one can define the two operations,*opening* (ψ_B) and *closing* (ϕ_B) of A defined by

$$\psi_B(A) = (A \ominus B) \oplus B \tag{2.40}$$

and

$$\phi_B(A) = (A \oplus B) \ominus B \ , \tag{2.41}$$

i.e. the erosion followed by dilation, or dilation followed by erosion respectively.

Performing a closing on a binary image will close small gaps (see Figure 2.11 and fill small holes in objects while retaining the shape and size of solid objects without close neighbours.

In the same manner, an opening will separate objects with thin joints (see Figure 4.10) and remove small enough objects. This can be used to remove noise in the foreground mask.

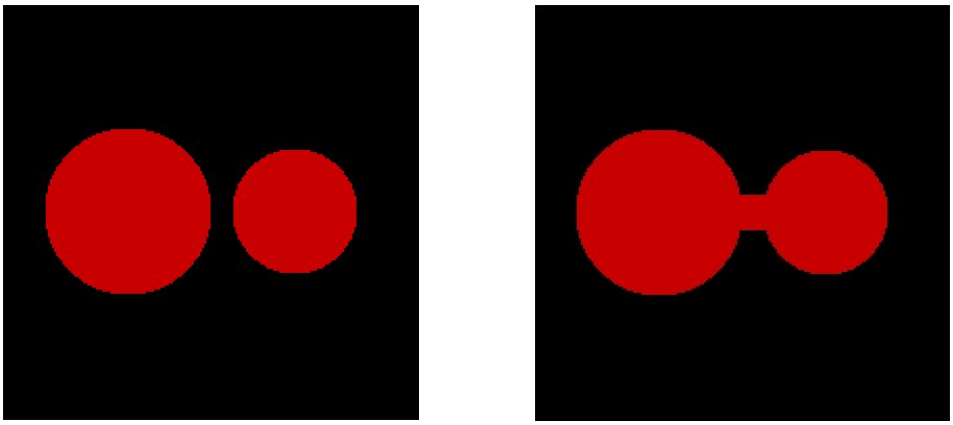

Figure 2.11: Objects before performing a closing (left) and after (right).

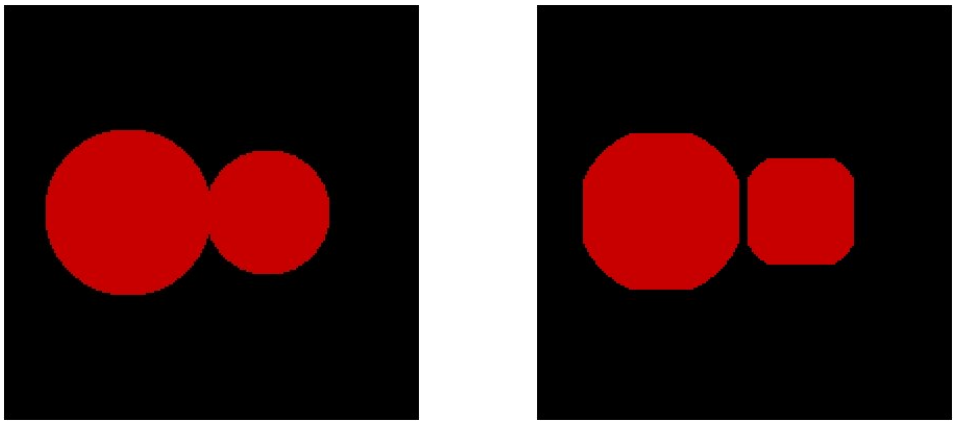

Figure 2.12: Object before performing an opening (left) and after (right).

2.9 Object detection

For the detection and separation of objects in images, we developed a simple recursive scheme. It is based around the assumption that neighbouring pixels in the same object should have approximately the same depth value. The algorithm begins by scanning the image until a foreground pixel is detected after which the recursive function starts.

The recursive method marks the current pixel as visited, calculates the distances to its neighbours and compares them to a defined threshold

$$\Delta d = d(x, y) - d(x_n, y_n) < \epsilon_{thresh}, \qquad (x_n, y_n) \in \mathcal{N}_4 \ , \tag{2.42}$$

where $\mathcal{N}_4$ is the 4-neighbourhood of the pixel (x, y). If (x_n, y_n) is close enough, the function runs recursively for that pixel.

If the resulting number of classified pixels are too few, the resulting object is noted as being noise. Otherwise the number of detected objects is increased. After this, the scanning procedure continues until no more foreground can be detected.

Chapter 3

Implementation

3.1 Algorithm - intersection with planes

In this section the proposed algorithm will be presented in full with all the necessary pre-processing of the video stream.

3.1.1 Background

The first step is to implement a reliable background model. This is done in the manner described in section 2.6.1.

3.1.2 Plane fitting

The second step of the algorithm is to detect the counter edges and to calculate the plane equations. By convolving with a y-direction Sobel filter (section 2.7.1), the top and bottom edges are detected. After systematically trying to fit a plane to the proposed edge points using RANSAC, a best choice model is chosen. Using the normal to this plane and two inliers on the inner counter edge, a second plane can be fitted to the inside.

3.1.3 Foreground segmentation

The method we used is the confidence modified threshold model presented in sections 2.3.1 and 2.4.2. While the GMM model should be a suitable option, we found that the threshold model provided more stable results with less noise. This model was implemented with different thresholds for various areas of the image. Wherever the background is close to a relevant object, the threshold is set to be significantly lower. This is because a person stretching over the counter can vanish from the foreground if pressing closely to it. Similarly, because of the higher amount of noise in areas far from the camera,

the foreground model will be severely affected by this if the threshold is set too low for those parts, see figures 3.1 through 3.3.

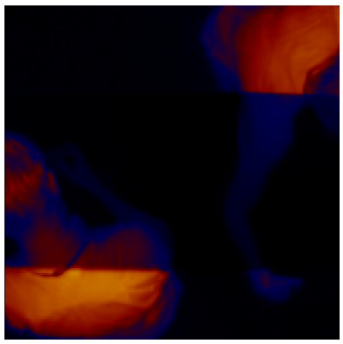

Figure 3.1: Background model subtracted from an image in a video sequence. It is clear how the area above the counter needs a lower threshold to be discerned from the background.

Figure 3.2: Foreground segmentation where a high threshold was used for the entire image. Parts of the objects close to the counter are merged with the background.

Figure 3.3: Foreground segmentation where a low threshold was used for the entire image. Areas over the counter are accurately segmented whereas the noise around it is unmistakably higher.

