2013 International Conference on 3D Vision

Depth and Intensity Based Edge Detection in Time-of-Flight Images

Henrik Schäfer, Frank Lenzen, Christoph S. Garbe
HCI, Heidelberg University
Speyererstr. 6, 69115 Heidelberg, Germany
Email: henrik.schaefer@iwr.uni-heidelberg.de

IVCI, Saarland University
Campus E2 1, 66123 Saarbrücken, Germany

***Abstract*—A new approach for edge detection in Time-of-Flight (ToF) depth images is presented. Especially for depth images, accurate edge detection can facilitate many image processing tasks, but rarely any methods for ToF data exist. The proposed algorithm yields highly accurate results through combining edge information both from the intensity and depth image acquired by the imager. The applicability and advantage of the new approach is demonstrated on several recorded scenes and through ToF denoising using adaptive total variation as an application. It is shown that results improve considerably compared to another state-of-the art edge detection algorithm adapted for ToF depth images.**

***Keywords*-depth imaging; Time-of-Flight cameras; edge detection; depth edges; intensity edges; shadow removal; denoising**

I. Introduction

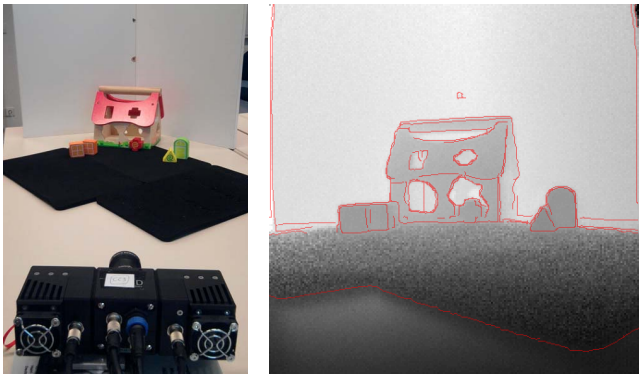


Figure 1. Photograph of test setup (left) and result of our proposed method (right) as overlay to depth data. (All figures contain color. Please refer to the electronic version of the paper.)

Edge detection has always been an important discipline in image processing. Most of the relevant information of natural scenes is contained in the edge maps. This is especially true for depth images, where depth discontinuities separate foreground objects from the background and can be used for many image processing tasks, such as denoising, segmentation or optical flow [6]. Standard edge-detection algorithms work well on scenes with large depth differences, but because of the strong noise of current Time-of-Flight (ToF) cameras, especially in weakly reflecting areas, small changes of depth are often hard to detect.

In this paper we present a new complex approach to edge detection. We adapt Canny's edge detector [2] to make it more robust to noise and apply it to depth and intensity image. The results of the edge detection is significantly improved by selectively fusing edge information from both channels of the ToF camera, which are recorded by the same sensor and therefore have no displacement issues. Problems persisting in these two modalities individually can be successfully resolved with this approach.

Particularly for depth edges hardly observable in the depth image we can significantly improve results from previous approaches. The data is acquired with a PMD CamCube 3 Time-of-Flight camera, but the presented method is not limited to this specific imager. The PMD system consists of two continuously modulated light sources, mounted left and right of the camera lens and suppresses background illumination. The sensor has a resolution of 200x200 pixels.

A. Related Work

Range scanners have been available for a long time. Different approaches have been proposed for edge detection on this type of data (e.g. [4], [8], [1], [15]). But only a few publications exist on edge detection for ToF data, despite strong similarities in terms of application.

Ye and Hegde [19] proposed an algorithm to find straight edges. Lejeune et al. [11] discuss edge detection tailored for ToF cameras and Kinect. They used an implementation of the Canny edge detector [2] that changes sensitivity according to the noise level. In [12] we presented an approach for edge detection in ToF images for denoising the depth data that gave a probability for a present edge, not a binary decision.

Concluding, to the best of the authors knowledge, besides our work the only general edge detection algorithm dedicated to ToF data is by Lejeune et al., adapting the sensitivity of Canny's algorithm but only applying it to the depth data, while we present a method incorporating intensity edges into the process. There is also a number of publications (e.g. [9],

978-0-7695-5067-1/13 $26.00 © 2013 IEEE
DOI 10.1109/3DV.2013.23

[13], [14]) that use input from additional cameras to process the depth data, but only for denoising or upsampling directly, not edge detection.

II. Theory

To achieve a highly accurate edge image, we perform edge detection on depth and intensity image of the ToF camera an then fuse the different results into one final edge map.
We start with the intensity edges and remove shadows and texture edges. Then we add the shadow-casting depth edges and high-threshold depth edges.

