Kinematic Model based Visual Odometry for Differential Drive Vehicles

Julian Jordan[1] and Andreas Zell[1]

***Abstract*— This work presents KMVO, a ground plane based visual odometry that utilizes the vehicle's kinematic model to improve accuracy and robustness. Instead of solving a generic image alignment problem, the motion parameters of a differential drive vehicle can be directly estimated from RGB-D image data. In addition, a method for outlier rejection is presented that can deal with large percentages of outliers. The system is designed to run in real time on a single thread of a mobile CPU.**

The results of the proposed method are compared to other publicly available visual odometry and SLAM methods on a set of nine real world image sequences of different indoor environments.

I. INTRODUCTION

A precise and robust self localization is a crucial component for many robotic tasks. Although a variety of different approaches exist, most publicly available methods are designed to work with a forward looking sensor. For obstacle detection and avoidance, a downward facing camera is more suitable to perceive obstacles directly in front of the vehicle, and for reliable detection of negative obstacles. Having images mostly displaying the ground is not beneficial for visual odometry for several reasons, especially in indoor environments: it can have a very low contrast, quickly repeating texture, the ground plane does not provide geometric information and is close to the camera, increasing the effect of motion blur. Additionally, reflections and overexposure are more likely to occur. In order to achieve robust and reliable results with the described setup, an adapted visual odometry is required.

This work proposes a visual odometry method specifically designed to work with a downward facing camera, mounted on a differential drive vehicle. Due to the non-holonomic constraints of a differential drive, the motion can be described by two parameters. It is shown that estimating these two parameters directly from image data can significantly improve the accuracy of the estimated motion. Furthermore, an efficient outlier rejection scheme is presented. It operates in the model parameter space instead of the data space. This increases robustness against image noise and allows removal of large outlier areas. The performance of the proposed method is evaluated on nine real world datasets and compared to five state-of-the-art approaches and its predecessor. KMVO is able to outperform all six methods used for comparison, while being able to run at 60hz on a single thread of a mobile CPU.

[1]Julian Jordan and Andreas Zell are with the Faculty of Science, Dept. of Computer Science, University of Tuebingen, 72076 Tuebingen, Germany julian.jordan,andreas.zell@uni-tuebingen.de

The outline of the paper is as follows: Section II gives a review of related work. In Section III the hardware setup and the orthogonal projection preprocessing step are described. Section IV derives the image warping function for the differential drive model. The image alignment and outlier rejection are described in Section V. The evaluation method and evaluation results are presented in Section VI. Finally, a conclusion is given in Section VII.

The contributions of this paper are:

- A formulation that allows to estimate the motion parameters of the kinematic model directly from image data.
- An efficient motion parameter based outlier rejection scheme.
- Error handling for situations where the kinematic model cannot describe the actual motion.
- Performance comparison with six existing visual odometry and visual SLAM methods.

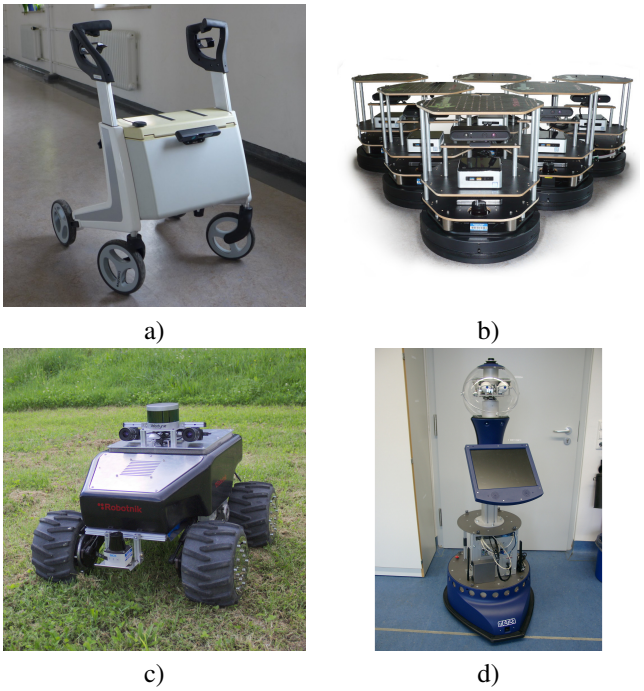

Fig. 1. a) The Bemotec beActive+e electric wheeled walker used in this work. b) Kobuki Turtlebot. c) Robotnik Summit XL. d) Metralabs Scitos G5. The proposed method is suitable for all four vehicles, since they are all based on a differential drive.