3.1.4 Noise reduction

In order to further reduce the noise in the images both for the background model and the video stream, the weighted Gaussian filter in 2.4.1 is implemented. After this, the binary foreground image is refined using a morphological opening with a small square structuring element (see section 2.5).

3.1.5 Object detection

The last part before the theft detection is object detection. This is done using the recursive classifier presented in section 2.9. The classifier saves data about the classified object such as its bounding box, center of mass, mass (number of pixels) and class. This is then used later in the algorithm when suspicious behaviour is detected.

3.1.6 Thief detection

In order to detect thefts we chose to check pixels for a single main criteria, namely being on the wrong side of both planes fitted to the counter. If a pixel is considered to be a possible theft, the algorithm checks to see if the object corresponding to that pixel has its center of mass on the inside or outside of the counter. If it is located on the inside of the counter the object is considered to be the cashier and is thus not a thief. Otherwise the pixel is still considered suspicious. This is done for all foreground pixels after which an image is created with possible thefts marked as 1 possible thefts and 0 for pixels marked as harmless.

This binary image is then processed with an opening with a rectangular structuring element. This is done in order to reduce the amount of remaining noise around the edges of objects which could trigger a false alarm.

If there still exist pixels which are marked as a theft, the algorithm will try to separate the objects in the image. This is done because certain situations could falsely trigger the alarm. These situations can be when the cashier shakes hand with or hands items to the customer. This would merge the cashier object with the customer object. By using a morphological opening with a wide structuring element with a size of about the same magnitude as the width of an arm, possible merged objects are separated by "cutting off" arms or otherwise thin parts of objects. The size and shape of this structuring element was individually decided empirically for all three cameras due to different resolution and field of view. The only thing left to do is to count the objects left in the image with the same class as the suspected pixel. If this number is greater than one, the theft alarm is disregarded. If, however, this number equates to one, the object is considered to be a thief, and the alarm is triggered. This can be seen in Figure 4.18.

3.2 Test environment

When designing a test environment, it is important to make sure that it is as general as possible for it to be a suitable model for more than one store. This enables for a more rigorous algorithm which can adapt to many situations in different (but similar) retail environments.

With this in mind we chose to work with two different types of surfaces, one with hardened glass and one with a white coat. After studying several jewelry and perfume shops, we could conclude that both these are frequently used in this type of environment, with the hardened glass counter being the most common one.

The lighting is modeled using moderate in-door light with the addition of three spotlights positioned in the ceiling straight above the counter. The camera is positioned in the ceiling above and slightly behind the counter. The setup can be seen in Figure 3.4 below.

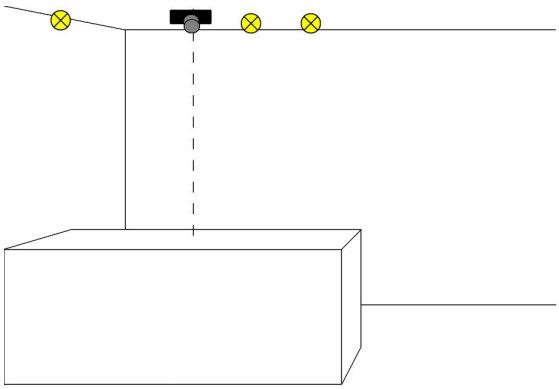

Figure 3.4: Test environment with the camera above with spotlights fitted to the ceiling above the counter.

The positioning of the camera is of great importance. Firstly, it is vital to get a good picture of as much of the relevant area as possible. Secondly, it can have implications on the design of the algorithm where the cameras is positioned. Thirdly, the problem with the wrapping distance of a TOF cameras can be troublesome if the camera is poorly positioned, i.e. if it is positioned so that some parts of the captured images are further away than the wrapping distance (see section 2.1.1).

3.2.1 Glass counter problem

The biggest issue with the algorithm is how it works with glass counters. As the tempered glass lets through all but a fraction of the near infrared light from the TOF camera, the counter is near impossible to detect with that technique. This means that the vital part of the algorithm with edge detection and plane fitting will not work. In our testing, we have made the infringement of the environment that the glass counter rests upon an edge of sort. This means that the edge detector will be able to detect this small border which in turn will enable the rest of the algorithm.

3.3 Data sets

Every set of test environment constellation is recorded in a series of scenarios with and without a theft. In order to vary the data as much as possible, the scenarios are divided into two main categories: with and without the presence of a cashier. These are then divided into more branches of different possible events, which are recorded a number of times in order to minimise the possibility for circumstantial errors and misses.

3.3.1 Without cashier

The only recorded event without the presence of a cashier is the case of a lone thief reaching for something below the counter:

Theft

A. Lone thief reaching over the counter, recorded 5 times.

3.3.2 With cashier

In the case of a cashier, two types of scenarios are tested: with or without a theft. These can then be divided into a few sub cases:

Theft

B. The cashier is busy talking to a customer, recorded 3 times.

C. The cashier is occupied with something else, e.g. unpacking an item, recorded 2 times.

No theft

D. The customer shakes hand with the cashier, recorded 2 times.

E. The customer puts something on the counter, recorded 2 times.

F. The customer points and makes vivid gestures towards something behind the counter, recorded 2 times.

G. The cashier is working alone without the presence of a customer, recorded 2 times.

3.4 Cameras

3.4.1 TOF cameras

The two TOF cameras tested in this thesis are the PMD CamCube 3.0 and the Soft Kinetic DS311. Their respective technical specifications can be seen in the table below, directly retrieved from the official data sheets [3] and [4].

	PMD CamCube	Soft Kinetic DS311
Depth resolution	200 × 200 pxl	160 × 120 pxl
Secondary picture type	Intensity (gray level)	RGB
Secondary resolution	200 × 200 pxl	640 × 480 pxl
Field of View	40° × 40°	57.3° × 42°
Illumination wavelength	870 nm	850 nm
Frames per second	40 - 60 fps	25-60 fps
Range	0.3 - 7 m	1 - 4 m

(a) PMD Camcube 3.0, [3]. (b) Smart Kinetic DS311, [4].

Figure 3.5: TOF cameras used in this thesis.

3.4.2 Structured light camera

The Microsoft Kinect is the structured light method camera used in this thesis. Its resolution of 640×480 pixels is much higher than that of the ToF cameras, but it has a shadowing problem (Figure 2.4). The Kinect also provides an RGB image at a resolution of 1600×1200 pixels. It works in up to 60 FPS and operates on the range 0.8 to 3.5 meters. The precision of the depth map at 2 meters is 1 cm. The depth map precision is provided in 11-bits, i.e. ranges from 0 to 2047.

