Truncated Signed Distance Function: Experiments on Voxel Size

Diana Werner(✉), Ayoub Al-Hamadi, and Philipp Werner

University of Magdeburg, Magdeburg, Germany
{Diana.Werner,Ayoub.Al-Hamadi}@ovgu.de

Abstract. Real-time 3D reconstruction is a hot topic in current research. Several popular approaches are based on the truncated signed distance function (TSDF), a volumetric scene representation that allows for integration of multiple depth images taken from different viewpoints. Aiming at a deeper understanding of TSDF we discuss its parameters, conduct experiments on the influence of voxel size on reconstruction accuracy and derive practical recommendations.

Keywords: TSDF · KinectFusion · 3D reconstruction

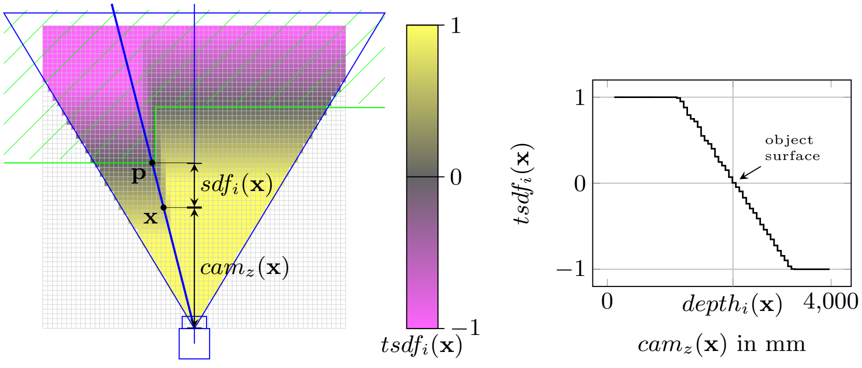
1 Introduction

Accurate 3D reconstruction in real-time has a lot of applications in entertainment, virtual reality, augmented reality and robotics. The introduction of Microsoft's low-cost RGB-D camera Kinect [5] has made 3D sensing available to everyone. So it also has boosted research and commercial activities in 3D reconstruction and its applications. Several popular and widely used approaches such as KinectFusion [4,6], Kintinious [10,11], the open source implementation of KinectFusion in the point cloud library (PCL) [12] or KinFu Large Scale [1] are based on the truncated signed distance function (TSDF).

TSDF is a volumetric representation of a scene for integrating depth images that has several benefits, e. g. time and space efficiency, representation of uncertainty or incremental updating [2]. It further is well-suited for data-parallel algorithms, i. e. for implementation on GPUs. The attained speed-up facilitates real time processing at high frame rate.

There has been some investigations on hole filling to generate more natural looking reconstructions, which is partly done automatic using the TSDF method [2], and can be done in an energy conservation way [7]. However, to our knowledge there has been no closer look at scenarios with multiple objects and other important questions on object size and resolution. For instance: Up to which point is a true hole reconstructed as a hole? In comparison with camera position and direction: Is the reconstruction influenced by distance or angle? Are there problems at the very left or right border of an object seen from an specific direction? In which way is the reconstruction influenced by the voxel size used in the world grid?

A. Campilho and M. Kamel (Eds.): ICIAR 2014, Part II, LNCS 8815, pp. 357–364, 2014.
DOI: 10.1007/978-3-319-11755-3_40



(a)

(b)

Fig. 1. 2D TSDF example. (a) Solid object (green), camera with field of view, optical axis and ray (blue), and TSDF grid (unseen voxels are white, for others see color bar). The signed distance value of voxel $\mathbf{x}$ is determined by the depth of the corresponding surface point $\mathbf{p}$ and the voxel's camera distance $cam_z(\mathbf{x})$. b) 1D TSDF sampled along the ray through $\mathbf{p}$ with $t = 1000$ mm. Object surface is at zero crossing.

In this paper we analyze the influence of distance, object size and angle to camera viewing direction. We will have a look at this with combination of several world grid voxel sizes.

This paper is structured as follows. Section 2 describes the TSDF in detail. It is shown how this function is generated from a depth map after sensing the environment and how one can compute the 3D reconstruction given the TSDF. Section 3 describes parameters and algorithmic options to consider for improving results. In section 4 we describe our experiments and discuss our results. Section 5 concludes this paper.

