Inverse Depth for Accurate Photometric and Geometric Error Minimisation in RGB-D Dense Visual Odometry

Daniel Gutiérrez-Gómez[1], Walterio Mayol-Cuevas[2] and J.J. Guerrero[1]

Abstract— **In this paper we present a dense visual odometry system for RGB-D cameras performing both photometric and geometric error minimisation to estimate the camera motion between frames. Contrary to most works in the literature, we parametrise the geometric error by the inverse depth instead of the depth, which translates into a better fit of the distribution of the geometric error to the used robust cost functions. We also provide a unified evaluation under the same framework of different estimators and ways of computing the scale of the residuals which can be found spread along the related literature. For the comparison of our approach with state-of-the-art approaches we use the popular dataset from the TUM for RGB-D benchmarking. Our approach shows to be competitive with state-of-the-art methods in terms of drift in meters per second, even compared to methods performing loop closure too. When comparing to approaches performing pure odometry like ours, our method outperforms them in the majority of the tested datasets. Additionally we show that our approach is able to work in real time and we provide a qualitative evaluation on our own sequences showing a low drift in the 3D reconstructions.**

I. Introduction

In the robotics community, visual localisation and mapping has become one of the most active research fields in the last decade. Traditional monocular vision systems track sparse features detected in the images, to both estimate the camera pose and build a map either using filtering [6] or bundle adjustment techniques [26] [18]. However, due to their purely projective nature, monocular vision systems do not directly provide depth measurements of the observed environment. This implies the existence of an unknown scale parameter for the camera poses and map estimates. In a SLAM context, this scale ambiguity results in an increased odometry drift and in initialisation issues.

One straightforward way to address the scale problem is to use stereo vision systems [29], [30], [23] in which a fixed baseline between two cameras allows for depth estimation. However with respect to monocular systems they are much more expensive, bigger and more difficult to calibrate, and also they cannot accurately measure the depth of distant scene points or poorly textured areas.

For this reason the recent advent of new RGB-D sensors has aroused great interest in the development of visual odometry and SLAM systems. Their cheapness and their ability to provide dense depth measurements of the environment in contrast to traditional stereo cameras makes them quite appealing to address not only localisation and mapping but also many other problems for which monocular systems are typically used. The main limitation is their use being limited to indoor environments.

First systems using RGB-D sensors for SLAM [9], [8] tended to adapt the sparse feature-based alignment methods from monocular vision, using the depth information to straightforwardly lift the features to 3D points and occasionally to apply the iterative closest point (ICP) algorithm for refinement of the pose estimate.

Preference for systems using sparse features for localisation might be caused by the fact that in monocular systems, dense odometry estimation inherently forced to estimate the dense depth or optical flow map between frames simultaneously or prior to the camera motion, which lead to ill posed problems with more unknowns than constraints and requiring from the use of regularisation and variational methods for their resolution [28]. In addition to this, dense methods require high computational power for real time computation of pixel-wise operations. In this sense, advent of new generation CPUs and high performance GPUs almost simultaneously to RGB-D sensors allowed for a significant cost reduction of dense algorithms due to new programming paradigms which allowed for high paralellisation of per pixel operations.

One of the first approaches for direct odometry estimation from dense RGB-D is KinectFusion [27], which only from the depth channel, is able to estimate the odometry and a smoothed 3D dense map by using an ICP algorithm. Almost alongside with KinectFusion, methods which take advantage of the dense depth map for direct odometry estimation by pixel-wise minimisation of the photometric error between intensity images were presented in [33] and [1]. These works were followed by [16], where the Student's t-distribution was proposed to model the photometric error and [19], proposing an Efficient Second order Minimisation (ESM) scheme for the estimation of camera motion. Contrary to vision-only systems [28], in these works the knowledge of depth directly allows to solve a, generally, well posed system of equations with just 6 unknowns for the rigid motion parameters by least squares minimisation. However these approaches ignore the geometric error between depth images. Estimation of the dense visual odometry by joint minimisation of photometric and geometric error was proposed initially in [35], followed by [5],[24] and [36], where authors propose to combine the cost functions from [33] and [27]. However, these works use

*This work was supported by spanish project DPI2012-31781, FEDER and FPU scholarship AP-2012-5507.

[1]Daniel Gutiérrez-Gómez and Josechu Guerrero are with the Departamento de Informática e Ingeniería de Sistemas (DIIS) y el Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Spain. {danielgg, josechu.guerrero}@unizar.es

[2]Walterio Mayol-Cuevas is with the Department of Computer Science, University of Bristol, UK. wmayol@cs.bris.ac.uk

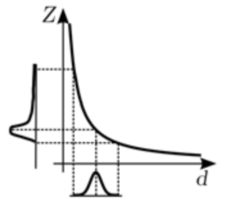

Fig. 1. Assuming that the disparity (d) error follows a Gaussian or, more generally, a symmetric distribution, the depth ($Z \propto \frac{1}{d}$) error distribution is not Gaussian, not even symmetric. The asymmetry is more pronounced for higher Z.

heuristic parameters to weight the contribution of both the photometric and geometric cost functions. In [15], Kerl *et al.* note this issue and propose the computation of an automatic scaling matrix based on the covariance of the photometric and geometric pixel residuals.

In all of the described approaches the use of a constant scaling parameter, either heuristic or automatic, for all the geometric residuals is prone to be a source of inaccuracies in the estimation process due to the quadratic grow of the depth uncertainty in RGB-D sensors [17]. Meilland *et al.*[25] take this fact into account, weighting the depth residuals by the inverse squared depth, but still use additional heuristic parameters to weight photometric and depth residuals. Also, since the depth is inversely proportional to the image disparity, whose error is usually assumed to follow a symmetric distribution, the depth error distribution is neither symmetric (Fig. 1), and thus subject to inaccuracies if minimised with the symmetric cost functions which are typically used, specially for large depths.