Figure 3.6: Microsoft Kinect used in this thesis.

Chapter 4

Results

The results will in this section be presented for each step in our proposed algorithm. In cases where the results can be compared to old results or with a reference image of raw data, these will be presented as well but discussed in the next chapter.

4.1 Edge detection

Sobel filtered background models of the white coated and glass counter (Figure 4.1), and the resulting edges from our edge detection (Figure 4.2).

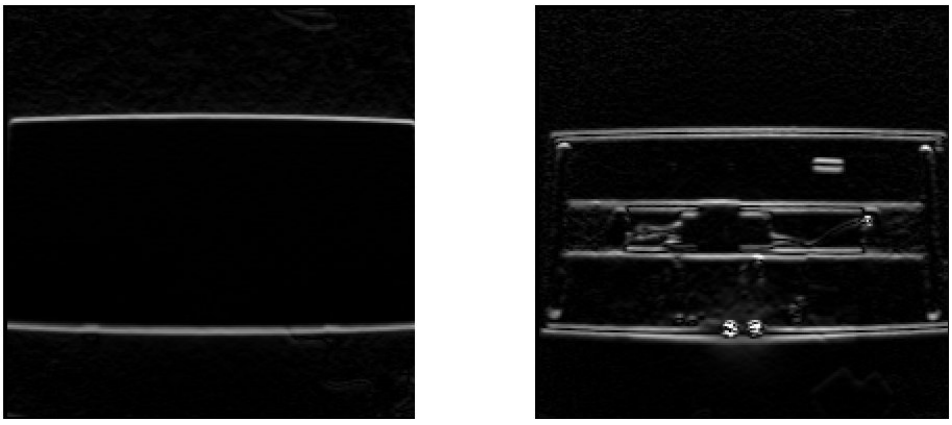

Figure 4.1: Sobel filtered background images used to detect the edge of the counter. White coated counter to the left and glass counter to the right.

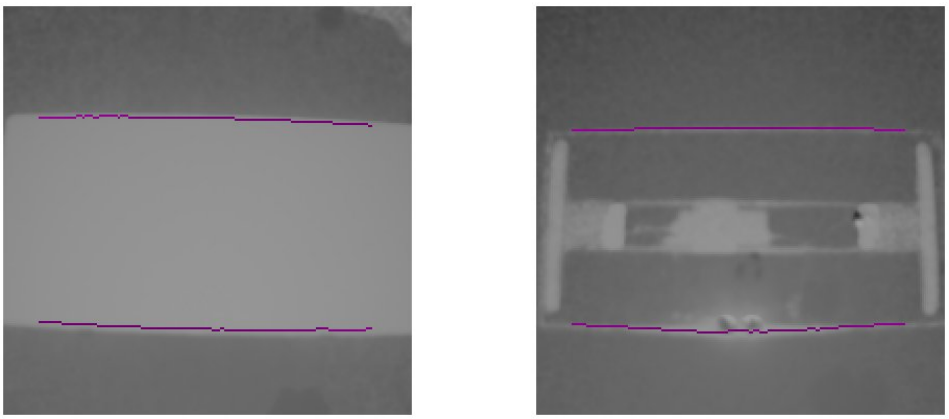

Figure 4.2: Detection of counter edge on the white counter (left) and glass counter (right).

The edge detection works similarly for the structured light camera, with the exception of the glass counter which introduces a lot of blank areas where the camera cannot obtain a depth value, see Figure 4.3

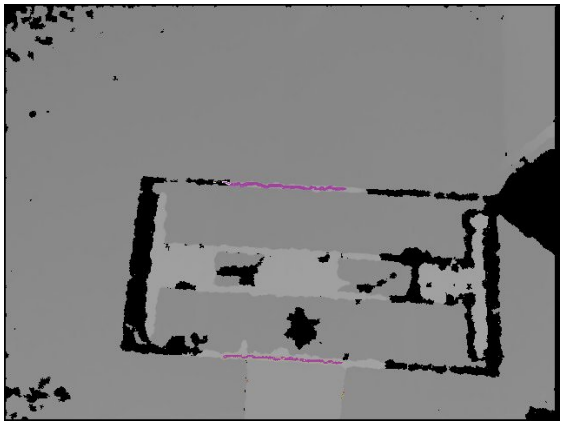

Figure 4.3: Detection of counter edge with the structured light camera on a glass counter. Black pixels are areas without any available depth data.

The RANSAC method ensures that the correct points (inliers) are used in the plane fitting part of our algorithm even though a large amount of outliers are present, see Figure 4.4

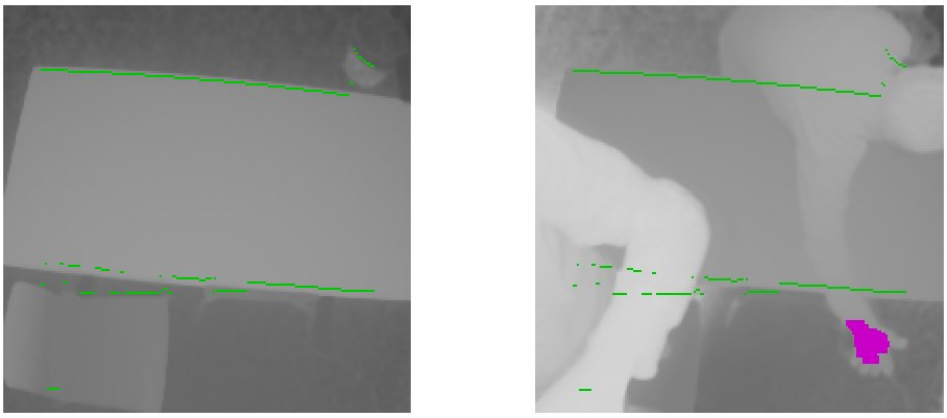

Figure 4.4: Edge detection with a lot of outliers due to a cluttered environment. Left image shows the detected edges and the right image shows that the plane is fitted correctly despite the large number of outliers.

Figure 4.5 below, shows one of the video clips from the structured light sequence where the algorithm fails to detect the top edge of the counter. This leads to an erronous plane as can be seen by the green classification of the cashier.