II. RELATED WORK

Visual odometry and visual SLAM are important components in many mobile robotic applications, from service

robots to autonomous cars. Numerous methods exist employing different approaches to estimate camera motion. Typically they can be classified into two categories: feature based and direct methods. Feature based methods use descriptors extracted at key points to establish correspondences between images. This reduction of data to be processed greatly improves computational performance, but also induces the loss of potentially useful information. A popular descriptor is ORB [1], which is used in [2] and [3]. Both methods include loop closure using Bag-of-Words, pose graph optimization and RANSAC for outlier rejection. [2] additionally performs bundle adjustment to refine map point positions. Feature based methods are especially prone to motion blur: if the appearance of a key point changes too much, the correspondence fails. If the whole image is affected by motion blur this may cause loss of tracking.

Direct methods try to minimize the photometric error between two or more images, mostly employing a Lucas-Kanade method [4]. Several improvements and extensions of this method were presented: e.g. [5] or [6]. They can further be split into sparse methods and dense methods. Sparse methods select portions of the image, e.g. based on the amplitude of the gradient as in [7] and [8]. As for feature based methods, this information selection can discard useful information, although [8] argues that image data is highly redundant and the effect of additional pixels decreases fast. Dense methods like [9] process the whole image, mitigating the risk of loosing important information, but increasing computational requirements. For outlier rejection, [7] and [9] use weights that are based on image residuals. Different weighting functions are tested: Huber and Tukey in [7] and t-distribution and Tukey in [9]. Using these weights for iteratively re-weighted least-squares can decrease the influence of outliers up to a certain outlier ratio [9].

For wheeled robots it is possible to reduce the degrees of freedom to three since they should move in an at least locally planar environment. This constraint allows to perform scale free monocular odometry [10], [11] and [12]. All three methods are designed for vehicles with two motion parameters. But they use a three parameter representation for image warping. This may lead to image alignment results that contradict the vehicle's motion model. While [10] uses non-holonomic constraints for efficient recalibration, they are not employed directly in the image alignment process. The non-holonomic constraints of a wheeled vehicle can be exploited to improve motion estimates: a feature based approach for an Ackermann based vehicle is described in [13].

III. Setup

The locally planar environment in which a wheeled robot moves in allows to describe the vehicle motion with three instead of six degrees of freedom. Planarity also allows the use of a kinematic model of the vehicle, allowing to decrease the number of parameters required to estimate to two, if employed on a differential drive vehicle. The vehicle used in this work is an electric wheeled walker equipped with additional sensors to make it an intelligent personal mobility assistant. As such there are limitations in terms of computational resources, weight and sensor costs. Its main purpose is to increase the mobility of visual or cognitive impaired people by helping them avoiding obstacles. This requires a robust and accurate visual odometry which works in different environments under challenging conditions.

A. Robotic Platform

The prototype is based on a Bemotec beActive+e electric wheeled walker shown in Fig. 1. It is further equipped with a Sick TiM561 LIDAR, a U-Blox GPS Module, a Razor 9DoF IMU and an Asus Xtion Pro Live, which is the sensor used for recording the data required by the proposed method. Although it is a shared control vehicle, its motion can still be described by the kinematic model of a differential drive robot with two powered rear wheels and two caster-like front wheels. The maximum electrically supported velocity is $0.81m/s$, which is the maximum velocity used for evaluation.