In monocular systems the main gain of using the inverse depth (measured along camera optical axis) or the inverse distance (measured along a ray) is the ability to easily deal with points at long distances [4] which leads to an improvement in performance [32]. In RGB-D systems, though there is no need to deal with points at distances greater than the maximum camera range, inverse depth has still some theoretical benefits. The inverse depth is proportional to the to the image disparity by a constant factor, which does not change with depth; and since the disparity error distribution is symmetric, inverse depth fits better with the symmetric error distributions implicitly assumed when minimising the sum of squared errors or any of the robust cost functions present in the related literature. In spite of this, to the best of our knowledge, the use of inverse depth for dense visual odometry with RGB-D cameras has been only proposed in [22] by Lui *et al.*, by extending the ICP algorithm from KinectFusion. However its performance is not benchmarked in large sequences against state-of-the-art methods for dense RGB-D odometry estimation.

The main contribution of this work is the formulation of a dense visual odometry algorithm which minimises both the photometric and the geometric error with the novelty of using an inverse depth parametrisation for the geometric error. We experimentally validate in the TUM benchmarking datasets [34] the better performance of the inverse depth-based geometric error. As additional contribution, though equivalent in essence, the problem formulation is slightly varied with respect to related works, first linearising the flow equations, obtaining generic linear 3D flow equations and then applying the assumption of rigid scene motion to get the linear constraints just on camera motion parameters. We also compare the performance using different robust cost functions (Huber, Tukey biweight and Student's t distribution-based estimator) and also two different methods in the state of the art to compute the uncertainty-based scaling parameters of an error distribution: one with the Median Absolute Deviation (MAD), and one with its Maximum Likelihood (ML) estimator given the cost function used for robust optimisation.

II. Linear Visual Odometry Constraints from Optical Flow

In this section we derive the visual odometry pixel-wise constraints through the flow equations obtained from the photometric and geometric constraints between two camera positions.

A. Optical flow equations

Let us denote two camera frames as A and B, at instants t and $t + \Delta t$ respectively. Given the intensity images $\mathcal{I}_A$ and $\mathcal{I}_B$, and inverse depth maps $\mathcal{W}_A$ and $\mathcal{W}_B$ defined over the image domain $\Omega \subset \mathbb{P}^2$, for an image point $\mathbf{p} = (u\ v\ 1)^T \in \Omega$ in frame A, the following constraints hold:

$$\mathcal{I}_B(\mathbf{p} + \mathbf{\Delta p}) = \mathcal{I}_A(\mathbf{p}) \tag{1}$$

$$\mathcal{W}_B(\mathbf{p} + \mathbf{\Delta p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T \mathbf{X}_B}, \tag{2}$$

where $\mathbf{X}_B$ is the 3D point lifted from pixel $\mathbf{p} + \mathbf{\Delta p}$ in frame B, $\mathbf{\Delta p} = (\Delta u\ \Delta v\ 0)^T$ is the displacement of one point from frame A to B, and $\mathbf{e}_\mathbf{z}^T = (0\ \ 0\ \ 1)$. The constraint in intensity assumes constant illumination of one scene point. The second constraint is the measurement model of the depth sensor at frame B.

Assuming small pixel displacements between frames we compute the flow equations from (1) and (2):

$$\nabla \mathcal{I}_A(\mathbf{p}) \mathbf{\Delta p} + \mathcal{I}_B(\mathbf{p}) = \mathcal{I}_A(\mathbf{p}) \tag{3}$$

$$\nabla \mathcal{W}_A(\mathbf{p}) \mathbf{\Delta p} + \mathcal{W}_B(\mathbf{p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T \mathbf{X}_B}, \tag{4}$$

where the gradient operators $\nabla \mathcal{I} = \left(\frac{\partial \mathcal{I}}{\partial u}\ \frac{\partial \mathcal{I}}{\partial v}\ 0 \right)$ and $\nabla \mathcal{W} = \left(\frac{\partial \mathcal{W}}{\partial u}\ \frac{\partial \mathcal{W}}{\partial v}\ 0 \right)$.

B. Projection model

A world point $\mathbf{X}$ is projected in the image point $\mathbf{p}$ by:

$$\mathbf{p} = \pi(\mathbf{X}) = \mathbf{K} \frac{\mathbf{X}}{\mathbf{e}_\mathbf{z}^T \mathbf{X}} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \frac{\mathbf{X}}{\mathbf{e}_\mathbf{z}^T \mathbf{X}}, \tag{5}$$

where $\mathbf{K}$ is the conventional calibration matrix, including the camera intrinsic parameters.

Inverse depth measurements $\mathcal{W}(\mathbf{p}) = \frac{1}{\mathbf{e}_\mathbf{z}^T\mathbf{X}}$ allow to lift 2D points from the image to 3D coordinates by the inverse projection function:

$$\mathbf{X} = \pi^{-1}(\mathbf{p}) = \frac{1}{\mathcal{W}(\mathbf{p})}\mathbf{K}^{-1}\mathbf{p}. \tag{6}$$

C. 3D flow equations

Flow constraints (3) and (4), can be manipulated to get constraints on the 3D flow at one pixel, which is denoted as $\mathbf{\Delta X_p} = \mathbf{X}_B - \mathbf{X}_A$.

