Journal of Applied Research and Technology

www.jart.icat.unam.mx

*Journal of Applied Research and Technology* 19 (2021) 644-652

Original

# Dense monocular simultaneous localization and mapping by direct surfel optimization

E. Trabes[a,b,c]* • L. Avila[b,c] • J. D. Gazzano[a] • C. F. Sosa Paez[a]

[a]*Universidad Nacional de San Luis, Facultad de Ciencias Físico Matemáticas y Naturales, Departamento de Electrónica, San Luis, Argentina*
[b] *Universidad Nacional de San Luis, Facultad de Ingeniería y Ciencias Agropecuarias, Departamento de Ingeniería, Laboratorio de Mecatrónica, San Luis, Argentina*
[c]*Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC), CONICET-UNSL, San Luis, Argentina*

Received 04 27 2020; accepted 08 03 2021
Available 12 31 2021

**Abstract:** This work presents a novel approach for monocular dense simultaneous localization and mapping. The surface to be estimated is represented as a piecewise planar surface, defined as a group of surfels each having as parameters the position and normal. These parameters are directly estimated from the raw camera pixels measurements using a Gauss-Newton iterative process. The representation of the surface as a group of surfels has many advantages. First, it allows recovering robust and accurate pixel depths, without the need to use a computationally demanding depth regularization schema. This has the further advantage of avoiding the use of a physically unlikely surface smoothness prior. What is more, new surfels can be correctly initialized from the information present in nearby surfels, avoiding also the need to use an expensive initialization routine commonly needed in Gauss-Newton methods. The method was written in the GLSL shading language, allowing the use of GPU devices and achieving real-time processing. The method was tested on benchmark datasets, showing both its depth and normal estimation capacity, and its quality to recover the original scene. Results presented in this work showcase the usefulness of the more physically grounded piecewise planar scene depth prior, instead of the more commonly pixel depth independence and smoothness prior.

*Keywords:* Depth estimation, visual odometry, SLAM

*Corresponding author.
*E-mail address:* emanueltrabes@gmail.com(E. Trabes).
Peer Review under the responsibility of Universidad Nacional Autónoma de México.

## 1. Introduction

One of the main abilities that unmanned vehicles should have is the capacity to estimate their relative position within an exploration environment. What is more, complex missions like resources detection and obstacle avoidance, require also that the vehicle be able to build a detailed map of its surroundings. A manner to acquire such knowledge is through the use of mapping methods, where a model of the nearby terrain is recovered from the measurements provided by onboard sensors. It's well known in the literature that the problem of positioning and mapping are interconnected, and both have to be solved simultaneously (Grisetti et al., 2010). Systems that are capable of position estimation and mapping are known in the literature as Simultaneous localization and mapping (SLAM) (Cadena et al., 2016). There exist different kinds of sensors that can be utilized to accomplish the SLAM task (Lemus et al., 2014). A very interesting choice is the monocular camera. These kinds of sensors have, with the current state of technology, very large resolutions and also very high refresh rates.

There are many SLAM systems currently present in the literature that successfully use a monocular camera as the only sensor for resolving the SLAM problem. One example is LSD-SLAM (Engel et al., 2014), which uses an epipolar search approach together with a Kalman filter for estimating the depth of some of the pixels of a frame taken from a position of reference. The pixels that, due to lack of texture in the frame, cannot be estimated accurately are detected and discarded, and their depths are not estimated. With the depth of the pixels and the position of the reference frame, a local map is constructed. For the frames taken in nearby positions, the information obtained from their relative position within the local map is used to improve the reference frame pixels depth estimation. Another example is DSO (Engel et al., 2018), which takes an even more restrictive approach with the selection of pixels to be used for depth estimation. Only a very small percentage of the pixels in the frame are used for depth estimation. The resulting map is a very sparse one, but the results in localizations are outstanding. Other works like ORB-SLAM (Mur-Artal et al, 2015) use a pre-processing step on the captured frames, to identify the areas where robust depth estimation is more plausible. This pre-processing utilizes a salience detector to filter the zones of the frame that are not well posed for depth and pose estimation. All of these methods do not try to recover a detailed map of the surrounding terrain, focusing their attention on correct position estimation. Therefore, the usefulness of these methods in missions that require detailed knowledge of the surrounding terrain is questionable.