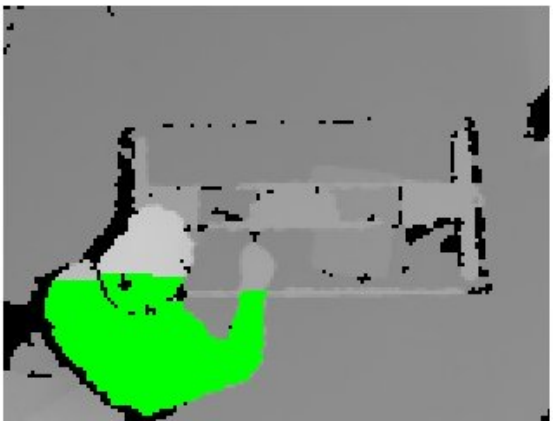

Figure 4.5: Failure to fit correct plane to the glass counter.

4.2 Foreground segmentation and noise reduction

Here, the resulting foreground segmentation will be presented along with the results from the noise reducing methods.

In Figure 4.6 below, the foreground detection using a simple distance subtraction method with a single threshold without any other noise reduction can be seen.

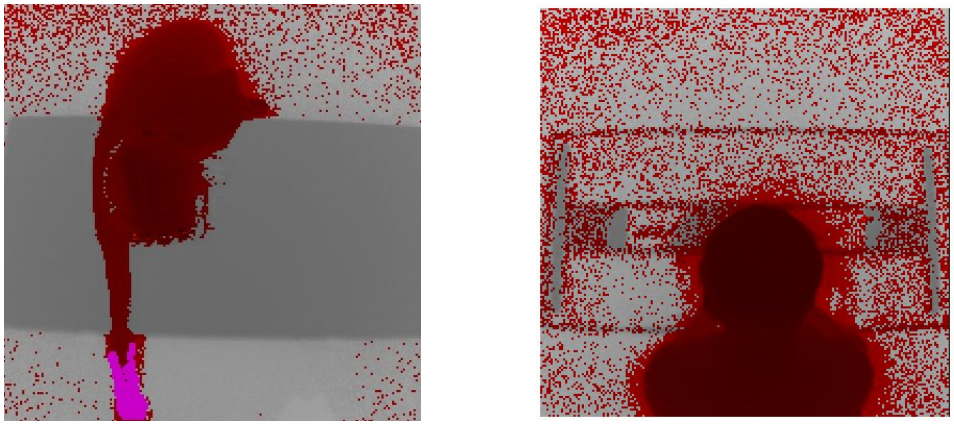

Figure 4.6: Simple foreground segmentation of the raw image data on the white coated counter (left) and glass counter (right).

The resulting filtered images after using the weighted spatial filter presented in section 2.7.1, compared to the unfiltered images can be seen in Figure 4.7 through 4.8.

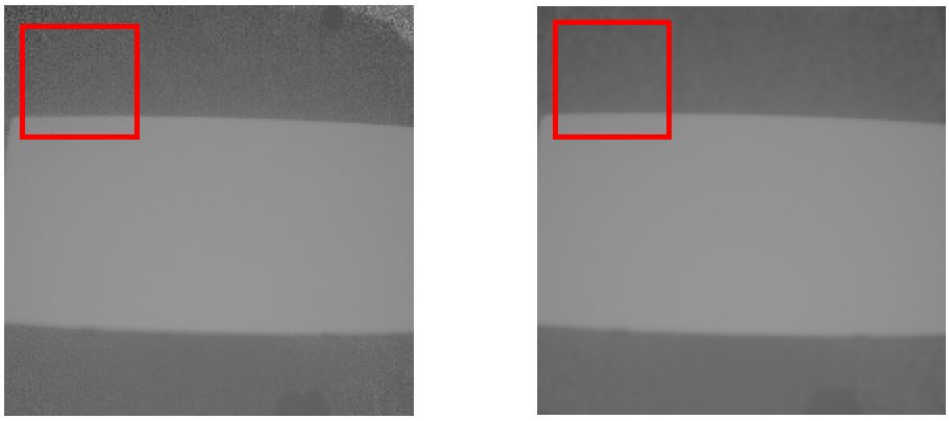

Figure 4.7: Noisy image without (left) and with (right) weighted spatial filter.

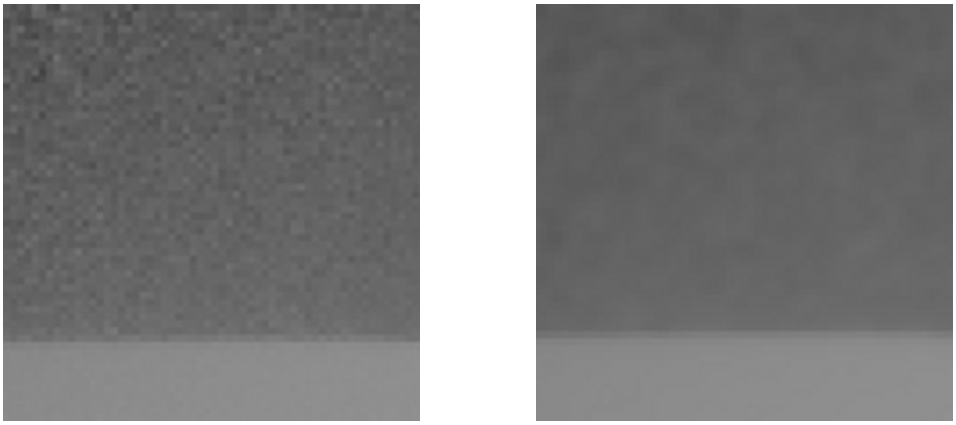

Figure 4.8: Zoomed window of the above image with unfiltered image (left) and filtered image (right).

Below (Figure 4.9) is a comparison of the background model without the confidence weight modification with the modified version of the method.

The result of the morphological opening on the foreground can be seen in Figure 4.10.

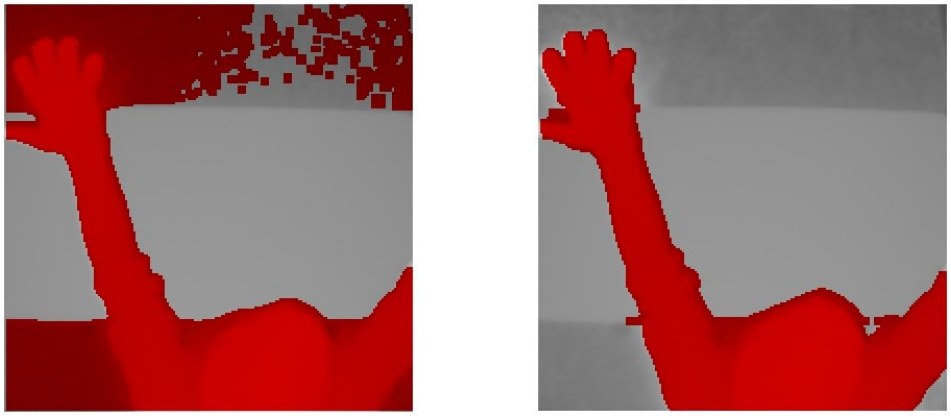

Figure 4.9: Foreground segmentation without (left) and with (right) confidence weighting.