2 TSDF

The signed distance function (SDF) was proposed to reconstruct a 3D model from multiple range images [2]. A d-dimensional environment is represented in a d-dimensional grid of equally sized voxels. The position of a voxel $\mathbf{x}$ is defined by its center. For each voxel there are two relevant values. First, $sdf_i(\mathbf{x})$ which is the signed distance in between voxel center and nearest object surface in direction of current measurement. In front of an object (in free space) the values are defined to be positive. Behind the surface (inside the object) distances are negative. Second, there is a weight $w_i(\mathbf{x})$ for each voxel to assess uncertainty of the corresponding $sdf_i(\mathbf{x})$. The subscript i denotes the i'th observation. Fig. 1a and the following equation define $sdf_i(\mathbf{x})$ precisely.

$$sdf_i(\mathbf{x}) = depth_i(pic(\mathbf{x})) - cam_z(\mathbf{x}) \tag{1}$$

$pic(\mathbf{x})$ is the projection of the voxel center $\mathbf{x}$ onto the depth image. So $depth_i(pic(\mathbf{x}))$ is the measured depth in between the camera and the nearest object surface point $\mathbf{p}$ on the viewing ray crossing $\mathbf{x}$. Accordingly, $cam_z(\mathbf{x})$ is the distance in between the voxel and the camera along the optical axis. Consequently, $sdf_i(\mathbf{x})$ is a distance along the optical axis as well.

In [4,6] the SDF has been truncated at $\pm t$. This is beneficial, because large distances are not relevant for surface reconstruction and a restriction of the value range can be utilized to memory footprint. The truncated variant of $sdf_i(\mathbf{x})$ is denoted by $tsdf_i(\mathbf{x})$.

$$tsdf_i(\mathbf{x}) = \max(-1, \min(1, \frac{sdf_i(\mathbf{x})}{t})) \tag{2}$$

In Fig. 1a $tsdf_i(\mathbf{x})$ of the voxel grid is encoded by color. Fig. 1b shows the TSDF sampled along a viewing ray.

As mentioned above, multiple observations can be combined in one TSDF to integrate information from different viewpoints to improve accuracy or to add missing patches of the surface. This is done by weighted summation, usually through iterative updates of the TSDF. $TSDF_i(\mathbf{x})$ denotes the integration of all observations $tsdf_j(\mathbf{x})$ with $1 \leq j \leq i$. $W_i(\mathbf{x})$ assesses the uncertainty of $TSDF_i(\mathbf{x})$. A new observation is integrated by applying the following update step for all voxels $\mathbf{x}$ in the grid. The grid is initialized with $TSDF_0(\mathbf{x}) = 0$ and $W_0(\mathbf{x}) = 0$.

$$TSDF_i(\mathbf{x}) = \frac{W_{i-1}(\mathbf{x})TSDF_{i-1}(\mathbf{x}) + w_i(\mathbf{x})tsdf_i(\mathbf{x})}{W_{i-1}(\mathbf{x}) + w_i(\mathbf{x})} \tag{3}$$

$$W_i(\mathbf{x}) = W_{i-1}(\mathbf{x}) + w_i(\mathbf{x}) \tag{4}$$

Most approaches set the uncertainty weight to $w_i(\mathbf{x}) = 1$ for all updated voxels and to $w_i(\mathbf{x}) = 0$ for all voxels outside the camera's field of view [4,6,9,12]. This simply averages the measured TSDF observations over time.

For surface reconstruction one can think of the TSDF like a level set. To find the object surface you look for the zero level. This is usually done by ray casting from a given camera viewpoint. For each considered ray the TSDF is read step by step until there is a switch in sign. The information of surrounding TSDF values is then interpolated to estimate the refined point of zero crossing along the ray. This point is returned as an object surface point.

3 Parameters and Algorithmic Options

The TSDF representation requires to select several parameters.

Grid volume size determines the dimensions of the TSDF grid, i. e. the maximum dimensions of the scene to reconstruct in a physical unit like mm. In practice it is bounded by the available Random Access Memory on the GPU. However, several previous works suggested to overcome this limitation by shifting the TSDF grid when the camera's field of view leaves the grid [1,8,10,11]. The grid dimensions increase with voxel size in constant memory footprint, however at cost of reconstruction accuracy.

Voxel size v is a crucial parameter as it influences memory requirements and surface reconstruction accuracy. If dimensions of a 3D grid are fixed, doubling the voxel size means to reduce the number of voxels to one-eighths. This is associated with the same reduction in memory footprint. Further, it reduces computational cost for updating the TSDF and for ray tracing. The other way around, an increased voxel size facilitates to increase the scene volume without needing more memory or increasing computational cost. However, an increase in voxel size comes along with a decrease in the level of representable details resp. with lowered reconstruction accuracy. So it is worth thinking about the optimal voxel size for a particular application. In Sect. 4 we conduct experiments to assess the influence of this parameter on the accuracy.