DTAM (Newcombe et al., 2011) tries to estimate depth in every pixel. A brute force plane sweeping approach (Hosni et al., 2013) in conjunction with a regularizer is used to accomplish this task. The system is still based on a pixel-wise epipolar search. And so, the estimation in areas in the frame where there is no texture is heavily regularized with the depth of nearby pixels, in order to recover somewhat correct depth estimations. REMODE (Pizzoli et al., 2014) uses an epipolar depth search together with a probabilistic measurement merging, again with a regularization at the end of the process. Zones in the frame where there is no texture must be, again, heavily regularized. The work of (Zienkiewicz et al., 2016) tries to estimate a mesh from the frames taken from a monocular camera. The system utilizes first a plane sweeping method similar to (Newcombe et al., 2011) to estimate the depth, and then this information is used to optimize every vertex of the mesh. The use of a pixel-wise estimation method makes again the zones without texture difficult to estimate and it requires the use of a depth regularizer.

It has been lately recognized in the literature that the depth smoothness hypothesis introduces a prior knowledge of the scene that many times is not accurate (Engel et al., 2018; Goodfellow et al., 2016). A better prior hypothesis on the depth of the scene is desirable. A surfel representation for the depth estimation could potentially resolve these issues. This would replace the depth smoothness prior with a more plausible piecewise planar depth prior, preserving strong discontinuities commonly found in man-made structures. Also, the representation of a part of the scene with the same parameters could avoid the need for a regularizer. Many works found in the literature have utilized surfels to represent the surface of the scene, but always using depth camera sensor like (Wang et al., 2019; Whelan et al., 2016; Yan et al., 2017).

This work presents a dense monocular SLAM system that does not utilize the depth smoothness prior. This method accomplishes the map estimation task by modeling the map, not as a group of independent points, but as a group of surfels. This takes advantage of the knowledge that the depth of most scenes can be modeled as a piecewise planar surface, instead of just a group of independent pixels. Every piecewise portion of the scene can be estimated with a surfel. This approach allows to directly restore robust and accurate pixel depths, even in zones where there is no good texture observation. Furthermore, this approach avoids the need for a regularization step. This way we can save on computational resources and, more importantly, it does not add a physically unlikely smoothness prior. New surfels can be initialized with the information of neighbor surfels. The entire system was written in the GLSL shading language, thus using the onboard GPU to achieve real-time. As far as the authors know, this is the

first time this approach is used for monocular depth estimation. The method was tested on various data sets and the results show that both the depth and the surfels can be recovered correctly, demonstrating the potential of the method to reconstruct the scene with high quality.

## 2. Direct surfel optimization

In this section, we fully detail our depth estimation approach by direct surfel optimization. This section is structured as follows. First, in section 2.1, we explain which parameters are needed to describe the surfels. Then, in section 2.2, we explain how we obtain the inverse depth of the entire scene from the surfels set. Afterward, in section 2.3, we explain how we initialize the surfles's parameters, before the optimization starts. In section 2.4 we present our surfel parameter optimization approach, using only the camera's raw measurements (thus making this a direct method). Finally, in section 2.5, we explain how this surfel set is used to estimate the scene depth from another camera pose.

### *2.1. Surfel parameterization*

Each surfel position $\boldsymbol{p}_s$ and normal $\boldsymbol{n}_s$ are defined relative to the pose $\boldsymbol{P}_{kf} \in$ SE(3) of a reference frame $\boldsymbol{F}_{kf}$. The surfels have 3 free parameters, 2 belonging to the degrees of freedom of its normal $\boldsymbol{n}_s$, and the other given by the inverse depth $\boldsymbol{id}_s$ of the ray $\boldsymbol{r}_s$ going from the origin of the reference frame to $\boldsymbol{p}_s$, whose coordinates are given by

$$p_s = r_s / id_s \tag{1}$$

The radius of the surfel r is selected so that all surfels have the same frame-space area in the reference frame $F_{kf}$, regardless of its normal or depth. This is done to guarantee the same quantity of pixel observation for every surfel, this way allowing to gather enough information for the Gauss-Newton estimation process.

Along with the aforementioned parameters, the last residual of each surfel and the time that the surfel was last seen and updated are stored for later use.

### *2.2. Inverse depth calculation*

With the information provided by the estimated surfel parameters, the inverse depth $id_u$ of every pixel u belonging to the reference frame $\boldsymbol{F}_{kf}$ can be obtained. Every $u$ has a corresponding ray $r_u$ with coordinates

$$r_s = K^{-1}\pi^{-1}(u) \tag{2}$$

where $K$ is the camera matrix, π() is the homogenizing function π([x,y,z]) = [x/z,y/z,1] and $u \in \mathbb{R}^3$ is a homogeneous pixel coordinate relative to $F_{kf}$. To be in the same plane, the point $p_s$ and $p_u = r_u/id_u$ must satisfy $n_s(p_s - p_u) = 0$, so $id_u$ is computed as

$$id_u = \frac{(K^{-1}\,\pi^{-1}(u))\cdot n_s}{(r_s/id_s)\cdot n_s} \tag{3}$$

The inverse depth $id_u$ will be calculated relative to a surfel s when the distance between $u$ and the coordinates of the surfel $p_s$ projected into the frame $F_{kf}$, $u_s = \pi(Kp_s)$, is less than the radius r of the surfel s in screen space, that is

$$||\pi(Kp_s) - u|| < r \tag{4}$$

The GLSL shading language is utilized for the implementation of the system, so it allows the introduction of a depth buffer for occlusion removal. This means that, if several surfel project the same pixel screen location, only the pixels with the closest depth to the camera are selected.