A. Edge Detection Algorithm

In order to find the edges we start the analysis with a structure tensor approach [5], [7] with an enhancement proposed by Köthe in [10]: for correct sampling, the point wise multiplication of the tensor elements requires doubling of the frequency of the derivatives. This is done by applying the derivative filters at each full and half pixel position. The square root of the eigenvalues and the corresponding eigenvectors are used as input data for Canny's algorithm [2], instead of the usual gradients.
To better cope with the strong noise of the depth images, we make a few assumptions and extend the hysteresis step. The standard hysteresis of Canny uses two thresholds t_h and t_l to select edges. A connected set of potential edge points above the lower threshold t_l is only accepted if at least one of the points has a value above the higher threshold t_h. Since we know the (normal) direction at each edge pixel, a direction penalty that does not allow sudden changes of the edge direction can be included in the algorithm. This is realized by calculating the scalar product of the two adjacent normal vectors. If the product is above a threshold $t_s \leq 1$, the pixels are part of the same set. This helps us to break up edges that go around a corner and treat them separately later on. Finally, we can remove edges that do not have a minimum length n_m, to get rid of some of the very short edges we find only due to noise. These extensions to Canny's hysteresis step leave us with an edge detector, much more robust to noise, yet still not robust enough to simply apply it to noisy depth images. If the parameters are tuned to find most depth edges, there are still plenty of false positives in the noisy areas.
To cope with this problem, we fuse three different edge images into one. We start with an edge image from the depth data and set the thresholds very high to avoid any false positives. This leaves us with a set of definite depth edges (cf. Fig. 2 left), but the finer edges are missing.
Then we lower the thresholds to get all depth edges, but also some false positives in the noisy areas of the image (cf. Fig. 2 right). We cannot use these edges directly, but we will need them later on to detect shadow edges (cf. II-B) from the intensity image.
Finally, we apply the algorithm to the intensity image of the ToF camera as well. These images suffer from less noise and edge detection gives more accurate results. Even areas of the same material and surface properties, but different depths are usually distinguishable, since the main light source is mounted on the camera and thus, the brightness decreases with $1/r^2$ in depth. For scenes with little texture, the intensity edges coincide well with the actual depth edges. And thus, a combination of the confident depth edges (cf. Fig. 2 left) and the intensity edges seems to be a good solution already. But if we look closely at an intensity image, we see intensity discontinuities and thus edges, that are not caused by depth differences (cf. Fig. 3).

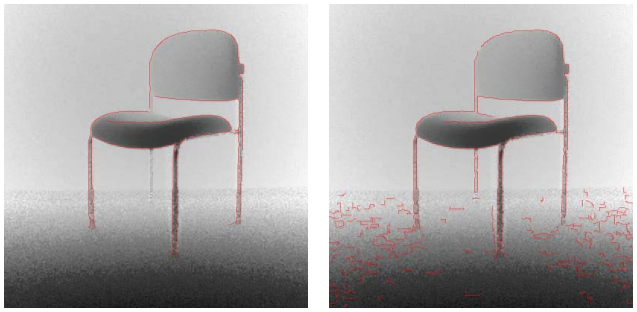


Figure 2. Edge pixels in depth data with high (left) and low (right) thresholds.

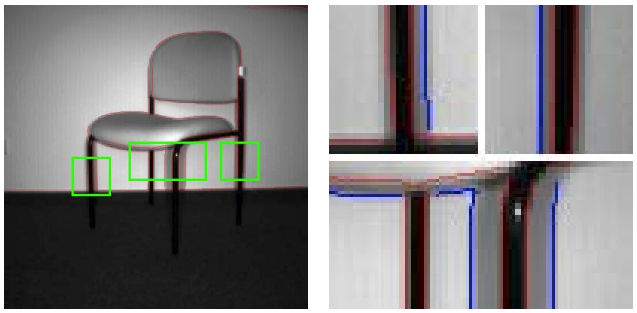


Figure 3. Edge pixels in intensity data and closeups with highlighted shadow edges in blue.

B. Removing Shadow Edges

Because the light sources of the ToF cameras cannot be mounted at the exact same position as the lens, parts of the light sources are always occluded close to depth discontinuities, causing shadows in the intensity image. These shadows are then detected as additional edges (cf. Fig. 3). In case of the PMD CamCube, the light sources are mounted left and right of the lens (cf. Fig. 4). Since the camera setup – in particular the position of the light sources – is known and the depth information d of the scene is given by the camera, the angle α' of a data point towards the light source can be calculated. All we need is the depth d_i of the pixel i and the horizontal distance to the optical axis x_i.

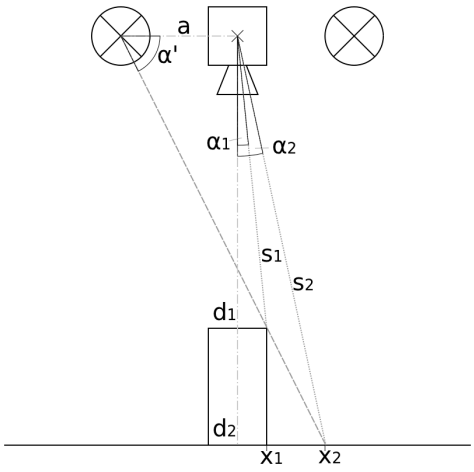


Figure 4. Geometry of camera and light paths.