Distance representation and truncation distance t i. e. the coding of distance values $TSDF_i(\mathbf{x})$ is crucial for the reconstruction accuracy. Intuitively, there should be as many quantization steps as possible to minimize information loss caused by rounding and maintain the accuracy of $TSDF_i(\mathbf{x})$. Especially, this is important near zero level, as those values of $TSDF_i(\mathbf{x})$ are used for surface estimation. So floating point is appropriate. In terms of memory footprint it is beneficial use two byte integer per voxel as in the implementation of PCL [12]. However, here the selection of t influences reconstruction accuracy. Two byte integer has 65,536 quantization steps to represent a distance, i. e. integer values between $-32,768$ and $32,767$. With a fixed point number coding and a given truncation distance t, signed distances in range $\pm t$ are mapped to $\pm 32,767$. So signed distances are quantized in steps of $\frac{t}{32,767}$, i. e. the quantization error is proportional to t and a lower t should be better. E. g. with $t = 1,000$ mm the coding will round the each distance to multiples of 0.03 mm whereas with $t = 10$ mm it will round to multiples of 0.0003 mm. On the other hand, t should be larger than length of voxel diagonal $\sqrt{d} \cdot v$ and the level of noise. A detailed analysis of this parameter is out of scope, but will be addressed in future work.

Next to the parameters there are some algorithmic options for variation.

TSDF update i. e. the integration of multiple observations in one TSDF can be accomplished in different ways. Above, we introduced the classical equally weighted sum update. Recently, [3] proposed to select $w_i(\mathbf{x})$ individually for each voxel based on the uncertainty of the measurement. The authors model $w_i(\mathbf{x})$ dependent on the corresponding depth value, as the depth estimation provided by the used Kinect sensor is more accurate in close range. They further propose two modified update methods and evaluate their benefit for the model reconstruction accuracy. Their variants of the TSDF update, which consider characteristics of the sensing hardware, outperform the classical update step in the presented experiments.

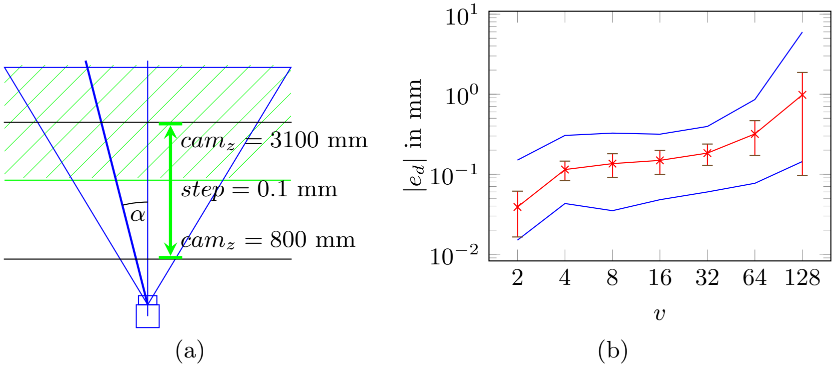


Fig. 2. Depth experiment. (a) Planar object and its variation. (b) Absolute depth error for different voxel sizes: mean with standard deviation (red), maximum and minimum (blue), $t = 255\,\text{mm}$.

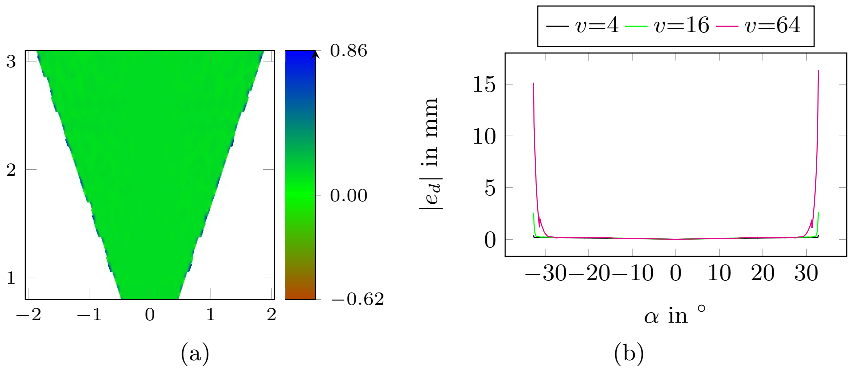
Surface reconstruction is done via ray tracing. You start at a point on the ray as near as possible to camera but in TSDF volume. From this you are going with a **individually chosen step size** to the next ray point and so on. You have to decide whether to look at the value $tsdf_i(x)$ for the voxel x related to a ray point or to interpolate a TSDF value from surrounding voxels. You also have to decide for an **interpolation** method. In PCL [12] and in other works like [4] the trilinear interpolation is used. After a zero crossing is detected between two ray points from positiv to negative you are able to compute a surface point via the chosen interpolation. You also can decide whether to stop the ray tracing after computing one surface point to reconstruct only surface points seen from camera or not.