First we compute the first order Taylor expansion of the inverse depth of one point at frame B:

$$\frac{1}{\mathbf{e}_\mathbf{z}^T\mathbf{X}_B} = \frac{1}{\mathbf{e}_\mathbf{z}^T\mathbf{X}_A} - \frac{1}{(\mathbf{e}_\mathbf{z}^T\mathbf{X}_A)^2}\mathbf{e}_\mathbf{z}^T\mathbf{\Delta X_p} + \mathcal{O}\left(\left|\left|\mathbf{e}_\mathbf{z}^T\mathbf{\Delta X_p}\right|\right|^2\right) \tag{7}$$

$$\approx \mathcal{W}_A(\mathbf{p}) - \mathcal{W}_A^2(\mathbf{p})\mathbf{e}_\mathbf{z}^T\mathbf{\Delta X_p}. \tag{8}$$

Using this relation and the camera projection model we get also:

$$\mathbf{\Delta p} = \mathbf{K}\frac{\mathbf{X}_B}{\mathbf{e}_\mathbf{z}^T\mathbf{X}_B} - \mathbf{K}\frac{\mathbf{X}_A}{\mathbf{e}_\mathbf{z}^T\mathbf{X}_A} \tag{9}$$

$$\overset{(8)}{=} \mathbf{K}\mathbf{X}_B\left(\mathcal{W}_A(\mathbf{p}) - \mathcal{W}_A^2(\mathbf{p})\mathbf{e}_\mathbf{z}^T\mathbf{\Delta X_p}\right) - \mathbf{K}\mathbf{X}_A\mathcal{W}_A(\mathbf{p}) \tag{10}$$

$$\overset{(5)}{=} \mathcal{W}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e}_\mathbf{z}^T\right)\mathbf{\Delta X_p}. \tag{11}$$

And substituting in (3) and (4) we get:

$$\mathcal{W}_A(\mathbf{p})\nabla\mathcal{I}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e}_\mathbf{z}^T\right)\mathbf{\Delta X_p} + \mathcal{I}_B(\mathbf{p}) - \mathcal{I}_A(\mathbf{p}) = 0 \tag{12}$$

$$\mathcal{W}_A(\mathbf{p})\left(\nabla\mathcal{W}_A(\mathbf{p})\left(\mathbf{K} - \mathbf{p}\mathbf{e}_\mathbf{z}^T\right) + \mathcal{W}_A(\mathbf{p})\mathbf{e}_\mathbf{z}^T\right)\mathbf{\Delta X_p} + \\ + \mathcal{W}_B(\mathbf{p}) - \mathcal{W}_A(\mathbf{p}) = 0. \tag{13}$$

D. Rigid motion

We have obtained general flow equations taking small pixel displacement as the only assumption. Only (3) presents a dense optical 2D flow estimation problem [12], while (12) and (13) involve a dense scene 3D flow problem [10]. Both are ill posed problems which require regularisation and variational methods to reach a solution.

We focus instead on RGB-D visual odometry estimation. This implies the assumption of a rigid scene, i.e., the displacements $\mathbf{\Delta X_p}$ of each of the $W_{im}H_{im}$ points projected on the image frame are due only to the motion of the camera, which has 6 DoF. Assuming a small motion described by the rotation and translation pair $({}_A\mathbf{R}^B, \mathbf{r}_B^A) \in \mathbb{SE}(3)$ we have:

$$\begin{aligned}\mathbf{\Delta X_p} &= {}_B\mathbf{R}^A\mathbf{X}_A + \mathbf{r}_B^A - \mathbf{X}_A \\ &= \left(\mathbf{I} + [\boldsymbol{\theta}_B^A]_\times\right)\mathbf{X}_A + \mathbf{r}_B^A - \mathbf{X}_A + \mathcal{O}\left(\left|\left|[\boldsymbol{\theta}_B^A]_\times^2\mathbf{X}_A\right|\right|\right) \\ &\approx \mathbf{r}_B^A - [\pi^{-1}(\mathbf{p})]_\times\boldsymbol{\theta}_B^A = \mathbf{M}(\mathbf{p})\boldsymbol{\xi}_B^A.\end{aligned} \tag{14}$$

where $[\cdot]_\times$ denotes the antisymmetric matrix from a vector. Note that $\boldsymbol{\xi}_B^A = (\mathbf{r}_B^A;\ \boldsymbol{\theta}_B^A)$ is not a twist, *i.e.*, $\boldsymbol{\xi}_B^A \notin \mathfrak{se}(3)$, since $\mathbf{r}_B^A$ is yet the translation part of the rigid motion. Eq. (14) leads to a well-posed problem with 6 unknowns for nearly $W_{im}H_{im}$ constraints, excluding pixels without depth measurements, with the following residuals:

$$r_\mathcal{I}(\mathbf{p}, \boldsymbol{\xi}) = \mathcal{W}_A(\mathbf{p})\nabla\mathcal{I}_A(\mathbf{p})(\mathbf{K} - \mathbf{p}\mathbf{e}_\mathbf{z}^T)\mathbf{M}(\mathbf{p})\boldsymbol{\xi} + \\ + \mathcal{I}_B(\mathbf{p}) - \mathcal{I}_A(\mathbf{p}) \tag{15}$$

$$r_\mathcal{W}(\mathbf{p}, \boldsymbol{\xi}) = \mathcal{W}_A(\mathbf{p})\left(\nabla\mathcal{W}_A(\mathbf{p})(\mathbf{K} - \mathbf{p}\mathbf{e}_\mathbf{z}^T) + \mathcal{W}_A(\mathbf{p})\mathbf{e}_\mathbf{z}^T\right)\mathbf{M}(\mathbf{p})\boldsymbol{\xi} + \\ + \mathcal{W}_B(\mathbf{p}) - \mathcal{W}_A(\mathbf{p}), \tag{16}$$

which can be straightforwardly minimised by standard Gauss-Newton least squares.

Note that in the monocular RGB case, no depth is provided and only the constraint (15) would be used. Thus $\mathcal{W}_A(\mathbf{p})$ becomes an unknown yielding an ill-posed problem with $W_{im}H_{im}$ constraints for $W_{im}H_{im} + 6$ unknowns, which is solved using optical flow variational methods [28], or by performing variable baseline stereo matching [7].