The distance of camera and light source a is fixed and the horizontal angle α can be deduced from the opening angle of the camera and the column the examined pixel belongs to. The vertical angle or distance to the optical axis is not important because it is the same for both camera and light source. We retrieve α' from

$$\tan(\alpha_1') = \frac{d_1}{a + x_1} = \frac{d_2}{a + x_2} = \tan(\alpha_2') \qquad (1)$$

with $x_i = d_i \tan \alpha_i$ (cf. Fig. 4).
This means that due to the horizontal alignment of the light sources and the camera, wide shadows are cast by non-horizontal edges. So if we find a non-horizontal edge in the intensity image that is located at the same angle α' towards one light source as an edge in the depth image parallel to it, the probability is very high, that the former is just a shadow. It is possible to see both edges only because each is at a different depth d_1/d_2 and angle α_1/α_2 towards the lens. The shadow has to be right of the depth edge, if α' is calculated towards the left light source, and to the left of the depth edge, if the angle is calculated towards the right light source. Therefore these edges can be removed. In addition, this indicates, that the shadow casting depth edge is a real depth edge and not due to noise or artifacts and can be included in the edge image.
On very close inspection, we also see very small shadows at horizontal edges. These shadows occur because the light sources are LED clusters with a certain extend. At horizontal edges, parts of the light sources are occluded. But since the vertical expansion much smaller than the horizontal, this effect is only noticable at large depth differences of foreground and background, but it can be calculated in the same way and should be considered for an accurate edge detection. In a similar way the algorithm can be changed to work on other camera systems. Raskar et al. used a similar approach to detect depth edges with Multi-Flash Imaging (cf. [16]).

C. Removing Texture Edges

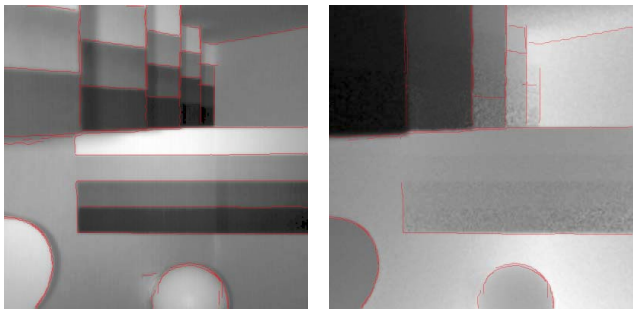


Figure 5. Edge pixels without shadows in intensity data before (left) and depth data after (right) removing texture edges.

Apart from shadows, there can also be texture edges, caused by different surface reflectivity without depth changes which have to be removed (cf. Fig. 5). We do this straight forward by calculating the difference in depth of two pixels on opposite sides of an edge pixel. The opposing pixels are selected along the normal vector $\vec{n}$. This can be done for a small neighborhood of N pixels, e.g. $N = 2$. The difference is then averaged for all pixels in a neighborhood along the connected edge E. If the depth difference

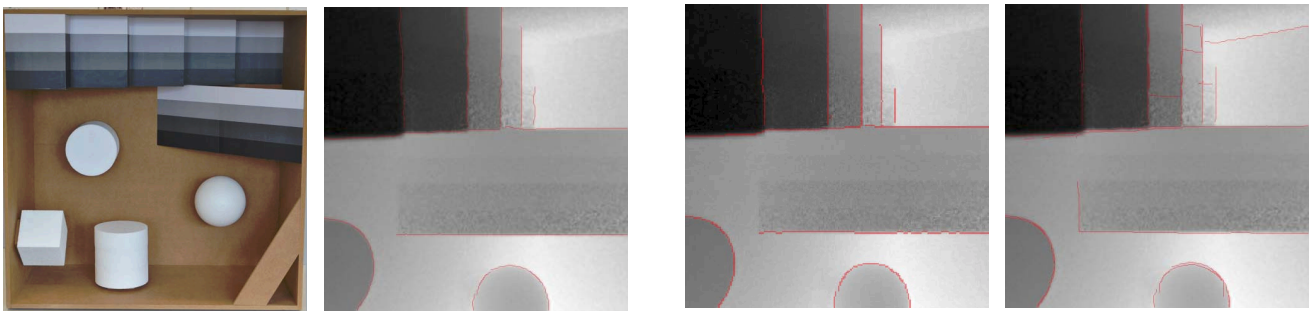
$$d_{diff} = \frac{1}{|E|N} \sum_{\vec{x} \in E} \sum_{j=1}^{N} |d(\vec{x} + j\vec{n}) - d(\vec{x} - j\vec{n})| \qquad (2)$$

is below a threshold t_d, the whole set can be considered a texture or color edge.

In case of a ridge edge pointing directly towards or from the camera, this method might fail. To avoid the removal of ridge edges we perform a similar calculation for the gradient magnitude of the depth data.
The large number of parameters is in practice not a big issue, since most of them can remain the same for different datasets. The main parameters are the standard thresholds for the Canny part, which mostly depend on the noise level.