During inverse depth computation, along with the depth information the index of the generating surfel $s_{ind}$ is saved, for subsequent utilization as described below.

### *2.3. Surfel parameterization*

To initialize the surfels, the reference frame $F_{kf}$'s inverse depth is computed. Clearly, in areas in the image where no surfel satisfies the equation 4, there will be no inverse depth measurement. These areas are identified as potential location to initialize a new surfel.

First, a search is carried out for an empty pixel $u_{ns}$ where a new surfel can be placed. This area must satisfy that there is no other inverse depth measurement nearby such that $\forall u_{nn}$, $\|u_{ns} - u_{nn}\| > \alpha r$, where $u_{nn}$ is a non-empty pixel.

Then, all neighboring surfels are searched for. A neighbor surfel have to satisfy that a corresponding inverse depth measurement pixel us is nearby, with a distance of less than $\|u_{ns} - u_s\| < \beta r$. The inverse depth for the new surfel is calculated as the mean of the estimated inverse depth, following equation 3 with respect to all nearby surfels $s$. The normal of the new surfel is initialized as the mean of all neighboring surfel's normal.

Insofar as the real surface is globally plane, this type of initialization will give good results. This allows to initialize the surfel with the approximate correct surfel normal and depth, which allows to initialize the Gauss-Newton optimization, without the need to search for surfel initializing parameters.

### *2.4. Surfel optimization*

The parameters of the surfels are estimated so that they minimize the cost function:

$$C(n_s, id_s) = \Sigma_{n\in\omega}\Sigma_{u\in s}|| I_{f_n}(u_p(n_s, id_s) - I_{kf}(u)) ||_h \tag{5}$$

which is the commonly used sum of the Huber normed photometric error (Engel et al., 2018; Newcombe et al., 2011; Pizzoli et al., 2014) between the reference frame intensity $I_{kf}$ and every frame $I_{fn}$ in a group Ω of frames taken before $I_{fn}$.

The coordinates $u_p$ of the pixel u as seen from the frame $I_{fn}$ can be calculated with

$$u_p(n_s, id_s) = \pi(KP_{kf}^{f_n}K^{-1}\pi(u) / id_u(n_s, id_s)) \tag{6}$$

where $P_{kf}^{f_n} \in SE(3)$ changes a point $p$ in the keyframe reference frame to the frame $fn$ reference frame.

To recover the parameters of each surfel $[n_s, id_s]$ that minimizes $C([n_s, id_s])$, a Levenber-Marquat (Engel et al., 2018) Gauss-Newton optimization approach is implemented. The Jacobian $\frac{\partial C([n_s,id_s])}{\partial [n_s,d_s]}$ is obtained by using the chain rule for derivatives:

$$J = \Sigma_{n \in \Omega} \frac{\partial C([n_s,d_s])}{\partial id_u} \frac{\partial id_u}{\partial [n_s,d_s]} \tag{7}$$

$\frac{\partial C([n_s,d_s])}{\partial id_u}$ can be further expressed as

$$\frac{\partial C([n_s,d_s])}{\partial id_u} = \Sigma_{n \in \Omega} \frac{\partial C([n_s,d_s])}{\partial I_f} \nabla I_{f_n}(u_p) \frac{\partial \pi(u)}{\partial u_p} \frac{\partial u_p}{\partial id_u} \tag{8}$$

where $\frac{\partial C([n_s,d_s])}{\partial I_f} = w(I_{f_n}(u_p) - I_{kf}(u))$, $w$ being the Huber norm coefficient correction to the squared norm, $\nabla I_{f_n}$ is the gradient of the frame fn, $\frac{\partial \pi(u)}{\partial u_p} = [1\ 0\ -u_x; 0\ 1\ -u_y; 0\ 0\ 0]$ and $\frac{\partial u_p}{\partial id_u} = KR_{kf}{}^{f_n}K^{-1}\pi^{-1}(u)$, where $R_{kf}{}^{f_n}$ is the relative rotation between the coordinate frames $P_{kf}$ and $P_{fn}$.