B. Orthogonal Projection

In order to perform 3 degrees of freedom image alignment the images have to be projected onto the ground plane. Since the RGB-D sensor provides depth information along with an RGB image, it is possible to perform an orthogonal projection of the input data along the z-axis. In addition to the RGB or intensity image, a mask image is stored describing which pixels are valid. Invalid pixels are not included in the optimization process. This allows to perform visual odometry in environments which are not planar, like in Fig. 5 b). Only the vehicle motion must follow the planarity constraint. See [14] for details of this orthogonal projection. For monocular cameras, the input images can be projected onto the previously calibrated ground plane using a homography. However, as described in [11], every deviation from this ground plane will affect the estimated result negatively.

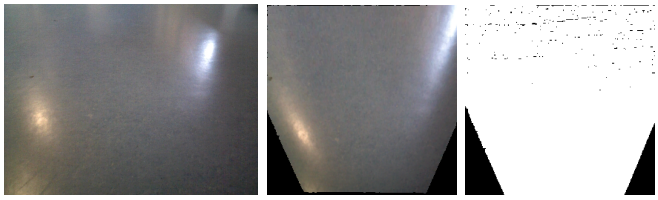

Fig. 2. Left: Original RGB image of a low contrast environment, Center: The orthogonal projection. Right: Mask image.

IV. Kinematic Model

In general the image alignment process for visual odometry consists of three components: (1) an optimization method for iteratively solving the non-linear problem, commonly Gauss-Newton or Levenberg-Marquardt are chosen. (2) a linearisation method e.g. Forward Compositional, Inverse Compositional or Efficient Second Order Minimization and (3) a warp parameter representation like $\mathfrak{se}2$ for 3DoF or $\mathfrak{se}3$ for 6DoF image alignment. See [5] and [6] for a more

detailed description of different methods for optimization, linearisation and warping. As a wheeled robot moves in an at least locally planar environment, the pose can be described by three parameters: x, y, θ. Given images depicting the ground plane, the visual odometry problem can be solved by finding the warp that minimizes the sum of squared differences between these images. Therefore the $\mathfrak{se}2$ parametrisation is an obvious choice for describing the image warp, as shown in [11] and [10]. Many wheeled vehicles are non-holonomic, like differential drive or Ackermann based vehicles, allowing to describe their motion by only two parameters while the robot itself moves in a 3DoF world. This over-parametrisation creates an ambiguity: Due to the locally linear character of small angle rotations it can be confused with a translation and vice versa. This problem increases with the distance from the image position to the center of rotation. This, especially in scenes with sub optimal image quality, can result in a motion estimate that minimizes the photometric error but describes a motion that is not feasible for the vehicle. Experiments have shown that for a differential drive vehicle with a camera mounted in the front and two powered wheels in the back, as depicted in Fig. 3, a distance from the rotation center to the sensor of about 1m is already large enough to observe this effect.

Conversely, if the vehicle performs a motion that cannot be described by the kinematic model, e.g. due to wheel slip, the proposed method detects this by comparing the values of the error functions of the $\mathfrak{se}2$ with the kinematic model alignment. If the error ratio exceeds a given threshold, the $\mathfrak{se}2$ alignment, as described in [14], is used.

A. Overview

The proposed method consists of the following processing steps:

1) Orthogonal projection of the RGB image and conversion to a gray scale image.
2) Full image alignment with $\mathfrak{se}2$.
3) If rotation and lateral motion are below a threshold go to 7).
4) Full image alignment with kinematic model.
5) Outlier rejection with kinematic model.
6) If kinematic model alignment is successful go to 8).
7) Outlier rejection with $\mathfrak{se}2$ parametrisation.
8) Update of the vehicle pose.

The full image alignment with $\mathfrak{se}2$ parametrisation is also used to reduce the number of iterations of the model based alignment by providing an initial estimate for r and $\Delta\theta$.