III. Visual Odometry Estimation by Iterative Optimisation

With the proposed residuals, $\boldsymbol{\xi}_B^A$ is computed as the solution to the following optimisation problem:

$$\boldsymbol{\xi}_B^A = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \rho\left(\frac{r_\mathcal{I}(\mathbf{p}, \boldsymbol{\xi})}{\sigma_{r_\mathcal{I}}}\right) + \rho\left(\frac{r_\mathcal{W}(\mathbf{p}, \boldsymbol{\xi})}{\sigma_{r_\mathcal{W}}}\right), \tag{17}$$

where $\rho(x)$ is a generic cost function which must be symmetric, definite positive and $\rho(0) = 0$. $\sigma_{r_\mathcal{I}}$ and $\sigma_{r_\mathcal{W}}$ are scaling parameters which capture the uncertainty in intensity and inverse depth residuals, and allow for normalisation of residuals in different magnitudes. The choice $\rho(x) = \frac{x^2}{2}$ results in standard least-squares linear optimisation. Nevertheless to gain robustness against outliers, *e.g.*, pixels belonging to non-static elements, robust M-estimators are usually employed. Optimisation with robust cost functions is addressed by the Iteratively Reweighted Least Squares algorithm (IRLS) [11], which results in a linear least-squares problem to be solved at each iteration:

$$\boldsymbol{\xi}_B^A = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{\mathbf{p}\in\Omega} \omega\left(\frac{\check{r}_\mathcal{I}(\mathbf{p})}{\sigma_{r_\mathcal{I}}}\right)\frac{r_\mathcal{I}^2(\mathbf{p}, \boldsymbol{\xi})}{\sigma_{r_\mathcal{I}}^2} + \omega\left(\frac{\check{r}_\mathcal{W}(\mathbf{p})}{\sigma_{r_\mathcal{W}}}\right)\frac{r_\mathcal{W}^2(\mathbf{p}, \boldsymbol{\xi})}{\sigma_{r_\mathcal{W}}^2}, \tag{18}$$

where $\check{r}_\mathcal{I}(\mathbf{p})$ and $\check{r}_\mathcal{W}(\mathbf{p})$ denote the initial residuals computed after updating the camera motion at previous iteration, and the weighting function $\omega(x)$ depends on the used M-estimator. Cost and weight functions for different M-estimators can be found in [37].

Rigid motion between frames is computed in a coarse-to-fine manner using image pyramids, performing a number of iterations at each pyramid level. Let us have the intensity and inverse depth image pairs $\{\mathcal{I}_k,\ \mathcal{W}_k\}$ and $\{\mathcal{I}_{k+1},\ \mathcal{W}_{k+1}\}$, between consecutive frames k and $k+1$. At the start and every time we step down to the next pyramid level, we set $\{\mathcal{I}_A,\ \mathcal{W}_A\} = \{\mathcal{I}_k^{(pyr)},\ \mathcal{W}_k^{(pyr)}\}$, and compute $\{\nabla\mathcal{I}_A,\ \nabla\mathcal{W}_A\}$.

Initial camera motion, expressed by the transform ${}_{k}\mathbf{T}^{k+1}_{(0)}$ is initialised assuming a constant velocity, *i.e.*, ${}_{k}\mathbf{T}^{k+1}_{(0)} = {}_{k-1}\mathbf{T}^{k}$.

After initialisation, the following steps are performed at each iteration γ: image warping, scaling parameters computation, optimisation and pose composition.

A. Image Warping

Image warping is performed at the start of every iteration in order to reset the incremental motion estimate to $\boldsymbol{\xi}_{B}^{A(\gamma)} = 0$, instead of accumulating it. At each iteration, $\{\mathcal{I}_{k+1},\ \mathcal{W}_{k+1}\}$ are warped towards frame k using the current motion estimate ${}_{k}\mathbf{T}^{k+1}_{(\gamma)}$, resulting in the warped images $\{\mathcal{I}_{B}^{(\gamma)},\ \mathcal{W}_{B}^{(\gamma)}\}$. This is done by reverse warping in the following steps:

- Given a pixel $\mathbf{p}$ in the destination warped image, the corresponding pixel $\mathbf{p}_{k+1}^{(\gamma)}$ in the source image is obtained as:

$$\mathbf{p}_{k+1}^{(\gamma)} = \mathbf{p} + \boldsymbol{\Delta}\mathbf{p}^{(\gamma)} = \mathbf{K}\frac{\mathbf{X}_{k+1}^{(\gamma)}}{\mathbf{e}_{\mathbf{z}}^{T}\mathbf{X}_{k+1}^{(\gamma)}}, \tag{19}$$

  with $\mathbf{X}_{k+1}^{(\gamma)} = {}_{k+1}\mathbf{R}^{k}_{(\gamma)}\mathbf{K}^{-1}\mathbf{p}\frac{1}{\mathcal{W}_{k}(\mathbf{p})} + \mathbf{r}^{k(\gamma)}_{k+1}$.

- By using (1) and (2) and resetting $\mathbf{X}_{k+1}^{(\gamma)} = {}_{k+1}\mathbf{R}^{k}_{(\gamma)}\mathbf{K}^{-1}\mathbf{p}\frac{1}{\mathcal{W}_{B}^{(\gamma)}(\mathbf{p})} + \mathbf{r}^{k(\gamma)}_{k+1}$, compute the warped intensity and inverse depth maps, $\mathcal{I}_{B}^{(\gamma)}$ and $\mathcal{W}_{B}^{(\gamma)}$ as:

$$\mathcal{I}_{B}^{(\gamma)}(\mathbf{p}) = \mathcal{I}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)}), \tag{20}$$

$$\mathcal{W}_{B}^{(\gamma)}(\mathbf{p}) = \frac{\mathbf{e}_{\mathbf{z}}^{T}\,{}_{k+1}\mathbf{R}^{k}_{(\gamma)}\mathbf{K}^{-1}\mathbf{p}}{1 - \frac{\mathbf{e}_{\mathbf{z}}^{T}\mathbf{r}^{k(\gamma)}_{k+1}}{\mathcal{W}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)})}}\mathcal{W}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)}), \tag{21}$$

  where $\mathcal{I}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)})$ and $\mathcal{W}_{k+1}(\mathbf{p}_{k+1}^{(\gamma)})$ are obtained by bilinear interpolation, which is efficiently computed with CUDA capable NVIDIA GPUs using *texture memory*. Warping is performed at the level with highest resolution. Once the warping is done, $\{\mathcal{I}_{B}^{(\gamma)},\ \mathcal{W}_{B}^{(\gamma)}\}$ are downsampled to the pyramid level where current optimisation step is taking place.