III. Experiments

As test data we use one data-set of an office *chair* (cf. Fig. 6), two different views of a test scene with different geometric objects arranges in a *box* and a scene with a child's *toy*. The first data-set of the *box* (cf. Fig. 7) is recorded from a short distance with very little depth noise. The second data-set (cf. Fig. 8) of a slightly changed setup is recorded from a larger distance with a shorter integration time and contains significantly more noise in the depth-data. The *shapes* (cf. Fig. 9) are recorded with medium integration time and noise level.



(a) photograph of the *box* (b) depth data and our depth edges (c) depth and results of method from [11] (d) our final result

Figure 7. If we compare (c) and (d), the main improvement are the detected ridge edges at the cost of a few texture edges.

For the first set of depth edges and the intensity edges of each dataset, we choose parameters that rule out any noise edges, while for the second set of depth edges we low the thresholds far enough to detect all depth edges high enough to cast a shadow.
In the *chair* scene, most edges have a large depth difference

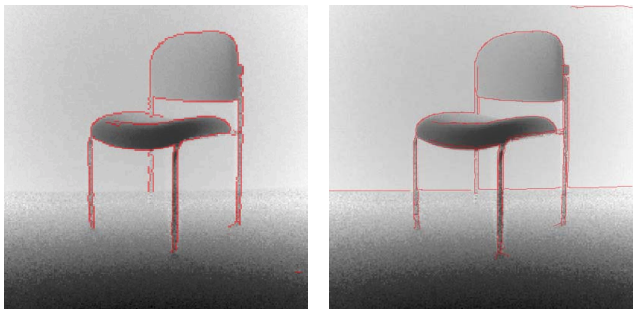
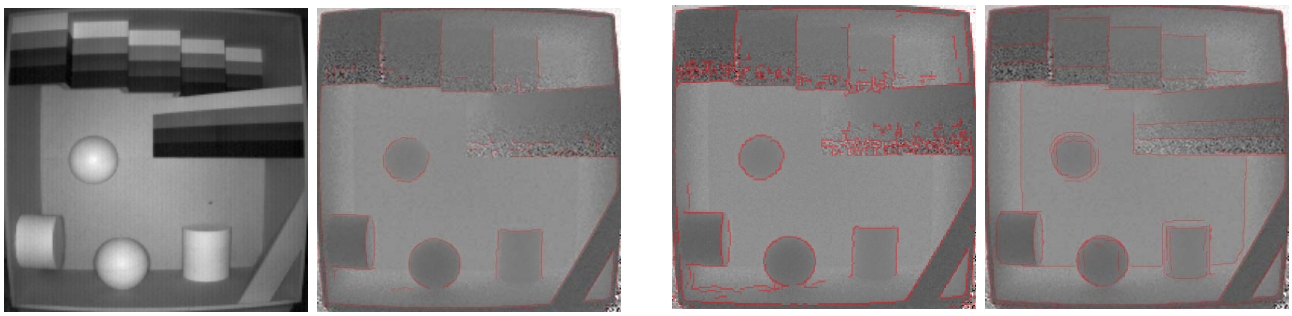


Figure 6. Edge pixels on depth data, calculated with method from [11] (left) and our final result (right)

towards the background. The advantage of using the intensity image shows at the rear leg of the chair, close to the wall and the carpet skirt along the wall, which is a very shallow depth edge. But the mere intensity data has its problems with the black legs of the chair in front of the dark carpet. We did not apply the texture edge-removal step here, since there are hardly any texture edges.
Applying only our proposed edge detection algorithm to the first data-set of the *box* (cf. Fig. 7b), we see that it produces satisfactory results and finds all of the jump edges. Compared to the gradient based approach (cf. Fig. 7c) introduced by Lejeune et al. in [11] there is no major difference, except for the higher resolution of our approach. If we apply our algorithm to the intensity image only, we find a lot more edges. Many of them are just texture edges, but some are ridge edges, such as the far corner of the *box*, or the socket of the sphere, which was completely covered up by noise in the depth image. After removing texture and shadow edges (cf. Fig 7d), we are left with the original depth edges, some additional ridge edges and a few left-over texture edges that were not properly removed because of the intensity related distance error [13]. The double edges are due to a mismatch of intensity and depth edges, because of noise and the nonlinear behavior of divided depth pixels.