4 Experiments and Discussion

In this section we conduct several experiments aiming at a deeper understanding of TSDF, i. e. at effects of spacial discretization in a grid on the reconstruction accuracy. To focus on these effects, we created synthetic depth maps which do not suffer from noise or other measurement errors. Another advantage of synthetic data is the availability of perfect ground truth. On purpose of simple illustration experiments are conducted with a 2D TSDF grid, i. e. we have a 1D depth map and a 2D surface. All the experiments demonstrate the result after a single depth measurement. Integration of multiple depth maps is out of scope and will be addressed in future work. We decompose the reconstruction accuracy in two components and investigate each in a dedicated experiment.

4.1 Depth Error

In this experiment we calculate the depth error e_d for each pixel of a synthetic depth map and the according depth reconstructed from the TSDF. The synthetic



(a)

(b)

Fig. 3. Angular depth error. (a) Map of e_d across TSDF volume for voxel size v of 64 mm. (b) Mean error along rays for different voxel sizes v.

depth maps contain a planar object surface which is perpendicular to the optical axis (see Fig. 2a). We placed the object 800 mm in front of camera position and move it from this position in steps of 0.1 mm till a distance of 3100 mm. The camera is placed as in Fig. 1a with optical axis passing through the middle of the TSDF grid and aligned with one of its axes. The size of the world grid is fixed to 4096 mm width and height in all experiments.

Fig. 2b shows the mean, standard deviation, minimum and maximum of the absolute depth error across tested field of view for several voxel sizes v. It is apparent that the absolute depth error increases with voxel size. However, the mean error increases slower than voxel size. Whereas the mean error is 0.04 mm for $v = 2$ mm (1.9 %), it is 0.97 mm for $v = 128$ mm (0.7 %). There is a similar effect for the maximum and minimum error.

Further, looking at the spacial distribution of the error one can observe that the most severe errors occur on the border of the field of view (see Fig. 3a) whereas the object distance seems to have no influence on reconstruction accuracy. To investigate the influence of the angle in between grid axis and ray we calculate the mean of absolute error along each viewing ray. The results are given in Fig. 3b. Here it is apparent that the error is at least two orders of magnitude larger on the borders of the field of view, and even more the higher v gets. The high error at the borders are artefacts caused by the trilinear interpolation. The PCL implementation of KinectFusion that we used for our experiments [12] does not pay attention to the fact that some TSDF voxels $\mathbf{x}$ involved in the interpolation may be unseen. For these voxels $tsdf_i(x) = 0$ from initialization. There has been no sensing for these voxels and therefore you can not expect to have a surface there, which would be true for seen voxles x with a TSDF value of 0. With increasing voxel size this error gets apparent, as more reconstructed 3D points are affected by this problem because the influence of the interpolation is one voxel size.

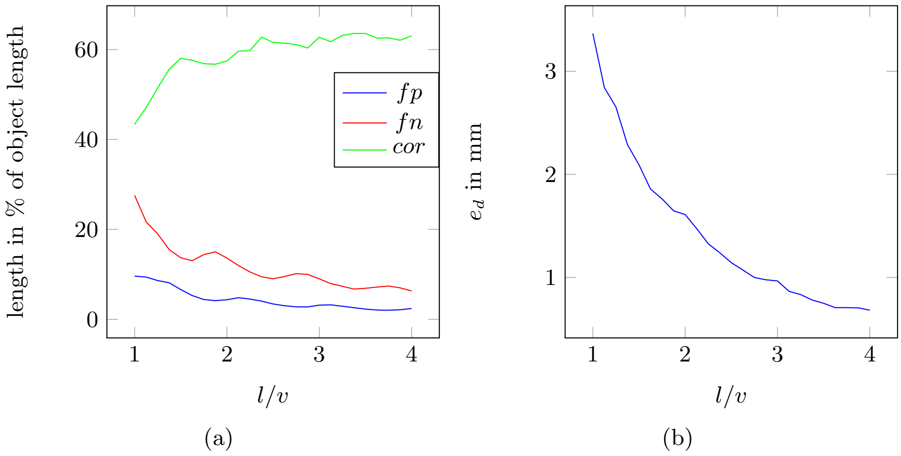


Fig. 4. Lateral depth error. (a) Average length of wrongly reconstructed (fp), wrongly not reconstructed (fn) and correctly reconstructed (cor) object in percent of true object length. (b) Average depth error in connection to ratio of true object length l and voxel soze v. Voxel size v is 64 mm.