B. Scaling parameters

We first compute the initial residuals at $\boldsymbol{\xi}_{B}^{A(\gamma)} = 0$:

$$\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p}) = \{\mathcal{I}_{B}^{(\gamma)}(\mathbf{p}),\ \mathcal{W}_{B}^{(\gamma)}(\mathbf{p})\} - \{\mathcal{I}_{A}(\mathbf{p}),\ \mathcal{W}_{A}(\mathbf{p})\}. \tag{22}$$

Scaling parameters $\sigma_{r_{\mathcal{I}}}$ and $\sigma_{r_{\mathcal{W}}}$ can then be computed by the Median Absolute Deviation (MAD):

$$\sigma^{MAD}_{r_{\{\mathcal{I},\mathcal{W}\}}} = 1.4286\,\underset{\mathbf{p}}{\mathrm{med}}\,|\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p}) - \underset{\mathbf{p}}{\mathrm{med}}\left(\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p})\right)|, \tag{23}$$

or alternatively they can be computed by their Maximum Likelihood (ML) estimator, noticing that for any cost function $\rho(\frac{r-\mu}{\sigma})$ we can obtain the associated likelihood function as:

$$f_{\rho}(r|\mu,\sigma) = \frac{K_{\rho}}{\sigma}\exp\left(-\rho\left(\frac{r-\mu}{\sigma}\right)\right), \tag{24}$$

where K_{ρ} is a scaling constant for $\int_{-\infty}^{\infty} f_{\rho}(r|\mu,\sigma)dr = 1$, and μ and σ the location and the scaling parameter of the residuals respectively. The scaling parameters would be computed by iteratively solving:

$$\left[\mu^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}, \sigma^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}\right] = \underset{\mu,\ \sigma}{\mathrm{argmin}} \sum_{\mathbf{p}\in\Omega} \log\sigma + \rho\left(\frac{\breve{r}_{\{\mathcal{I},\mathcal{W}\}}(\mathbf{p}) - \mu}{\sigma}\right), \tag{25}$$

taking $\rho(x) = \frac{x^2}{2}$ in the first iteration to compute the initial seed and then switching to our selected cost function. Location parameters $\mu^{ML}_{r_{\{\mathcal{I},\mathcal{W}\}}}$ are also calculated since the scaling parameters depend on their estimate. To reduce the computational cost we select a sample of $N = 10000$ pixels to compute the scaling parameters.

C. Robust optimisation

We consider for comparison 3 different cost functions frequently used in the related literature: Huber [14], Tukey biweight [2] and a Student's t-distribution-based estimator [21], which will be denoted as Student in advance. The constants for Huber and Tukey estimators are set up to 1.345 and 4.685 respectively. The number of degrees of freedom of the Student estimator is set up to $\nu = 5$ as in [16].

Given an M-estimator, once we have the initial residuals and the scaling parameters the computation of the weights for IRLS is straightforward. This is followed by the computation of $\boldsymbol{\xi}_{B}^{A(\gamma)}$ by solving the linear optimisation in (18).

D. Motion update

After each iteration the motion estimate between frames k and $k+1$ is updated by the current incremental estimate:

$${}_{k}\mathbf{T}^{k+1}_{(\gamma+1)} = \begin{pmatrix} \exp([\boldsymbol{\theta}_{B}^{A(\gamma)}]_{\times}) & \mathbf{r}_{B}^{A} \\ 0 & 1 \end{pmatrix}^{-1} {}_{k}\mathbf{T}^{k+1}_{(\gamma)}. \tag{26}$$

E. Enhancement of the computational performance

Optimisation is performed in a coarse-to-fine scheme at 3 pyramid levels (160x120, 320x240 and 640x480). The naive approach offers the highest precision performing a fixed number of 10 iterations at each level. This results in cost per frame of about 50 ms, which is broke down into the costs of the different processes in Fig. 2. Alternatively, to improve the time performance we consider the following optimisations:

- Skip optimisation on the highest resolution level.
- In Fig. 2 it can be observed that one important fraction of the time is employed in estimating the scale parameters $\sigma_{\mathcal{I}}$ and $\sigma_{\mathcal{W}}$. This cost can be completely eliminated if we fix the scaling parameters for the optimisation. We propose taking $\sigma_{\mathcal{I}} = 5$, with $\mathcal{I}(\mathbf{p}) \in (0,\ 255)$ and $\sigma_{\mathcal{W}} = 0.0025$ m^{-1}. The choice for $\sigma_{\mathcal{I}}$ is justified by tests with static sequences while $\sigma_{\mathcal{W}}$ stems from the precision of the disparity measurements [20].
- Instead of warping at the highest resolution and then down-sampling at each iteration, a coarser but more efficient alternative would be downsampling $\{\mathcal{I}_{k+1},\ \mathcal{W}_{k+1}\}$ before optimising and warping on the current pyramid level.

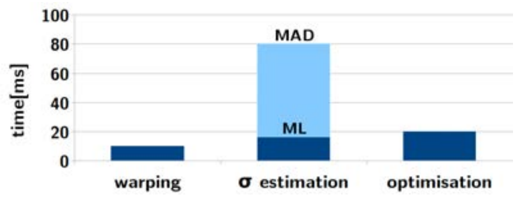

Fig. 2. Costs of the processes involved in the computation of the RGB-D visual odometry.