The performance of both – the proposed and Lejeune's algorithm – on depth images changes a lot, when a different setup is used in which the box is recorded from a greater distance and with a shorter integration time (cf. Fig. 8). The lower intensity leaves us with much more noise in the depth data and makes edge detection very difficult. Lejeune's method produces a lot of false positive detections in the noisy area, resulting in a poor quality of the final edges. Moreover, it misses some of the obvious depth edges (cf. Fig. 8c). In contrast, the proposed approach is robust against the strong noise in the ToF data. After the first step of our algorithm, there are almost no false positive detections caused by noise (Fig. 8b). There are, however, false positive detections originating from shadows and textures in the intensity image.
In some parts of the image, we observe the occurrence of multiple edges, where only one edge should be found. Examples of this are in the lower part of the test image, where the polystyrene spheres and the cylinder are located. These multiple edges are caused by the surface properties of the polystyrene and the shallow viewing angles, reflecting less and less light towards the camera and creating a very strong intensity gradient inside the actual object boundaries. Some of these false or multiple edges are successfully removed in the shadow and texture edge removal steps of our approach. Some extra texture edges remain at areas where the intensity related distance error is not negligible or noise and scene geometry interfere with the texture removal.
For the fourth dataset (cf. Fig. 9) we see again, that our method (f) handles the noisy areas very well. Also for edges with a small depth difference, the combined approach works better. It detects e.g. the depth edges between the two



(a) ToF intensity (b) depth and depth edges of our algorithm (c) depth and results of method from [11] (d) our final result

Figure 8. The gain of our proposed approach in (c) compared to (b) can be seen clearly at the noisy regions as well as at the edges between the sides and the back of the *box*.

cuboids or the right edge of the triangular brick.
To sum up, the proposed approach is able to find a larger set of relevant edges than the method proposed by Lejeune, together with a much higher robustness against noise, only with the drawback that some false detections occur.

IV. Application – Denoising ToF Images

Edge detection algorithms often are employed to acquire input data for other image processing tasks. As an example we consider here the task of denoising, where edge information can be used to avoid smoothing across edges. We use the example of ToF denoising to further demonstrate the benefits of the proposed edge detection method. For denoising ToF data we proposed in [12] an approach based on adaptive total variation (TV) regularization, which requires information about the position and orientation of edges. Such information can be provided by the proposed approach. We show in the experiments below that using the edge information from the proposed approach instead of other edge detectors ([11], [12]) improves the denoising results. Before presenting these experiments, we briefly recall the approach presented in [12] for the readers convenience and describe some modifications to adapt this approach to the new edge data.

A. Adaptive Total Variation Denoising

In the following, adaptive TV denoising is described in a discrete setting. To this end, we consider a regular pixel grid of size $n \times m$, on which the ToF data $d^0 \in \mathbb{R}^{n\times m}$ are provided. We retrieve denoised depth data $d \in \mathbb{R}^{n\times m}$ by solving the convex optimization problem

$$\min_{d\in\mathbb{R}^{n\times m}} \frac{1}{2}\left(\sum_{i,j=1}^{n,m} w_{i,j}(d_{i,j}-d^0_{i,j})^2\right)+\mathcal{R}_1(d)+\mathcal{R}_2(d), \quad (3)$$

where $w_{i,j}$ are weights accounting for locally changing noise variance (cf. [12]), and $\mathcal{R}_1(d)$ and $\mathcal{R}_2(d)$ are regularization terms of first and second order, respectively, defined in detail below.

The first term provides an anisotropic total variation regularization:

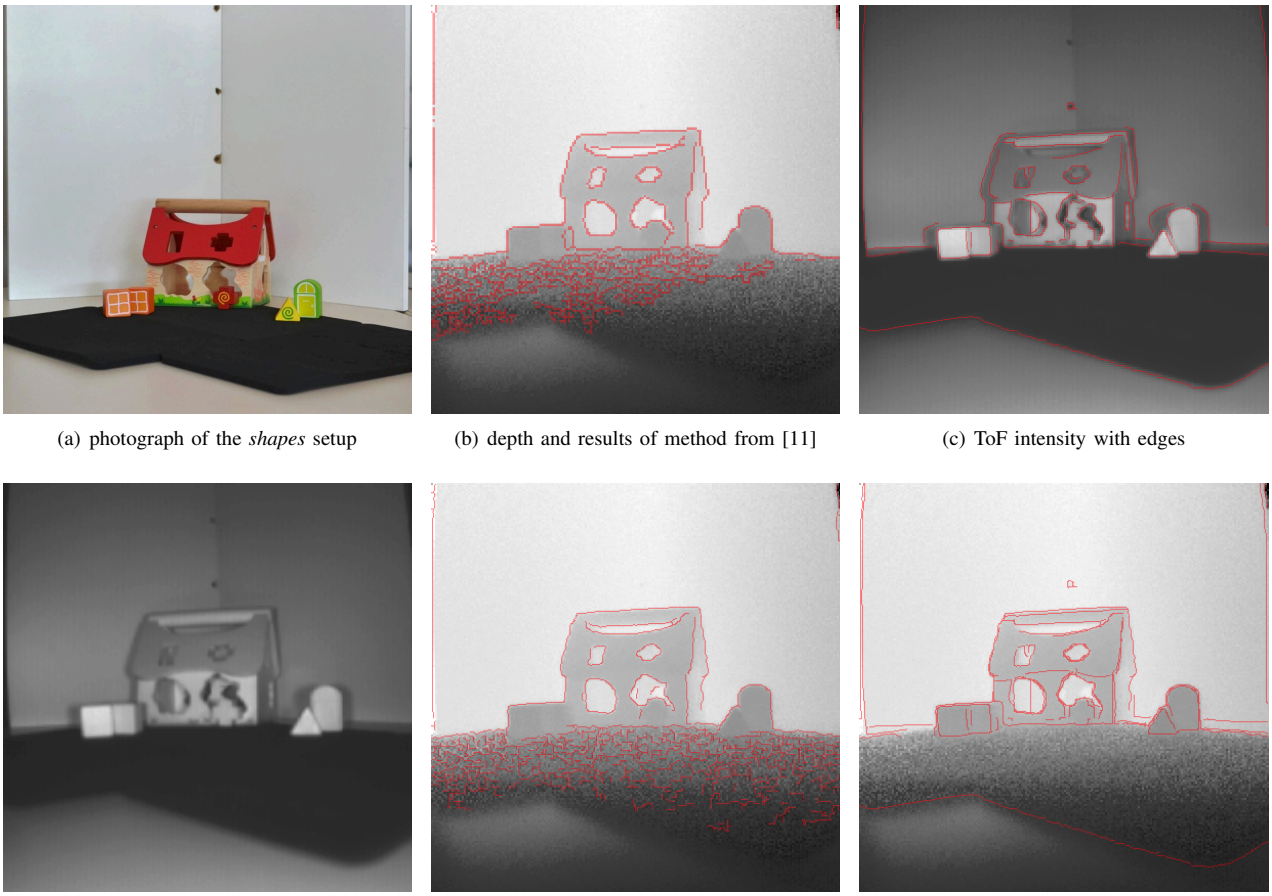
$$\mathcal{R}_1(d) := \sum_{i,j=1}^{n,m} \sqrt{(Ld)_{ij}^\top A_{i,j}(Ld)_{i,j}}, \quad (4)$$

where $(Ld) := (D_x d, D_y d)^\top$ is the discrete gradient of d based on discrete derivative operators D_x, D_y of first order (cf. [18]) and the matrices $A_{i,j} \in \mathbb{R}^{2\times 2}$ define the local anisotropy of the regularization. To determine this anisotropy, we require the location of depth edges encoded by a variable $s_{i,j} \in [0,1]$, i.e. $s_{i,j} \approx 1$ if an edge is located at pixel (i,j) and $s_{i,j} \approx 0$ otherwise, as well as normalized edge normals $v_{i,j} \in \mathbb{R}^2$. These data are provided by the proposed edge detection method. We use $A_{i,j} := \tilde{\alpha}_{i,j}^2 v_{i,j} v_{i,j}^\top + \beta^2(\mathrm{Id} - v_{i,j}v_{i,j}^\top)$ in order to penalize gradients in normal direction by a factor $\tilde{\alpha}_{i,j}$ and in orthogonal direction by a factor β. To achieve an anisotropic smoothing ($\tilde{\alpha}_{i,j} \ll \beta$, i.e. a smoothing mainly parallel to edges) only at edges, and a homogeneous smoothing (i.e. $A_{i,j} = \beta^2\mathrm{Id}$) elsewhere, we set $\tilde{\alpha}_{i,j} := s_{i,j}\alpha + (1-s_{i,j})\beta$ with fixed smoothing parameters $0 < \alpha \ll \beta$.

The second regularization term is given as

$$\mathcal{R}_2(d) := \gamma \sum_{i,j=1}^{n,m} |(Hd)_{i,j}|_2 \quad (5)$$

with smoothing parameter $\gamma \geq 0$, where $Hd := (D_{xx}d, D_{xy}d, D_{yx}d, D_{yy}d)^\top$ is the discrete Hessian of d based on discrete derivative operators $D_{xy}, D_{xy}, D_{yx}, D_{yy}$ of second order (cf. [18]). $\mathcal{R}_2(d)$ serves the purpose of smoothing surfaces in the depth map and, in particular, preventing stair-casing effects which standard TV regularization [17] is known for. However, we do not want such a smoothing over edges. To this end, we slightly modify the approach in [12] and take into account the double resolution edge indicator set provided by the proposed approach, i.e. in Hd differences of d between adjacent pixels are artificially



(a) photograph of the *shapes* setup (b) depth and results of method from [11] (c) ToF intensity with edges

(d) ToF intensity (e) our depth edges (f) our final result

Figure 9. Our method (f) performs better than (b) in noisy areas and at small depth steps, like the wooden bricks left and right of the house.

set to zero, if our approach finds an edge in-between these pixels.

We solve (3) numerically by applying a primal-dual algorithm [3].