B. Differential Drive Model

A vehicle pose in a 2D world at time t is described by $\mathbf{p}_t = (p_{xt}, p_{yt}, p_{\theta t})$. The differential drive model describes the vehicle's motion with two parameters: The distance of the rear axis center to the center of rotation r and the rotation angle $\Delta\theta$, see Fig. 3. Since the vehicle body is rigid, all points on the vehicle perform a rotation around the same center by the same angle. This includes the position of the camera and its field of view, allowing to estimate the parameters $\mathbf{m}' = (r, \Delta\theta)$ directly from the image data. After estimating $\mathbf{m}'$ the robot pose is updated using:

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \begin{pmatrix} r(\sin(p_{\theta t} + \Delta\theta) - \sin(p_{\theta t})) \\ r(-\cos(p_{\theta t} + \Delta\theta) + \cos(p_{\theta t})) \\ \Delta\theta \end{pmatrix}. \quad (1)$$

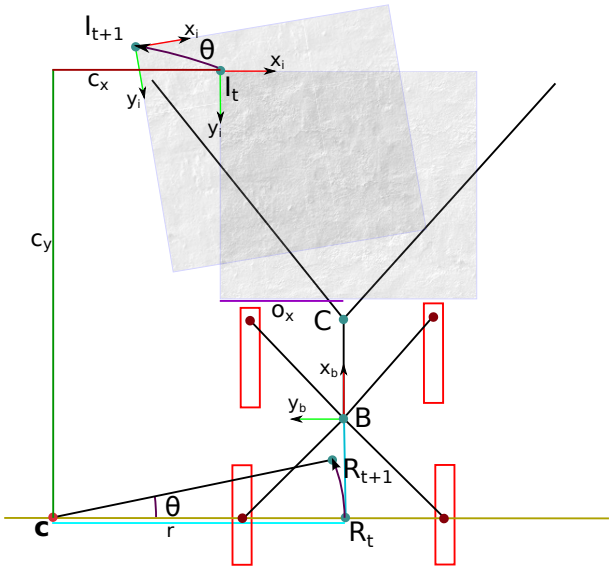

Fig. 3. Scheme of the differential drive model: B is the base link frame, R_t is the center of the rear axis at time t, $\mathbf{c}$ is the current center of rotation, C is the camera frame and I_t is the current image frame. The position of the rotation center in image coordinates is described by c_x and c_y. o_x is the known distance along the x-axis between the current image origin and the current rear axis center R_t.

In Fig. 3 two coordinate representations are used: The base link frame B is described in world coordinates with units $[m]$ and radians, all other frames are described in image coordinates with units pixels and radians. Therefore two conversion functions are required to convert world to image coordinates $q = f_{wi}(p)$ and vice versa $p = f_{iw}(q)$. f_{wi} is used for converting the fixed vehicle model parameters to image coordinates, $q = f_{wi}(p)$ is required to convert the image alignment results back to world coordinates.

C. Image Warp

To directly estimate the vehicle's motion parameters by minimizing the photometric error between two images, the image warp used for the optimization process has to be parametrised accordingly. Since the robot's motion is described by a rotation around a point on the rear axis, the same must hold for the images. The warping function is therefore parametrized with the rotation's center $c_x = f_{wi}(r) - o_x$ and angle $\theta = \Delta\theta$.

The rotation $\mathbf{R}_c$ around a point $\mathbf{c} = (c_x, c_y)$ is described by:

$$\begin{aligned} \mathbf{R}_c &= \mathbf{T}_c \mathbf{R}_\theta \mathbf{T}_c^{-1} \\ &= \begin{pmatrix} \cos(\theta) & -\sin(\theta) & (1-\cos(\theta))c_x - \sin(\theta)c_y \\ \sin(\theta) & \cos(\theta) & \sin(\theta)c_x + (1-\cos(\theta))c_y \\ 0 & 0 & 1 \end{pmatrix} \end{aligned} \quad (2)$$

where $\mathbf{T}_c$ is the translation matrix to the center and $\mathbf{R}_\theta$ is the rotation matrix around angle θ. Since the image coordinate system's z-axis is flipped, the rotational part is transposed. While c_y is known from the vehicle model, c_x and the rotation angle θ are free parameters. An image can be interpreted as function $I(\mathbf{i})$ of a point $\mathbf{i} = (i_x, i_y)$ that returns an intensity value, warping an image is equivalent to transforming the point positions by a function ω: $I(\omega(\mathbf{i}, \mathbf{m}))$