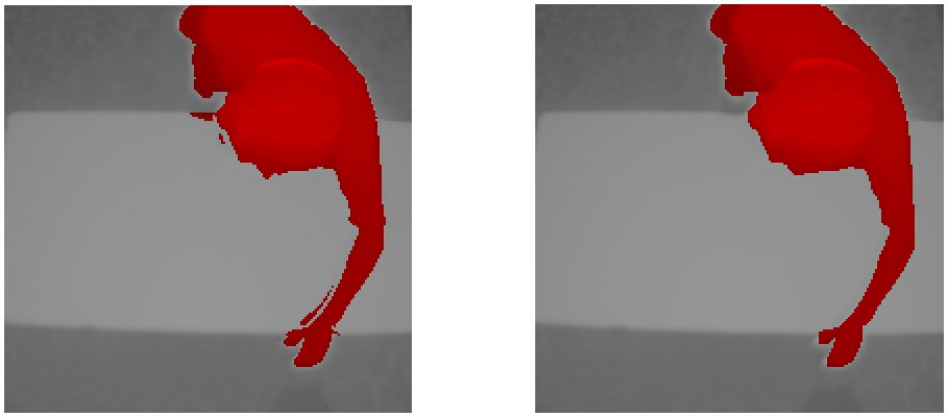

Figure 4.10: Foreground segmentation without (left) and with (right) morphological opening.

Finally, some sample results when running the modified threshold model on the filtered images can be seen in Figure 4.11 below.

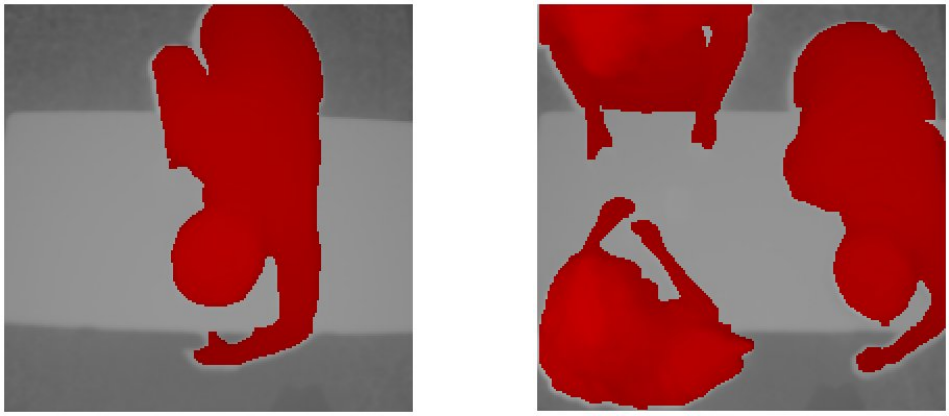

Figure 4.11: Foreground segmentation of a single (left) and multiple objects (right).

4.3 Object detection

Below follows some examples of separately classified objects.

Figure 4.12: Object detection of a single object (left) and multiple objects (right).

4.4 Theft detection

4.4.1 Correctly detected thefts

TOF cameras

Here follows some samples of correctly classified thefts. Purple areas in the images represent pixels classified as thievery behaviour and green pixels are areas which would have been classified as thievery had they not belonged to the cashier.

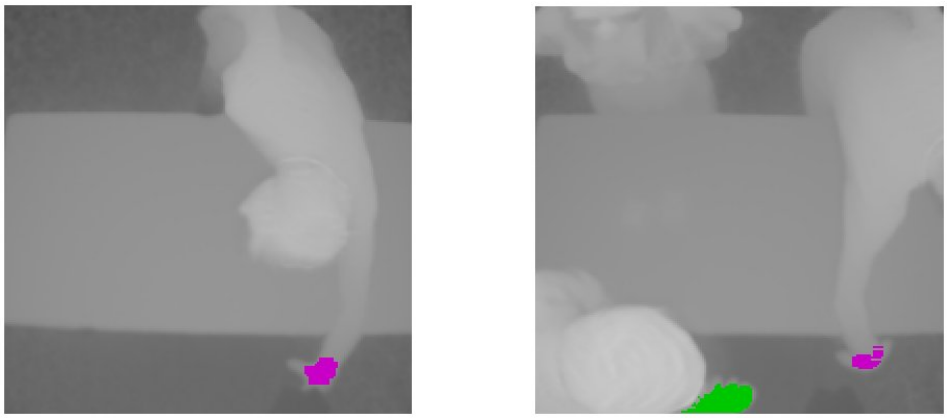

Figure 4.13: Theft detection of a lone thief (left) and thief with (an unquestionably ignorant) cashier (right) with a white coated counter.

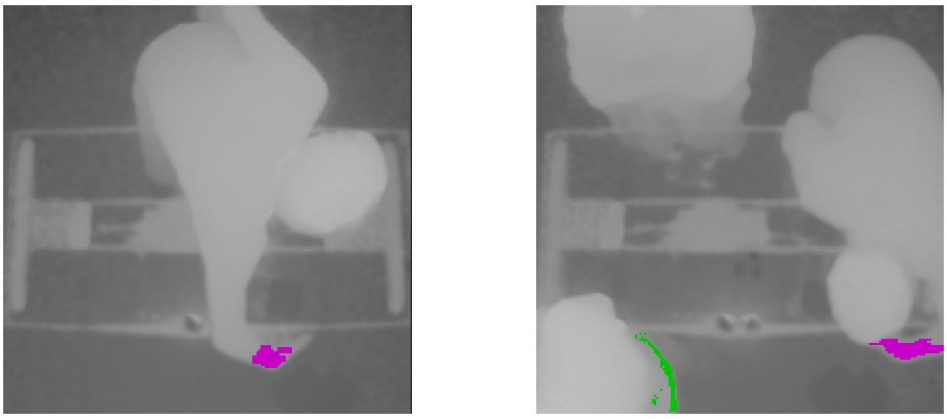

Figure 4.14: Theft detection of a lone thief (left) and thief with cashier (right) with a glass counter.