Furthermore, $\frac{\partial id_u}{\partial [n_s,d_s]}$ can be expressed as

$$\frac{\partial id_u}{\partial n_s} = \frac{r_s/id_s}{K^{-1}\pi^{-1}(u)\cdot n_s} - \frac{(r_s/id_s)K^{-1}\pi^{-1}(u)\cdot n_s}{K^{-1}\pi^{-1}(u)\cdot n_s} \tag{9}$$

$$\frac{\partial id_u}{\partial d_s} = \frac{K^{-1}\pi^{-1}(u)\cdot n_s}{r_s \cdot n_s}$$

As $\frac{\partial id_u}{\partial [n_s,d_s]}$ does not depend upon $I_{f_n}$ or its pose $P_{f_n}$, it can be taken out of the sum and computed only once for all $I_{f_n} \in \Omega$, which allows to reduce the computational burden of the GPU.

Notice that even $n_s$ has two degrees of freedom, the optimization as calculated with the equation 9 utilizes three degrees of freedom for the normal. This is done to avoid such normal expressions as in cylindrical parameterizations, which adds singularities to the optimization procedure that can cause problems. It was chosen to implement a 3 degree of freedom parameterizations, and then normalize the resulting updated normal

### 2.5. Surfel reference frame change

A new reference frame $F_{kf_1}$ is selected as in the approach described in (Engel et al., 2014). First, the new reference frame pose $P_{kf_0}^{kf_1}$ is estimated, relative to the previous reference frame $P_{kf_0}$. If there are surfels previously estimated in the last reference frame $F_{kf_0}$, its surfels are passed to the new reference frame. The position and normal of each surfel are changed as

$$p_{s_{kf_1}} = P_{kf_0}^{kf_1} p_{s_{kf_0}}$$
$$n_{kf_1} = R_{kf_0}^{kf_1} n_{kf_0} \tag{10}$$

where $R_{kf_0}^{kf_1} \in SO(3)$ is the rotation of $P_{kf_0}^{kf_1}$. The ray $r_{s_{kf_1}}$ can be easily calculated as $\pi(p_{kf1})$.

## 3. Implementation

The proposed system is implemented using the parallel computing capabilities of common GPUs, by utilizing OpenGL core 3.3. This way good performance is achieved without the need to have a high performance system capable of CUDA or OpenCL.

This depth estimation approach is integrated with the LSD-SLAM system (Engel et al., 2014), thus having position estimation and map management. It was found that this system is a good choice, because of its state of the art performance while being open-source and easy to expand upon.

### 3.1. System evaluation

The system was evaluated in 3 benchmark datasets available in (TUM, Department of Informatics, 2017). These datasets depict 3 different scenes. The first dataset was taken on a foodcourt, with tables and benches in view. The street road present in the scene has a very smooth texture, so estimating its depth can be difficult. The second dataset was taken from inside a launch buffet, so the scene has many non-diffuse surfaces, such as the floor and doors. The last scene was taken walking around a machine building, so there are many open spaces and intricate structures. The dataset was obtained using a wide angle global shutter camera. The images are undistorted, and all camera calibration parameters are given by the dataset's authors.

In the following sections, results show how the depth and the normal of the objects are recovered for different environments. The color codes present in the figures are as follows. For the normal estimations, blue represents vectors whose directions are facing to the right edge of the frame, and red represents vectors which are pointing to the right of the

frame. Similarly, green represents vectors pointing up, and yellow are vectors pointing down. For the depth estimation figures, black represents a low inverse depth, and therefore high depth. Conversely, white represents high inverse depths, and so depths close to the camera focal point.

For the following evaluations, the camera image has a 640x480 resolution, and the surfel size was set to a radius of 10 pixels.

**Foodcourt dataset:** Results from the foodcourt dataset are shown in Fig. 1. It can be seen that the resulting depth is correctly recovered, even in

the low texture sections of the street. The first column of figure 1 depicts a backward moving movement. This means that the pixels in the lower part of the image are pixels just entering the frame, and so they have received very few observations. This highlights the usefulness of the surfel initialization scheme explained in section 2.3. The third column of figure 1 depicts a forward movement, showing that both types of movements allow correct normal and depth estimations. In the second column it can be seen that the guarding fence, the food truck and the floor have correct depth and normal estimations. This part of the dataset is particularly challenging, because the guarding fence was seen a few frames ago from the opposite side, so its normals had the inverse direction. The same happens with the normals of the food truck.

**Eccv dataset:** This dataset is particularly challenging, because of the specularities present in almost all of the surfaces. The floor, the ceiling, and even some of the furniture present in the scene have this characteristic. The first column of Fig. 2 shows that the floor depth and normal were correctly estimated, even when there are specularities and blurriness. In the second, third and fourth columns of Fig. 2 it can be seen that the depth and normal of the ceiling were correctly recovered. The movement in the second row was backward, so all of the pixels in the lower part of the frame are new and with very few observations.

Again this shows the usefulness of the surfel initialization.