B. Denoising Results

We apply the above denoising method to the ToF depth maps shown in Figs. 1 and 7 (see also top row of Fig. 10). We compare three different variants of adaptive TV denoising, which are based on different algorithms for edge detection. Firstly, we consider the our previous approach in [12] (second row), where the edges are also detected by jointly applying a structure tensor to intensity and depth data, but without any post-processing. Secondly, we use the denoising method described above in combination with Lejeune's edge detection (third row). Finally, we combine the above denoising method with the edge information from the proposed approach (bottom row).

The method from [12] handles the noisy areas quite well, but also blurs the edges (cf. stairs in first column/house in second column). The results of the adaptive TV denoising with the edges from Lejeune's algorithm generate sharp depth edges, but the additional edges in the noisy areas of the images prevent a proper smoothing of these areas. Instead of a smooth slope in the depth map, several plateaus of constant depth are generated. Adaptive TV denoising only in combination with the proposed edge detection algorithm is able to completely eliminate the strong noise while at the same time both preserving sharp object boundaries and smooth slopes of planar surfaces.

We remark that in the results of our edge detection algorithm some additional texture edges from the intensity image may still remain, for example in the *box* data set at the stairs or the ramp, or in the *shapes* data set between the two cubic objects (see close-up in third column). These additional edges, depending whether they coincide with real depth edges (cubic objects) or not (stairs/ramp), have a positive or negative effect on the denoising result. In the *box* data set the additional texture edges prevent the denoising algorithm from smoothing the complete surfaces, while in

the *shapes* data set they make some structures like the two cubic objects distinguishable in the depth image, where the other two algorithms fail to separate the different depths. We conclude that, when using additional edge information from the intensity data, it has to be verified that additional edges indeed coincide with real depth edges. As mentioned before, deriving additional criteria for that is part of our future work.

V. Conclusion & Outlook

We have introduced a new algorithm for edge detection in Time-of-Flight depth images and evaluated it by comparing it to another recent approach, tailored for ToF depth images. Our proposed algorithm uses both the depth and the intensity image to find depth edges, allowing it to outperform other algorithms which exclusively rely on the depth or the intensity image. In order to achieve highly accurate results, shadows in the intensity images are detected and their edges removed by the new approach. With this technique we created an algorithm that overcomes the essential problems of edge detection in ToF images.

As an exemplary application, which requires edge detection as a preprocessing step, we have considered an adaptive denoising method. We have shown that the performance of this denoising method significantly increases, when our edge detection results are used instead of other state-of-the-art algorithms.

In our approach, we assume an illumination model with point-light sources, which is a sufficiently accurate approximation for most cases. However, future models of shadow detection could include spatially expanded light sources and therefore also work for cameras with concentrically aligned leds, which also suffer from shadows, but less distinctly. The partly occluded light source does not only change the intensity by casting shadows, but also the average time of flight for non-center pixels, which is why it should definitely be considered for high accuracy measurements.

For future work, we will focus on improved accuracy of the approach, such as exact matching of depth and intensity edges, scene geometry and a distinction between step and ridge edges, which is an important issue for denoising or optical flow.

Acknowledgements

The work presented in this article has been co-financed by the Intel Visual Computing Institute. The content is under sole responsibility of the authors.

References

[1] P. Boulanger, F. Blais, and P. Cohen. Detection of depth and orientation discontinuities in range images using mathematical morphology. In *Pattern Recognition, Proceedings, 10th International Conference on*, 1990.

[2] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1986.

[3] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *JMIV*, 2011.

[4] S. Coleman, B. Scotney, and S. Suganthan. Edge detecting for range data using laplacian operators. *IEEE Trans. Image Process.*, 2010.

[5] W. Förstner. A framework for low-level feature extraction. In *ECCV*, LNCS, 1994.

[6] S. B. Gokturk, H. Yalcin, and C. Bamji. A time-of-flight depth sensor-system description, issues and solutions. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*. IEEE, 2004.

[7] G. Harris and M. Stevens. A combined corner and edge detector. *Proc. of the 4th Alvey Vision Conference*, 1988.

[8] X. Jiang and H. Bunke. Edge detection in range images based on scan line approximation. *Computer Vision and Image Understanding*, 1999.

[9] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07. ACM, 2007.

[10] U. Köthe. Edge and junction detection with an improved structure tensor. In *Proc. of the 25th DAGM Symposium on Pattern Recognition*, LNCS. DAGM, 2003.

[11] A. Lejeune, S. Piérard, M. Van Droogenbroeck, and J. Verly. A new jump edge detection method for 3D cameras. In *International Conference on 3D Imaging (IC3D)*, 2011.

[12] F. Lenzen, H. Schäfer, and C. S. Garbe. Denoising time-of-flight data with adaptive total variation. In *Advances in Visual Computing*, LNCS. Springer Berlin / Heidelberg, 2011.

[13] M. Lindner and A. Kolb. Calibration of the intensity-related distance error of the pmd tof-camera. In *Optics East*. International Society for Optics and Photonics, 2007.

[14] J. Park, H. Kim, Y.-W. Tai, M. S. Brown, and I. Kweon. High quality depth map upsampling for 3d-tof cameras. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011.