4.4.2 Missed thefts

During one test clip did the fully implemented algorithm fail to detect a theft. This can be seen in Figure 4.15 where the cashier blocks the view of part of the thief's arm.

Figure 4.15: Algorithm fails to detect a theft because of a blocking cashier.

4.4.3 False alarms

False alarms with TOF cameras are sometimes triggered by noise occurring around the edges of hands and arms. This can somewhat be reduced (see 3.1.2). Below, an example of an avoided false alarm (Figure 4.16) and a false alarm which could not be removed (4.17).

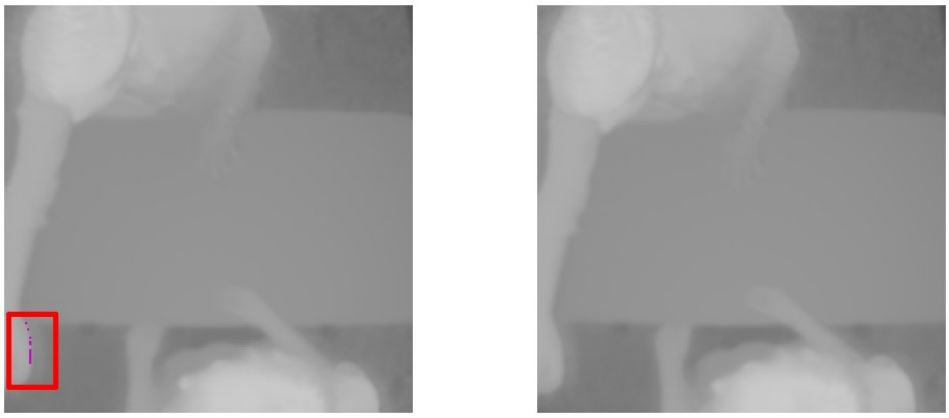

Figure 4.16: Left image shows a false alarm due to noise along the edges of the customer's hand. The right image shows the same scenario where the falsely classified pixels have been removed with a morphological opening.

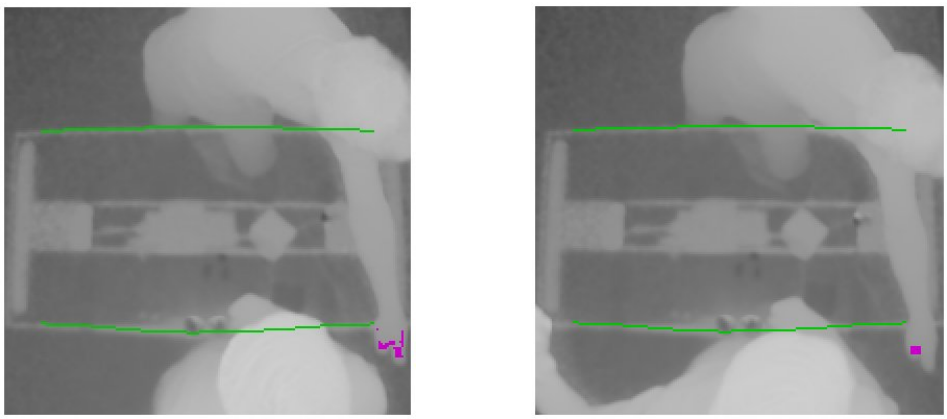

Figure 4.17: Left image shows a false alarm due to noise along the edges of the customer's hand. The right image shows the same scenario where the algorithm has tried to remove the falsely classified pixels but failed due to the heavy amount of noise and the object's proximity to the forbidden volume.

4.4.4 Object splitting

The results of the large morphological opening on theft mask when a possible theft is detected can be seen in Figure 4.18 below.

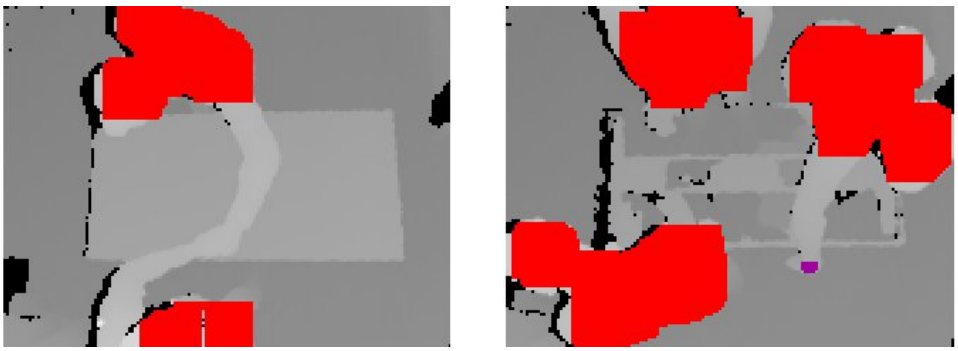

Figure 4.18: Splitting of objects with a large opening on the binary thief mask. To the left, a false alarm is avoided because of this and to the right a theft is correctly reported. Videos recorded with the Microsoft Kinect.

4.4.5 Detection results

Here follows the results of the full algorithm. As a comparison, the results for the algorithm without selected parts of the pre-processing, and entirely without noise reduction (raw data) is presented. For more detailed accounting of each recorded scenario, see Appendix B.

It is worth noting that the structured light camera does not deliver amplitude values, as this is a strictly TOF related attribute. Therefore, amplitude based steps are not included in the test results for the Kinect.

PMD CamCube 3.0

Full algorithm

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	2/8	16/18

Algorithm on raw data

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	9/10	2/8	15/18
Glass counter	8/10	2/8	14/18

Algorithm without opening on foreground

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	1/8	17/18
Glass counter	9/10	2/8	15/18

Algorithm without opening on binary thief image

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	1/8	17/18
Glass counter	10/10	2/8	16/18

Algorithm without modified background model

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	1/8	17/18
Glass counter	9/10	3/8	14/18

Algorithm without amplitude weighted spatial filter

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	1/8	17/18
Glass counter	10/10	2/8	16/18

Algorithm without object splitting

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	2/8	16/18

Soft Kinetic DS311

Full algorithm

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	1/8	17/18

Algorithm on raw data

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	2/8	16/18
Glass counter	10/10	1/8	17/18

Algorithm without opening on foreground

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	1/8	17/18

Algorithm without opening on binary thief image

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	1/8	17/18
Glass counter	10/10	1/8	17/18