**Machine dataset:** This dataset has the particularity that it was taken outside, and the sky is a very prominent part of most of the frames. As it was a cloudy day, the sky has texture, even when it is very smooth with a very low gradient. As can be seen in Fig. 3, the depth of the sky was correctly recovered, even with the conditions described before.

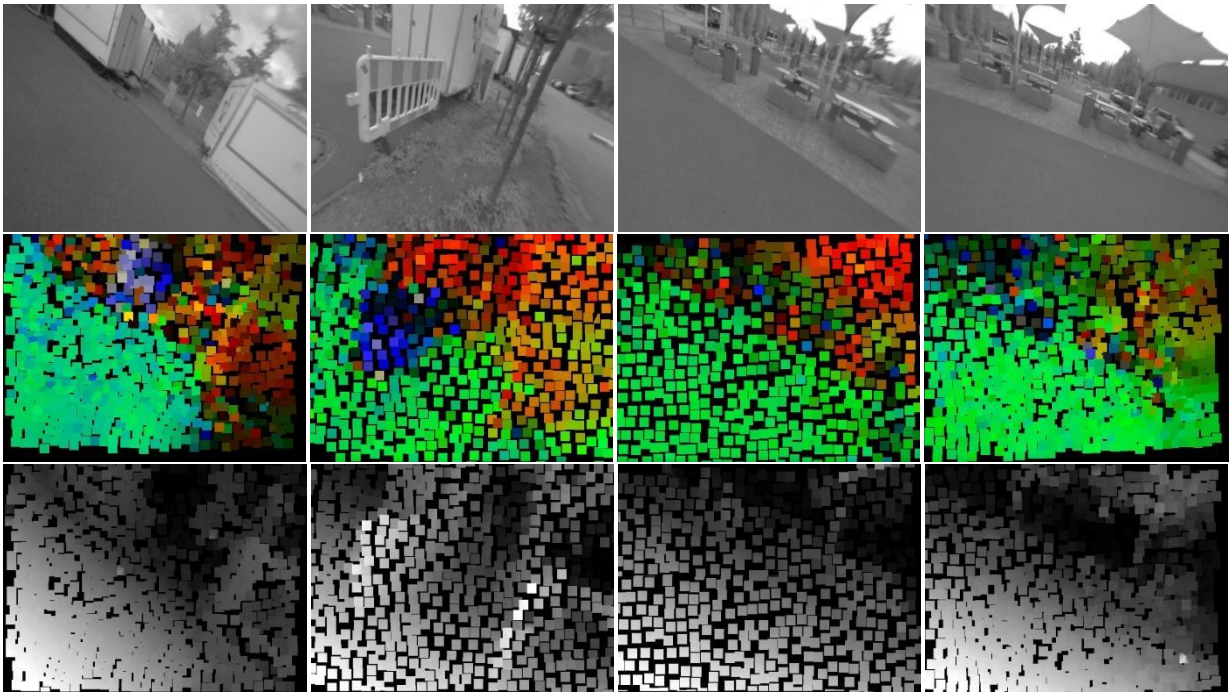

Figure 1. Foodcourt dataset results. First column: raw frame. Second column: estimated normals. Third column: estimated depth.