From [2] the function for warping an image point $\mathbf{i}$ by parameters $\mathbf{m} = (c_x, \theta)$ is:

$$\omega'(\mathbf{i}, \mathbf{m}) = \begin{pmatrix} \cos(\theta) i_x - \sin(\theta) i_y + (1 - \cos(\theta)) c_x - \sin(\theta) c_y \\ \sin(\theta) i_x + \cos(\theta) i_y + \sin(\theta) c_x + (1 - \cos(\theta)) c_y \end{pmatrix}. \quad (3)$$

Linearisation of $\omega'(\mathbf{i}, \mathbf{m})$ around $\theta = 0$ using a first order Taylor expansion gives:

$$\begin{aligned} \omega(\mathbf{i}, \mathbf{m}) &= \omega'(\mathbf{i}, \mathbf{m})|_{\theta=0} \\ &= \begin{pmatrix} i_x - i_y \theta - c_y \theta \\ i_x \theta + i_y + c_x \theta \end{pmatrix}. \end{aligned} \quad (4)$$

V. Image Alignment

Image alignment is done by minimizing the sum of squared differences between the previous image I_t and the current image I_{t+1} by finding the optimal warp parameters $\mathbf{m}$. The error function over all pixels is:

$$E(\mathbf{m}) = \sum_i (I_{t+1}(\mathbf{i}) - I_t(\omega(\mathbf{i}, \mathbf{m})))^2 \quad (5)$$

If the photo-consistency assumption holds, there are parameters $\mathbf{m}_{opt}$ for which $E(\mathbf{m}_{opt}) = 0$. $f_{iw}(\mathbf{m}_{opt})$ then describes the movement performed by the vehicle. To find the warp parameters $\mathbf{m}$ that minimize $E(\mathbf{m})$ between two consecutive images I_t and I_{t+1}, an iterative non-linear least squares method is used, as formulated in [15]:

$$\begin{aligned} \mathbf{m}_{n+1} =& \mathbf{m}_n - \mu_n (\mathbf{J}_n^T \mathbf{C}_D^{-1} \mathbf{J}_n + \mathbf{C}_M^{-1})^{-1} \\ & (\mathbf{J}_n^T \mathbf{C}_D^{-1} (I_{t+1} - I'_t)) + \mathbf{C}_M^{-1} (\mathbf{m}_n - \mathbf{m}_{prior})) \end{aligned} \quad (6)$$

where n is the current optimizer iteration, μ_n is the step width, $\mathbf{J}_n$ is the current Jacobian, $\mathbf{C}_D$ is the data covariance, $\mathbf{C}_M$ is the model covariance and $\mathbf{m}_{prior}$ is the model prior. The previous image I_t is transformed with the current parameter estimate $I'_t = I_t(\omega(\mathbf{i}, \mathbf{m}_n))$ after each iteration for updating the pixel-wise image differences $I_{t+1} - I'_t$ and the Jacobian $\mathbf{J}_n$.

Three termination criteria are used for the optimization: The maximum number of iterations $n > n_{max}$, a minimum error value $E(m_n) < \epsilon$ and a minimum step size $|m_{n+1} - m_n| < \delta$. Whenever one of these criteria is met, the optimizations stops.

In general the Jacobian $\mathbf{J}$ is the derivative of the warped image with regard to the model parameters, this derivative can be calculated by using the chain rule:

$$\mathbf{J} = \frac{\partial I(\omega(\mathbf{i}, \mathbf{m}))}{\partial \mathbf{m}} = \frac{\partial I(\mathbf{i})}{\partial \mathbf{i}} \frac{\partial \omega(\mathbf{i}, \mathbf{m})}{\partial \mathbf{m}} \quad (7)$$

The proposed method uses Efficient Second order Minimization (ESM) like formulation to calculate the Jacobian, see [16] and [5] for a detailed derivation of the ESM. ESM uses a Taylor expansion of the error function and the Jacobian to approximate the Hessian of the cost function, the ESM based Jacobian is:

$$\mathbf{J}_n = \frac{1}{2} \left(\frac{\partial I_{t+1}(\mathbf{i})}{\partial \mathbf{i}} + \frac{\partial I'_t(\mathbf{i})}{\partial \mathbf{i}} \right) \frac{\partial \omega(\mathbf{i}, \mathbf{m})}{\partial \mathbf{m}} \quad (8)$$