Algorithm without modified background model

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	2/8	16/18
Glass counter	10/10	1/8	17/18

Algorithm without amplitude weighted spatial filter

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	1/8	17/18

Algorithm without object splitting

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	1/8	17/18

Microsoft Kinect

Full algorithm

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	0/8	18/18

Algorithm on raw data

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	4/8	14/18
Glass counter	10/10	6/8	12/18

Algorithm without opening on foreground

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	0/8	18/18
Glass counter	10/10	0/8	18/18

Algorithm without opening on binary thief image

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	3/8	15/18
Glass counter	10/10	2/8	16/18

Algorithm without object splitting

Counter type	Theft detection rate	False alarm rate	Total correct
White coated counter	10/10	1/8	17/18
Glass counter	10/10	2/8	16/18

Chapter 5

Discussion and conclusion

After testing the algorithm with the different cameras, it is time to discuss its strengths and weaknesses. This will be done for each category presented in the result section separately. After this, the final results and the differences in the different cameras will be discussed.

5.1 Edge detection

Detecting the edges, and especially the inner edge of the counter is a vital part of the setup of the algorithm. The proposed method to use a Sobel filter and scan from the top and bottom of this resulting image, G_y, for high values seems to work well. An early plan was to make a least squares fitting of a plane to all of these edge points, but this idea was soon discarded because of the large amount of outliers present in some scenes. These outliers would severely corrupt the resulting top plane as least squares is unsuitable for models where deviant observations cannot be regarded as a value from the tail of a normal distribution.

Instead of the least squares method, the final edge detection algorithm uses the RANSAC method described in 2.8. This method was chosen despite it only providing a good model with a certain probability. This is because its rigidity to a large amount of outliers when iterated enough times is a very important aspect when the environment is not precisely known for every unique shop. The rigidity was tested by trying the algorithm on different odd environment setups where the cluttered floor around the counter provided us with a large amount of outliers. As seen in Figure 4.8, the algorithm proved to work even in these demanding situations.

A possible cause for alert is the amount of dead areas around the counter edges on the glass counter when using a structured light camera. This could possibly make the edge detection improbable enough for the RANSAC to

work properly. This has not been a problem while testing the camera, but could possibly be an issue with a more complex environment.

5.2 Foreground segmentation and noise reduction

The foreground segmentation has been a crucial point during the course of this project. A good and solid model works as a good foundation for the rest of the algorithm. If the method detects too much foreground, the risk for false alarms is significantly higher, and a too harsh model risks missing vital foreground. This can in turn lead to missed thefts.

The amplitude modified model used in the end proved to perform well in both ordinary and difficult situations. The presence of a foreground object perturbs the depth measurements for its surroundings due to light scattering, as discussed earlier in section 2.2. This means that the unmodified threshold model detects a foreground object in these perturbations, whereas the modified model does not (see section 2.4.2). This is clearly visualised in Figure 4.4.

The methods inclusion of different thresholds for different parts of the images works (as seen in Figure 3.1 through 3.3), very well. As balancing the threshold parameters is a fine line between minimising noise and keeping the objects discernible from the background, it can be very difficult to set a value that will work perfectly in both these aspects. Because of this some objects, such as arms, are sometimes merged into the counter. This has not proven to be a significant problem yet, as the rather large number of frames during which a theft occurs, means that the risk of missing a theft in every frame is very small.

The effects of the morphological opening on the foreground image is also clear, as is the smoothing of the amplitude weighted spatial filter. Even though some noise in the segmentation is not so bad, every attempt to minimise it should be an improvement. When inspecting the result tables in 4.4.3, some of these noise reductions do not seem to alter the resulting accuracy of the algorithm on the recorded videos. This however is a modified truth. When trying the different versions of the algorithm live, the noise reducing methods seem to stabilise the algorithm and reducing its tendencies to trigger false alarms, whilst not reducing the theft detection rate. In addition, these sub algorithms complement each other in the sense that removing two or more, significantly lowers the hit rate of the method.

5.3 Object detection

The object detection part of the algorithm works well despite its simplicity. Because of its elementary nature, it does not track objects between frames, and does have problems with it merging different objects. This does however not seem to pose a real trouble for the accuracy of the algorithm. Object tracking would make it considerably easier to update the background and merge static objects into the background, but this has not caused any problems in the test scenarios, nor in any live testing we have done. Object tracking could also make more intelligent separation of objects possible and enable for more stable disregarding of any suspect behaviour caused by the cashier.

All these factors could provide for a more rigid algorithm which could adapt to further more difficult situations that the authors have not thought of or tested, but as the accuracy of the algorithm is very good as it is, this seems excessive.

5.4 Theft detection

Theft detection using the intersection with two planes seem to work very well. The method is stable and seems to only fail under questionably realistic circumstances where the customer waves frantically over the counter, is very close to the prohibited areas, or is obscured by a present cashier. The separation of objects described in section 3.1.2 may seem crude but almost entirely removes the risk of false alarms when the cashier object is merged with a customer object by for example shaking its hand or handing it money.

Its most obvious flaws are its dependence on the specific retail environment and that the edge detection works, as well as its inability to find glass surfaces. The first two problems have been minimized by making the algorithm more robust to changes and deviant environment setups, but the latter problem is something we have not been able to remedy. We instead imagine a work around of the issue, where calibration lists are fitted to the edges of the counter during the camera setup. These lists can be removed once the background model has been created, after which the algorithm will work as intended.

5.4.1 TOF cameras

Due to the low signal-to-noise ratio, the pre-processing parts of the algorithm compose a vital part of the finished method. Some of them, such as the amplitude modified background model, and the morphological operations on the foreground mask, directly improves the result on our test sequences

whereas some don't. It is crucial to point out that even though these do not directly affect the result of the tests, the empiric testing on live situations have shown a much more stable and reliable behaviour when processed with for example the amplitude weighted filter.

The results for the unprocessed video sequences show a great difference between the DS311 and the CamCube. This is because the SmartKinetic device has a built-in temporal filter which reduces nose, whereas the PMD camera gives us the raw data.

5.4.2 Structured light camera

The largest camera specific issue with the structured light camera, is its tendency for missing depth data in quite large areas of the image. This is something that could lead to detection failure once presented to a non artificial environment where obstructing items could hide larger portions of the disk.

Another problem with these dead areas, is how to incorporate them into the background model. The issue arises when an area which has previously been dead suddenly is not. This can happen for a couple of reasons and be handled in a few different ways, firstly if a foreground object is introduced between this area and the camera. In this situation, it would be good to correctly classify it as a foreground object. This however can mean trouble in the second scenario, when moving a background object that has obstructed the view of the structured light pattern. This can cause overly large faulty foreground objects if handled in the same way.