IV. Experiments

We first evaluate the precision of our approach with different configurations comparing it to other works in the literature using the RGB-D dataset from Technische Universität München (TUM) [34]. Secondly we study with the same datasets, how both the precision and the computation speed are affected when applying the options for enhancement of computational performance from section III-E. Finally we show a qualitative evaluation of our visual odometry method showing the 3D reconstructions obtained in our own sequences.

The experiments were performed on a desktop computer with Ubuntu 12.04 32-bits and equipped with an Intel Core i5-2500 CPU at 3.30 GHz, 8GB and a NVIDIA GeFroce 660GTX GPU with 2GB of memory. The implementation was done as an extension of the large scale KinectFusion large scale algorithm [3] from the Point Cloud Library (PCL) [31], where the original ICP system for odometry estimation has been completely substituted by our method. The algorithms for dense volumetric mapping and volume shifting are kept unchanged. A fork of PCL including the described approach is available for download[1].

A. Precision

We evaluated all the possible combinations of robust cost functions (Huber, Tukey and Student), geometric error parametrisation (depth, inverse depth) and residual scale estimators (MAD, ML). Table I shows the different values of the RMSE for the translational drift, measured in m/s, using different approaches. Best results are obtained with the Student estimator, with little difference between using MAD or ML estimators for the scaling parameters. Parametrisation of the geometric error with inverse depth yields an improvement over the depth parametrisation in all the datasets except in $fr2/desk$ where the performance both geometric error parametrisations is almost identical. The performance of Student estimator is better in general, and more stable when changing the rest of configurable parameters. Huber estimator offers in general the lowest precision. Tukey estimator has a bad performance if we use the ML estimator for σ. However its performance is comparable to Student's if the MAD estimator is used.

With respect to other state-of-the-art RGB-D visual odometry methods, our frame-to-frame approach has the lowest error in the $fr1/desk2$ and $fr2/desk$ datasets. In the

[1]`https://github.com/dangut/pcl`

TABLE I

Translational drift relative root mean square error (RMSE) in meters per second using different methods for RGB-D visual odometry estimation

Estimator	Geom. error	σ	fr1/desk	fr1/desk2	fr1/room	fr2/desk
Student	depth	ML	0.0278	0.0425	0.0504	**0.0115**
Student	depth	MAD	0.0271	0.0439	0.0490	0.0121
Student	invDepth	ML	**0.0260**	**0.0387**	0.0491	0.0121
Student	invDepth	MAD	**0.0260**	0.0396	0.0485	0.0122
Tukey	depth	ML	0.0292	0.0808	0.0502	0.0143
Tukey	depth	MAD	0.0271	0.0527	0.0483	0.0142
Tukey	invDepth	ML	0.0381	0.0720	0.0485	0.0135
Tukey	invDepth	MAD	0.0287	0.0422	**0.0471**	0.0131
Huber	depth	ML	0.0322	0.0495	0.0681	0.0193
Huber	depth	MAD	0.0322	0.0496	0.0662	0.0233
Huber	invDepth	ML	0.0289	0.0453	0.0640	0.0209
Huber	invDepth	MAD	0.0280	0.0435	0.0606	0.0219
FOVIS ([13], comp. in [36])			0.0604	-	0.0642	**0.0136**
ICP+RGB-D [36]			0.0393	-	0.0622	0.0208
ESM + Tukey + Aff. Il. [19]			0.0302	0.0526	0.0397	0.0147
VP [24]			0.0259	-	**0.0351**	0.0147
RGB+D [15]			0.036	**0.049**	0.058	-
RGB+D+KF [15]			0.030	0.055	0.048	-
RGB+D+KF+Opt [15]			**0.024**	0.050	0.043	-

TABLE II

Translational drift relative root mean square error (RMSE) in meters per second minimising different types and combinations of errors

Geom. error	Min. errors	fr1/desk	fr1/desk2	fr1/room	fr2/desk
none	phot	0.0312	0.0513	0.0503	0.0119
invDepth	phot+geom	0.0261	0.0395	0.0502	0.0120
depth	phot+geom	0.0275	0.0434	0.0511	0.0116
invDepth	geom	0.0332	0.0454	0.0495	0.0356
depth	geom	0.0377	0.0562	0.0555	0.0465

$fr1/desk$ the precision is slightly worse compared to the method with lowest error, while for $fr1/room$, [24] shows the best precission. For the rest of the experiments if not otherwise specified we use the configuration with Student robust cost function, inverse depth parametrisation for the geometric error and the ML estimator for the scale parameters.

For a more detailed evaluation of how the combination of both geometric and photometric errors affect the precision of the estimate and better assess the gain of using inverse depth instead of depth for the geometric error, we have performed another set of tests minimising the photometric error only, geometric error only and the combination of both photometric and geometric error. For the cases where geometric error is used we have switched between inverse depth and depth based errors. It can be observed in Table II that when considering the geometric error only the superiority of using inverse depth residuals is clearer.

B. Performance vs precision

We evaluated how the precision and computational performance are affected after applying the modifications proposed in Sec. III-E to decrease the computational cost of our approach. Following the order in which they are presented, we denote these as lvl followed by the level index at which optimisation is stopped, σFix and $pyrFirst$. Results of using one or a combination of these modifications are shown in Table III. It can be observed that either stopping optimisation at level 1 or using constant scale we achieve a

TABLE III

TRANSLATIONAL DRIFT AND AVERAGE AND MAXIMUM COMPUTATION TIME PER FRAME FOR DIFFERENT OPTIONS TO ENHANCE THE COMPUTATIONAL PERFORMANCE

Approach	RMSE[m/s]				time	
	fr1/desk	fr1/desk2	fr1/room	fr2/desk	mean[ms]	max[ms]
naive	0.0260	0.0388	0.0490	0.0120	47	50
lvl1	0.0268	0.0407	0.0492	0.0120	26	28
pyrFirst	0.0270	0.0401	0.0491	0.0127	40	42
pyrFirst+lvl1	0.0282	0.0416	0.0498	0.0138	18	20
σFix	0.0260	0.0389	0.0498	0.0112	34	35
σFix+lvl1	0.0271	0.0399	0.0500	0.0121	17	17
σFix+pyrFirst	0.0272	0.0397	0.0498	0.0118	26	27
σFix+pyrFirst+lvl1	0.0287	0.0409	0.0500	0.0143	9	10

computation time in the limits of the camera frame rate of 30 Hz with little lost in precision, and even we can reach a reduction to 9ms applying all the proposed optimisations.