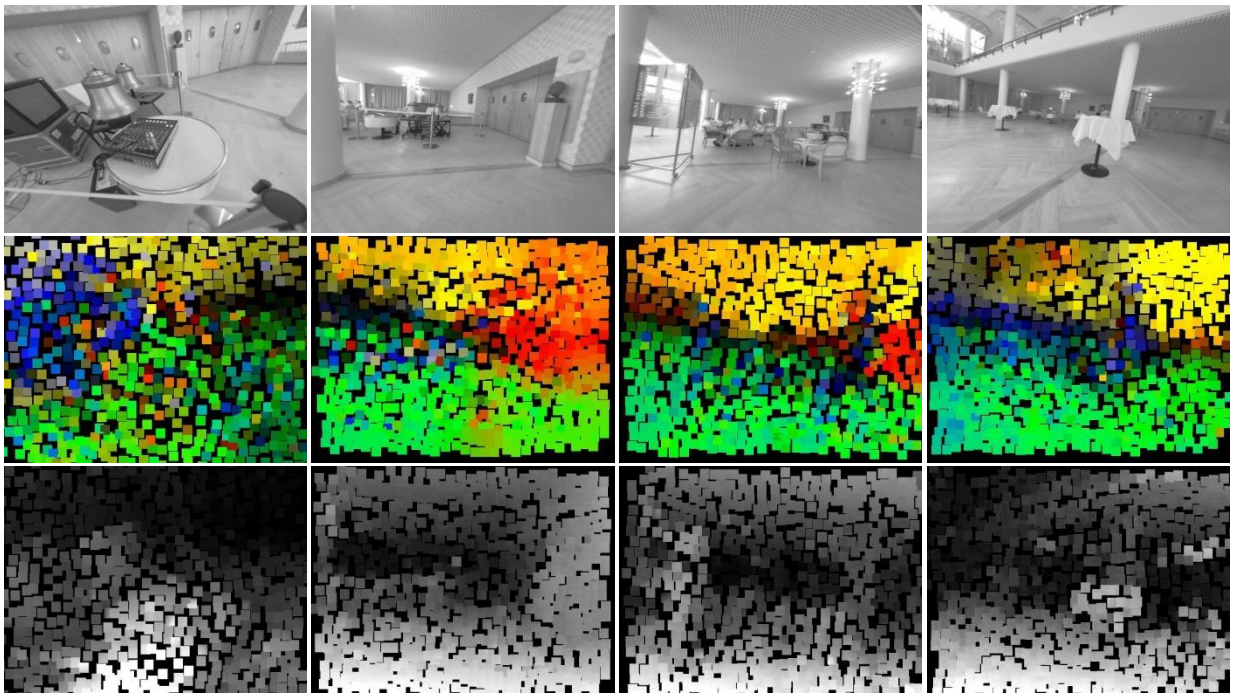

Figure 2. Eccv dataset results. First column: raw frame. Second column: estimated Normal. Third column: estimated depth.

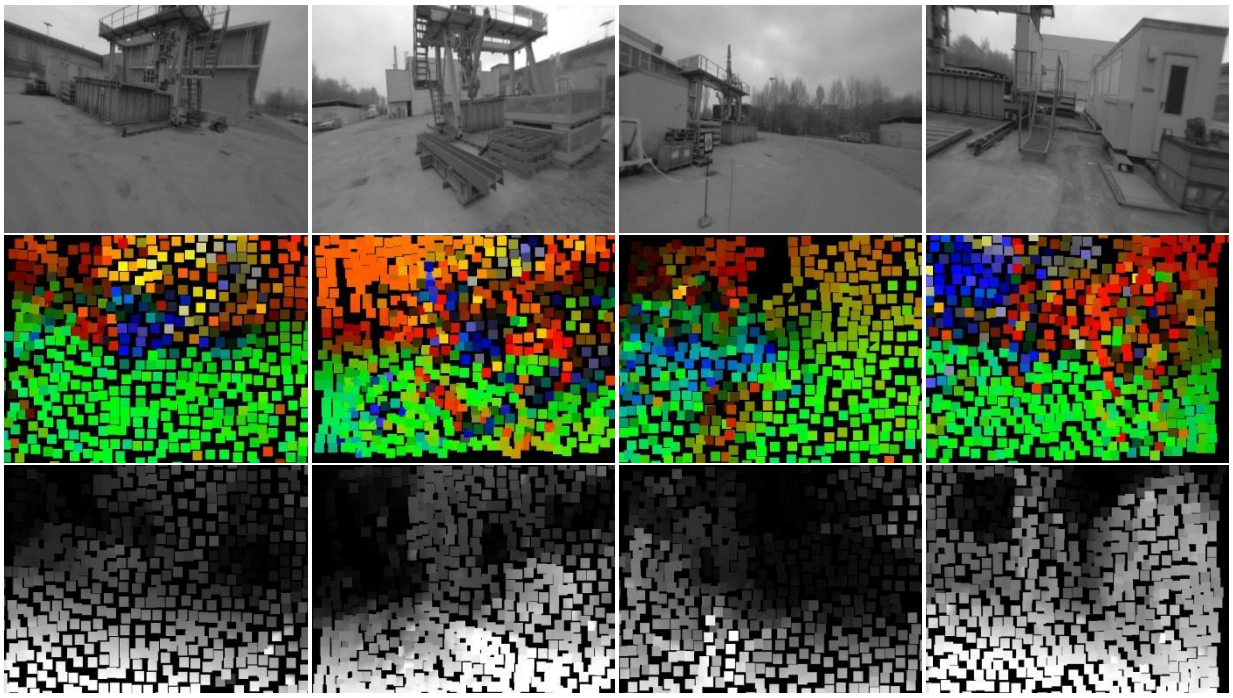

Figure 3. Machine dataset results. First column: raw frame. Second column: estimated normals. Third column: estimated depth.

### 3.2. Usefulness of normal estimation

The usefulness of the joint depth and normal estimation was tested, by modifying the Levenberg-Marquardt estimation algorithm described in 2.4. In equation 9, $\frac{\partial id_u}{\partial n_s}$ was set to [0.0,0.0,0.0], effectively just optimizing for $id_s$, similarly to approaches like (Engel et al., 2018).