The amount of noise present in the videos captured with the Microsoft Kinect seems very low compared to the TOF cameras used. This is probably because the product contains noise reducing algorithms developed by Microsoft. However, as the software is protected, we have not found any information about the pre-processing done in the device

5.5 Future work

The algorithm as such can be improved somewhat by different means. For example, the ability to track objects could prove useful and might stabilise the algorithm. The ability to adapt to different complex environments, for example with oddly shaped counters with curved sides or more than one level is another aspect that has been left untouched due to the scope of this thesis. Generally speaking, our algorithm contains quite a few hard coded parameter values, such as integration time, thresholds, etc. For a more robust implementation, some work could be done to automatically scale these

values to adapt to different environments.

The problem with missing data from the structured light camera is an issue which could be handled using spatial statistic algorithms. These should with relatively high probability reconstruct the dead areas using the information obtained from surrounding pixels.

What seems like the most pressing issue however are the cameras themselves. The low resolution combined with a very narrow field of view limits the application of today's TOF cameras to either working with small counters or by using multiple cameras. If working with several cameras, a new problem arises where the modulated light from neighbouring cameras interfere with the depth acquisition.

5.6 Summary

We have in this thesis designed an algorithm for detecting thefts over a counter in retail environment. The proposed method handles all parts from image processing with noise reduction, to image analysis with object detection and theft detection. The resulting algorithm could be used in an embedded system camera and work as an alerting system which warns present personnel of the potential theft.

Evaluation of the cameras in terms of deciding which technique is best suited for this purpose is difficult. Partly because TOF and structured light cameras have unique pros and cons, but also because of the different states the cameras are delivered in. We can, however, conclude that the algorithm (with the addition of the TOF noise reduction), is robust enough to handle both techniques very well. This means that future developing of the method does not have to be limited to only one of these technologies, but can be fitted to which one seems the most promising in the future.

Bibliography

[1] Axis Communication AB, *Financial Statement 2011*

[2] Dr.-Ing. C. Schaller, *Time-of-Flight A New Modality for Radiotherapy*, University of Erlangen-Nürnberg

[3] PMD Technologies, *http://www.pmdtec.com/*

[4] Soft Kinetic, *http://www.softkinetic.com/*

[5] Dr.-Ing. T. Ringbeck et al. *A 3D Time Of Flight Camera for Object Detection*, PMD Technologies.

[6] M. Profittlich, *CamCube Software Development Tutorial*, PMD Technologies

[7] M. Frank et al.*Theoretical and Experimental Error Analysis of Continuous-Wave Time-Of-Flight Range Cameras*, Department of Computer Science, ETH, Zurich.

[8] M. Frank et al. *Denoising of continuous-wave time-of-flight depth images using confidence measures*, Department of Computer Science, ETH, Zurich.

[9] A. Störmer et al. *Depth Gradient Based Segmentation of Overlapping Foreground Objects in Range Images*, Bertrand Ingenieurbüro

[10] Z. Zivkovic, *Improved Adaptive Gaussian Mixture Model for Background Subtraction*, Intelligent and Autonomous Systems Group, University of Amsterdam.

[11] S. Fuchs et al. *Extrinsic and Depth Calibration of ToF-cameras*, DLR, Institute of Robotics and Mechatronics, Oberpfaffenhofen, Germany.

[12] S. May et al. *3D time-of-flight cameras for mobile robotics, Real-time scattering compensation for time-of-flight camera*, Schloss Birlinghoven, D-53754 Sankt Augustin, Germany.

[13] S. Foix et al. *Lock-in Time-of-Flight (TOF) Cameras: A Survey*, IEEE Sensors Journal, vol. 11, no. 3, March 2011.

[14] J. Mure-Dubois et al. *Real-time scattering compensation for time-of-flight camera*, University of Neuchatel, Institute of Microtechnology, 2000 Neuchatel, Switzerland.

[15] Martin A. Fischler et al. *Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartography*, SRI International.

[16] D. Fofi et al. *A Comparative Survey on Invicible Structured Light*, IUT Le Creusot, France.

[17] V. Castaneda, N. Navab. *Time-of-Flight and Kinect imaging*, Computer Aided Medical Procedures, Technische Universität München, Germany.

[18] *http://en.wikipedia.org/wiki/Structured-light_3D_scanner*

[19] A. K. Mishra et al. *3D Surveillance System Using Multiple Cameras*, Department of Electrical and Computer Engineering, National University of Singapore.

Appendix A

Retail Environment Pictures

Appendix B

Detailed test results

Here the complete test results are presented for all cameras and for each scenario described in section 3.3.

B.1 PMD CamCube

B.1.1 Full algorithm

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.1.2 Algorithm on raw data

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	2/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.1.3 Algorithm without opening on foreground

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.1.4 Algorithm without opening on thief mask

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	1/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.1.5 Algorithm without modified background model

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	1/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	4/5	-
B	3/3	-
C	2/2	-
D	-	1/2
E	-	1/2
F	-	2/2
G	-	0/2

B.1.6 Algorithm without amplitude weighted filter

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.1.7 Algorithm without object splitting

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.2 Soft Kinetic DS311

B.2.1 Full algorithm

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.2.2 Algorithm on raw data

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.2.3 Algorithm without opening on foreground

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.2.4 Algorithm without opening on thief mask

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	1/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.2.5 Algorithm without modified background model

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	2/2
E	-	2/2
F	-	2/2
G	-	0/2

B.2.6 Algorithm without amplitude weighted filter

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.2.7 Algorithm without object splitting

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	1/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.3 Microsoft Kinect

B.3.1 Full algorithm

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	0/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.3.2 Algorithm on raw data

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	2/2
F	-	2/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	2/2
E	-	2/2
F	-	2/2
G	-	2/2

B.3.3 Algorithm without opening on foreground

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	0/2
G	-	0/2

B.3.4 Algorithm without opening on thief mask

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	1/2
F	-	2/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	0/2
F	-	2/2
G	-	0/2

B.3.5 Algorithm without object splitting

White coated counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	0/2
E	-	1/2
F	-	0/2
G	-	0/2

Glass counter

Scenario	Theft detected	False alarm
A	5/5	-
B	3/3	-
C	2/2	-
D	-	1/2
E	-	1/2
F	-	0/2
G	-	0/2