C. 3D reconstruction

Though for volumetric 3D mapping we use the original functions in KinectFusion, a good quality of the reconstructed dense 3D volume depends critically on the drift introduced by the visual odometry algorithm. Thus, we also present qualitative results of our approach showing the reconstruction of some of the tested RGB-D datasets from the TUM, and also two different datasets acquired by us in one laboratory of approximately 90 m^2 and the corridor of our department with a length of more than 80 m.

Our acquisitions were carried out with an Asus Xtion Pro RGB-D camera attached to a laptop by an arm-clamp system. The camera was only calibrated with a linear pinhole model without distortion parameters for the RGB sensor. The depth sensor is not calibrated, taking the depth values directly as provided by the sensor; and we use the hard-coded stereo pair calibration for depth and RGB registration. All the reconstructions are obtained just using our modified version of KinectFusion without performing loop closure.

Qualitative results for the datasets $fr1/desk$ and $fr1/room$ are shown in Figs. 3 and 4. It can be observed the great level of detail in $fr1/desk$, which indicates a low drift in the reconstruction. In the $fr1/room$ there are some zones, like the table at right, where the quality of the 3D reconstruction is poor. This occurs generally when mapping the same area under large camera motion and when revisiting a previously mapped place. In these cases new depth maps integrated into the mapped volume conflict with the stored map generating artefacts. However for zones which are swept during less time, as occurs in the rest of the sequence, the drift during mapping is low and the map reconstruction is more accurate. Attached to this paper we provide a video demonstration of our method on the $fr2/desk$ dataset.

Results for our laboratory and the corridor sequences are shown in the figure 5 and 6 respectively. The precision of the reconstruction can be assessed from the comparison to RGB images from similar points of view. Note also that given that our method is frame-to-frame and does not perform loop closure, the drift both in the laboratory, reflected in the discontinuity in the right wall and in the corridor, reflected in the slight curvature of its side view, are relatively low.

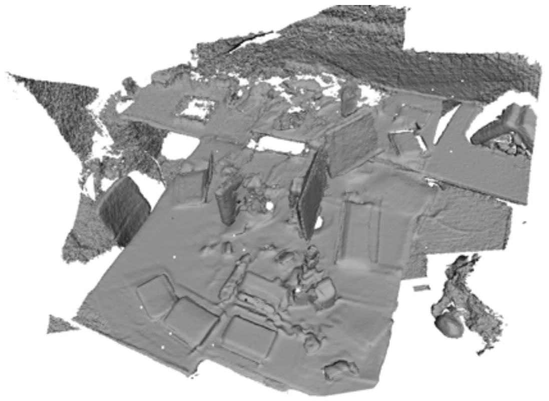

Fig. 3. Dense 3D reconstruction of the $fr1/desk$ dataset.

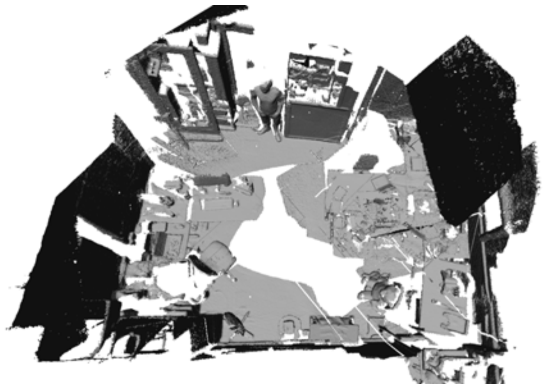

Fig. 4. Dense 3D reconstruction of the $fr1/room$ dataset. Note how the shape of the room is accurately captured. Black part on the right top corner of the $fr1/room$ map corresponds to the ceiling reconstruction viewed from outside the volume.

V. CONCLUSIONS

In this paper we have presented a new visual odometry system on GPU based on the alignment between consecutive frames by minimisation both on the photometric and geometric error. Our system is implemented as an extension of the KinectFusion implementation Kinfu Large Scale in PCL, where the original ICP algorithm for frame alignment and visual odometry computation has been completely substituted by our method. The main contribution of our proposal is using the inverse depth instead of the depth to parametrise the geometric error, as well as allowing to switch between different robust estimators, residuals' scale estimators or geometric error parametrisation for comparative purposes. Our method shows its competitiveness with other state-of-the-art methods outperforming them in the majority of the tested datasets. Also, with the introduction of some changes to increase the computational performance our system is able to reach 9 Hz performance with the GPU device NVIDIA GeForce 660GTX used in the experiments, without hindering too much the accuracy of the method.

REFERENCES

[1] C. Audras, A. Comport, M. Meilland, and P. Rives. Real-time dense rgb-d localisation and mapping. *Aust. Conf. on Robotics and Automation (ACRA)*, 2011.

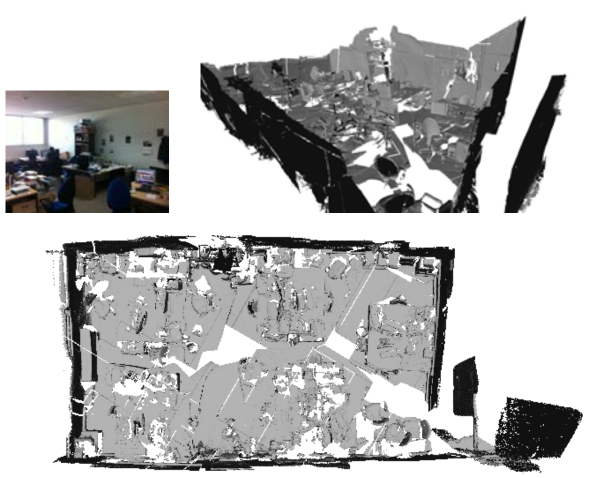

Fig. 5. (Top-left) RGB image of the laboratory, (Top-right) KinectFusion 3D reconstruction using our method for visual odometry and (bottom) plant view of the complete 3D mesh.