The resulting surface reconstruction of the foodcourt dataset with and without normal estimation can be seen in Fig. 4. The resulting scene reconstruction has a quality not present in the reconstruction made without normal estimation.

### 3.3. Impact of surfel size selection

In this evaluation, the surfel radius was set to 12 pixels wide. Results of the scene reconstruction can be seen in Fig 5. The widening of the surfel causes a coarse reconstruction result, losing many details present on the scene. Nevertheless, the proposed approach can reconstruct the scene correctly, including the surfels on the floor and benches. Without using normal estimation the resulting reconstruction has a very poor quality, at the point of almost being unrecognizable, as seen in Fig 5.

## 4. Conclusions

The approach described here can estimate a monocular dense depth map as a set of surfels, directly from the raw image pixels. As far as the authors know, this is the first time this approach is utilized for monocular depth estimation. This presents several advantages over methods more commonly found in the literature, that implement pixel depth estimation followed by a regularization schema. Firstly, there is no need for a computationally costly and physically unlikely depth regularization. Also, the information from the normal estimation can be used to initialize new neighboring surfels, thus being able to initialize the Gauss Newton optimization approach without the need to perform further computations.

The method was tested in several datasets, showing that the depth and surfels can be recovered correctly. The normal estimation allows for a more precise scene reconstruction, even when the surfel pixel radius is incremented considerably.

Most monocular depth estimation approaches present in the literature perform pixel wise depth estimation in conjunction with depth regularization methods. The results obtained in this paper suggest that more plausible depth priors can be utilized for scene reconstruction.

Since the proposed method has as objective fast robust dense depth estimation, unlike most of the methods currently present in the literature that aim just for sparse depth estimation, a proper comparison with the benchmark methods would only be possible in a real-time setting. Therefore, the next step aims to implement our and other benchmark methods on a mobile robot with an integrated GPU computer.

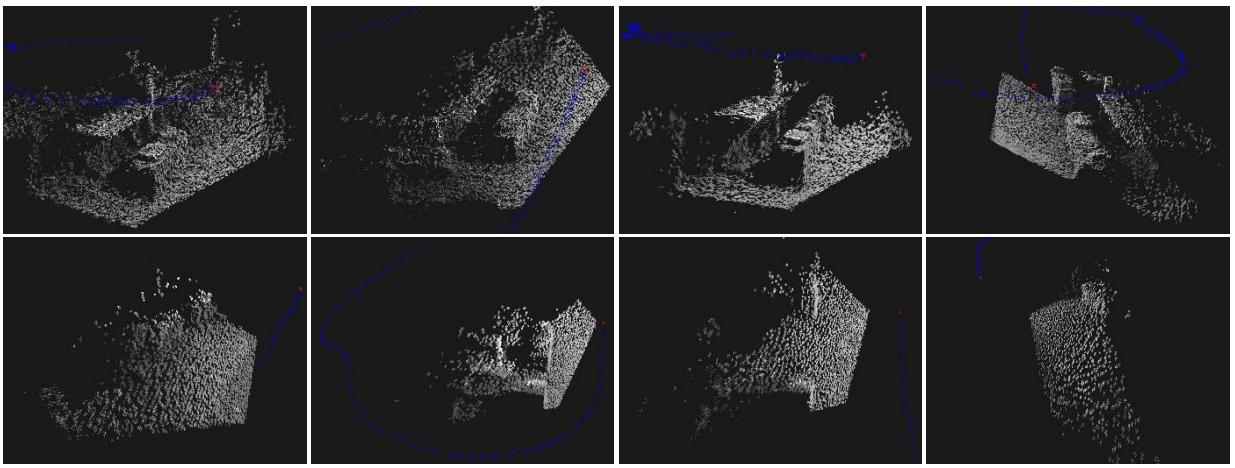

Figure 4. Detail of reconstruction, using a surfel radius of 5 pixels. First row: proposed method. The benches and tables present in the scene are easily recognizable, resulting from the correct surface normal estimation. Second row: without estimating surfels normals. This time the scene cannot be easily recognized, and artifacts caused by incorrect surfel normals are present.