4.2 Lateral Error

With this experiment we investigate how the ratio of voxel size v and the length l of an planar object, located perpendicular to the optical camera axis, influence the reconstruction accuracy. The objects were moved in a camera distance from 1280 mm to 1536 mm, wich are 4 voxels with 64 mm size. We chosed step sizes 8 mm in vetical and horizontal direction. The object bounds laid inside the viewing area with an distance of 128 mm from border. For all objects of same length generated for this experiment we calculated mean values and looked at depth error d_e and at the length fp, fn and cor which are the length of the reconstructed object wich are wrongly reconstructed, wrongly not reconstructed and correctly reconstructed in % of the true object length.

In Fig. 4 you can assert that all of these length together with e_d are clearly influenced by the ratio of l and v. It is also obvious that the objects are reconstructed too small in average.

5 Conclusion

In this paper we gave a detailed look at TSDF and the parametric and algorithmic options. We showed that for PCL's implementation depth errors are in same magnitude for voxel sizes 4 to 64 mm. The errors are 2 magnitudes larger at the border of viewing field due to interpolation effects. For lateral errors there is a strong relation between ratio of object length and voxel size and the reconstruction accuracy whith the object being too small in average. The negative impact of increase in voxel size is lower for depth error than for lateral error.

Acknowledgments. This work was supported by Transregional Collaborative Research Centre SFB/TRR 62 ("Companion-Technology for Cognitive Technical Systems") funded by the German Research Foundation (DFG).

References

1. Bondarev, E., Heredia, F., Favier, R., Ma, L., de With, P.H.: On photo-realistic 3D reconstruction of large-scale and arbitrary-shaped environments. In: 2013 IEEE Consumer Communications and Networking Conference (CCNC), pp. 621–624. IEEE (2013)
2. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, pp. 303–312. ACM, New York (1996)
3. Hemmat, H.J., Bondarev, E., de With, P.H.N.: Exploring distance-aware weighting strategies for accurate reconstruction of voxel-based 3D synthetic models. In: Gurrin, C., Hopfgartner, F., Hurst, W., Johansen, H., Lee, H., O'Connor, N. (eds.) MMM 2014, Part I. LNCS, vol. 8325, pp. 412–423. Springer, Heidelberg (2014)
4. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A.: KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In: Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, pp. 559–568. ACM (2011)
5. Microsoft: Kinect (2014), http://www.xbox.com/en-us/kinect/
6. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: real-time dense surface mapping and tracking. In: Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2011, pp. 127–136. IEEE Computer Society, Washington, DC (2011)
7. Paulsen, R.R., Bærentzen, J.A., Larsen, R.: Regularisation of 3D signed distance fields. In: Salberg, A.-B., Hardeberg, J.Y., Jenssen, R. (eds.) SCIA 2009. LNCS, vol. 5575, pp. 513–519. Springer, Heidelberg (2009)
8. Roth, H., Vona, M.: Moving volume KinectFusion. In: Proceedings of the British Machine Vision Conference, pp. 1–11. BMVA Press (2012)
9. Whelan, T., Johannsson, H., Kaess, M., Leonard, J.J., McDonald, J.: Robust tracking for real-time dense RGB-D mapping with kintinuous. Technical Report 031. MIT-CSAIL (2012), http://hdl.handle.net/1721.1/73167
10. Whelan, T., Johannsson, H., Kaess, M., Leonard, J.J., McDonald, J.: Robust real-time visual odometry for dense RGB-D mapping. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 5724–5731. IEEE (2013)
11. Whelan, T., Kaess, M., Fallon, M., Johannsson, H., Leonard, J., McDonald, J.: Kintinuous: Spatially extended KinectFusion. Technical Report 020. MIT-CSAIL (2012), http://hdl.handle.net/1721.1/71756
12. Willow Garage and other contributors: Open source implementation of KinectFusion in PCL 1.7.1 (2014), http://www.pointclouds.org/downloads/