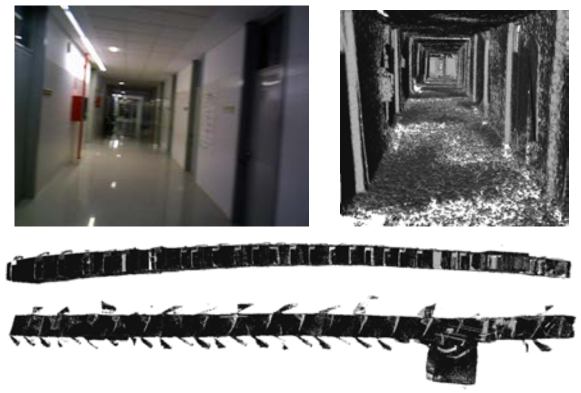

Fig. 6. (Top-left) RGB image of the corridor, (Top-right) pcl KinFu 3D reconstruction with our visual odometry method, (middle) side view and (bottom) plant view of the complete 3D mesh. Note the challenging of the sequence due to the poor texture of the corridor and light reflexes.

[2] A. E. Beaton and J. W. Tukey. The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16(2):147–185, 1974.

[3] E. Bondarev, F. Heredia, R. Favier, L. Ma, and P. H. N. de With. On photo-realistic 3d reconstruction of large-scale and arbitrary-shaped environments. In *IEEE Consumer Communications and Networking Conf. (CCNC)*, pages 621–624, 2013.

[4] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse depth parametrization for monocular slam. *IEEE Trans. on Robotics (T-RO)*, 24(5):932–945, 2008.

[5] D. Damen, A. P. Gee, W. W. Mayol-Cuevas, and A. Calway. Egocentric real-time workspace monitoring using an rgb-d camera. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1029–1036, 2012.

[6] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Int. Conf. on Computer Vision (ICCV)*, volume 2, pages 1403–1410, 2003.

[7] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *IEEE Int. Conf. on Computer Vision (ICCV)*, 2013.

[8] N. Engelhard, F. Endres, J. Hess, J. Sturm, and W. Burgard. Real-time 3d visual slam with a hand-held rgb-d camera. In *RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum*, 2011.

[9] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *Int. Symp. on Experimental Robotics (ISER)*, 2010.

[10] E. Herbst, X. Ren, and D. Fox. Rgb-d flow: Dense 3-d motion estimation using color and depth. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2276–2282, 2013.

[11] P. W. Holland and R. E. Welsch. Robust Regression Using Iteratively Reweighted Least-Squares. *Communications in Statistics: Theory and Methods*, A6:813–827, 1977.

[12] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185 – 203, 1981.

[13] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Fox, and N. Roy. Visual odometry and mapping for autonomous flight using an rgb-d camera. In *Int. Symp. of Robotics Research (ISRR)*, 2011.

[14] P. J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[15] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *IEEE/RSJ Conf. on Intelligent Robots and Systems (IROS)*, pages 2100–2106, 2013.

[16] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for rgb-d cameras. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 3748–3754, 2013.

[17] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012.

[18] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM Int. Symp. on Mixed and Augmented Reality (ISMAR)*, 2007.

[19] S. Klose, P. Heise, and A. Knoll. Efficient Compositional Approaches for Real-Time Robust Direct Visual Odometry from RGB-D Data. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[20] K. Konolige and P. Mihelich. Technical description of kinect calibration. `http://wiki.ros.org/kinect_calibration/technical`, 2010, [Online; accessed 25-September-2014].

[21] K. L. Lange, R. J. A. Little, and J. M. G. Taylor. Robust Statistical Modeling Using the t Distribution. *J. of the American Statistical Association*, 84(408):881–896, 1989.

[22] W. L. D. Lui, T. J. J. Tang, T. Drummond, and W. H. Li. Robust egomotion estimation using ICP in inverse depth coordinates. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1671–1678, 2012.

[23] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. Rslam: A system for large-scale mapping in constant-time using stereo. *Int. J. of Computer Vision (IJCV)*, 94(2):198–214, 2010.

[24] M. Meilland and A. Comport. On unifying key-frame and voxel-based dense visual SLAM at large scales. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[25] M. Meilland and A. I. Comport. Super-resolution 3d tracking and mapping. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 5717–5723, 2013.

[26] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real-time localization and 3d reconstruction. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[27] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Int. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[28] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Int. Conf. on Computer Vision (ICCV)*, pages 2320–2327, 2011.

[29] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23:3–20, 2006.

[30] L. M. Paz, P. Piniés, J. D. Tardós, and J. Neira. Large scale 6 dof slam with stereo-in-hand. *IEEE Trans. on Robotics (T-RO)*, 24(5):946–957, 2008.

[31] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[32] J. Solà, T. Vidal-Calleja, J. Civera, and J. M. Montiel. Impact of landmark parametrization on monocular ekf-slam with points and lines. *Int. J. on Computer Vision (IJCV)*, 97(3):339–368, 2012.

[33] F. Steinbrücker, J. Sturm, and D. Cremers. Real-time visual odometry from dense rgb-d images. In *ICCV Workshops*, pages 719–722, 2011.

[34] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ Int. Conf. on Intelligent Robot Systems (IROS)*, 2012.

[35] T. Tykkala, C. Audras, and A. I. Comport. Direct iterative closest point for real-time visual odometry. In *ICCV Workshops*, 2011.

[36] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald. Robust real-time visual odometry for dense rgb-d mapping. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013.

[37] Z. Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59 –76, 1997.