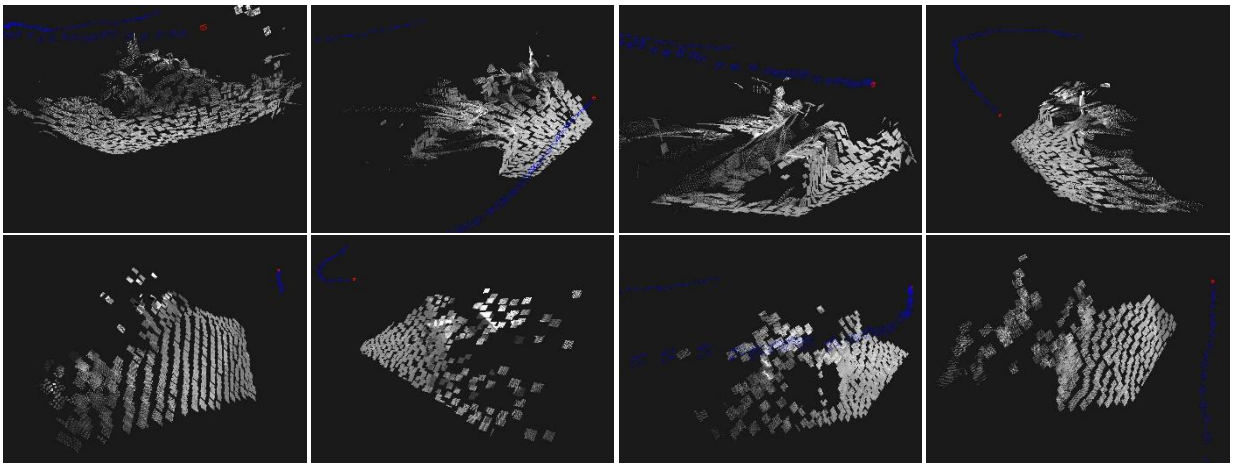

Figure 5. Detail of reconstruction with a surfel radius of 12 pixels. The first row shows results with the present method, while the second row shows results without estimating the surfels normal. With the proposed method, even with such a coarse surfel reconstruction size, the benches and tables present in the scene can be still recognized, and the floor is correctly recovered. On the other hand, without estimating the surfels normal, the scene is almost completely unrecognizable, and there are heavy plane-depth related artifacts.

## References


Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics, 32*(6), 1309–1332. https://doi.org/10.1109/TRO.2016.2624754

Engel, J., Koltun, V., & Cremers, D. (2017). Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence, 40*(3), 611-625. https://doi.org/10.1109/TPAMI.2017.2658577

Engel, J., Schöps, T., & Cremers, D. (2014). LSD-SLAM: Large-scale direct monocular SLAM. In European conference on computer vision (pp. 834-849). Springer, Cham. https://doi.org/10.1007/978-3-319-10605-2_54

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. https://bit.ly/3kPjQOC

Grisetti, G., K¨ummerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine, 2*(4), 31–43. https://doi.org/10.1109/MITS.2010.939925

Hosni, A., Rhemann, C., Bleyer, M., Rother, C., & Gelautz, M. (2013). Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(2), 504-511. https://doi.org/10.1109/TPAMI.2012.156

Lemus, R., Diaz, S., Gutierrez, C., Rodriguez, D., & Escobar, F. (2014). Slam-R algorithm of simultaneous localization and mapping using RFID for obstacle location and recognition. *Journal of applied research and technology, 12*(3),551-559. https://doi.org/10.1016/S1665-6423(14)71634-7

Mur-Artal, R., Montiel, J. M. M., & Tard´os, J. D. (2015). ORB-SLAM: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics, 31*(5), 1147-1163. https://doi.org/10.1109/TRO.2015.2463671

Newcombe, R. A., Lovegrove, S. J., & Davison, A. J. (2011). DTAM: Dense tracking and mapping in real-time. In *2011 international conference on computer vision* (p. 2320-2327) https://doi.org/10.1109/ICCV.2011.6126513

Pizzoli, M., Forster, C., & Scaramuzza, D. (2014). REMODE: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2609-2616). IEEE. https://doi.org/10.1109/ICRA.2014.6907233

TUM Department of Informatics (2017). LSD-SLAM: Large-Scale Direct Monocular SLAM. Accessed: 2017-07-22. https://bit.ly/3qT5fWx

Wang, K., Gao, F., & Shen, S. (2019). Real-time scalable dense surfel mapping. In *2019 International Conference on Robotics and Automation (ICRA)* (p. 6919-6925). https://doi.org/10.1109/ICRA.2019.8794101

Whelan, T., Salas-Moreno, R. F., Glocker, B., Davison, A. J., & Leutenegger, S. (2016). Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research, 35*(14), 1697-1716. https://doi.org/10.1177/0278364916669237

Yan, Z., Ye, M., & Ren, L. (2017, November). Dense visual slam with probabilistic surfel map. *IEEE Transactions on Visualization and Computer Graphics,23*(11), 2389-2398. https://doi.org/10.1109/TVCG.2017.2734458

Zienkiewicz, J., Tsiotsios, A., Davison, A., & Leutenegger, S. (2016). Monocular, real-time surface reconstruction using dynamic level of detail. In *2016 Fourth International Conference on 3D Vision (3DV)* (pp. 37-46). IEEE. https://doi.org/10.1109/3DV.2016.82