[15] B. Parvin and G. Medioni. Adaptive multiscale feature extraction from range data. *Computer Vision, Graphics, and Image Processing*, 1989.

[16] R. Raskar, K.-H. Tan, R. Feris, J. Yu, and M. Turk. Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. In *TOG*. ACM, 2004.

[17] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 1992.

[18] S. Setzer, G. Steidl, and T. Teuber. Infimal convolution regularizations with discrete l1-type functionals. *Comm. Math. Sci.*, 2011.

[19] C. Ye and G.-P. Hegde. Robust edge extraction for swissranger sr-3000 range images. In *ICRA*. IEEE, 2009.

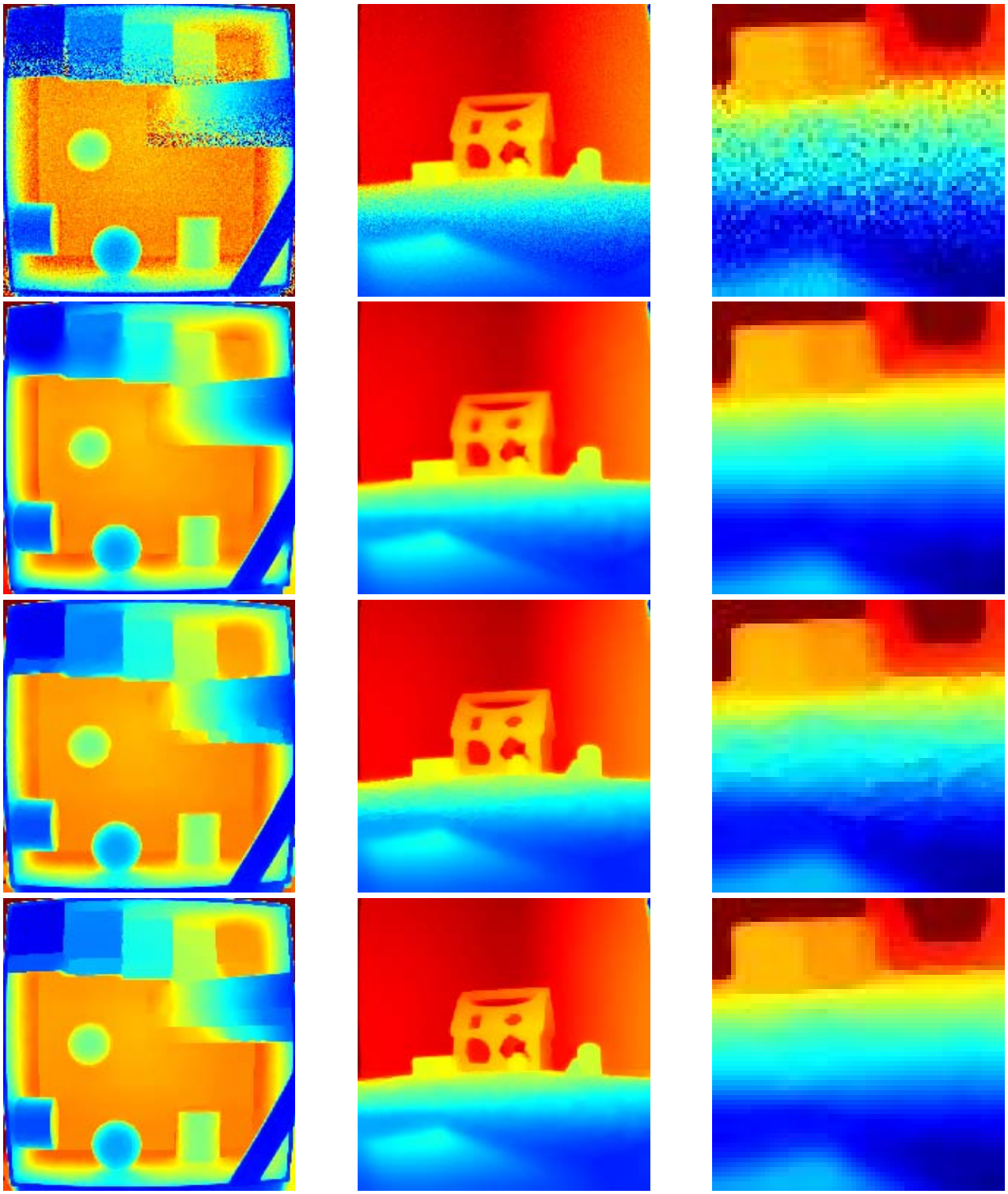


Figure 10. Denoising depth data. Original depth image (top row), denoising as described in [12] (second row), anisotropic TV with edges calculated with algorithm described by Lejeune et al. [11] (third row) and with edges from the proposed approach (bottom row). The third column is a close-up of the second column with enhanced contrast. We observe better quality of the denoised depth map (sharpness of edges, regularity of faces) using the edge information of the proposed algorithm.