Image gradients can be obtained by using e.g. a Sobel operator:

$$\begin{aligned} \frac{\partial I'_t(\mathbf{i})}{\partial \mathbf{i}} &= \nabla I'_t = (\nabla_x I'_t, \nabla_y I'_t) \\ \frac{\partial I_{t+1}(\mathbf{i})}{\partial \mathbf{i}} &= \nabla I_{t+1} = (\nabla_x I_{t+1}, \nabla_y I_{t+1}) \end{aligned} \quad (9)$$

where $\nabla_x I$ and $\nabla_y I$ is the gradient in x resp. y direction. Deriving the warping function (4) with regard to the parameters $\mathbf{m}$:

$$\frac{\partial \omega(\mathbf{i}, \mathbf{m})}{\partial \mathbf{m}} = \begin{pmatrix} 0 & (-i_y - c_y) \\ \theta & (i_x + c_x) \end{pmatrix} \quad (10)$$

Plugging (9) and (10) into (8) results in the Jacobian for one pixel, which is one row of the $\mathbf{J}_n$ matrix:

$$(\nabla y \theta, \nabla x (-i_y - c_y) + \nabla y (i_x + c_x)) \quad (11)$$

with

$$\nabla x = \left(\frac{1}{2} (\nabla x I_{t+1} + \nabla_x I'_t \right) \quad (12)$$

$$\nabla y = \left(\frac{1}{2} (\nabla y I_{t+1} + \nabla_y I'_t \right) \quad (13)$$

The data covariance $\mathbf{C}_D$ in (6) is set to 1, all pixel intensities are independent and equally likely. The model covariance $\mathbf{C}_M$ has to be set appropriately to achieve fast and robust convergence of the optimisation. This is necessary because the ranges of the two model parameters differ by orders of magnitude: While c_x can have huge values, (in fact for a straight forward driving vehicle it is $\pm \inf$, θ has a typical range of $[-0.2, 0.2]$. During evaluation $\mathbf{C}_M$ was set to:

$$\mathbf{C}_M = \begin{pmatrix} 10^4 & 0 \\ 0 & 10^{-3} \end{pmatrix} \quad (14)$$

For the model prior $\mathbf{m}_{prior}$ the previously performed motion has shown to be a reasonable choice. If available, estimates from other sensors like wheel odometry or an IMU could also be used.

A. Outlier Removal

For visual odometry, outliers are image regions that do not reflect the actual motion of the camera. These must be excluded from the optimization process.

A popular way for outlier removal is the use of iteratively re-weighted least squares, as presented in [9] and [7]. After each optimizer iteration, a residual image is created and, based on these pixel wise residuals, a weight for each pixel is calculated. In the next optimizer iterations, these weights are multiplied with the corresponding Jacobian row to weight the pixel's influence on the optimization.

The proposed method uses an approach which takes into account the fact that outlier pixels usually have a spatial relation since effects like overexposure, reflections or moving objects are unlikely to appear only pixel wise.

The outlier rejection is performed after the alignment of the whole image terminates. The optimized parameters m_r are used as the initial estimate for the outlier rejection.

In order to find outlier regions, the image is split into blocks $\mathcal{B}$ of a fixed size, as shown in Fig. 4. For each block $\mathcal{B}_k$ the Jacobian $\mathbf{J}_k$ and the image difference $\mathbf{d}_k = (I_{t+1} - I'_t)$ of the contained pixels are calculated. With $\mathbf{J}_k$ and $\mathbf{d}_k$ for each block a single parameter update step $\mathbf{m}_k$ is estimated

$$\mathbf{m}_k = \mu_n(\mathbf{J}_k^T\mathbf{J}_k + \mathbf{C}_M^{-1})^{-1}(\mathbf{J}_k^T\mathbf{d}_k) + \mathbf{C}_M^{-1}(\mathbf{m}_r)). \quad (15)$$

Each $\mathbf{m}_k$ describes the direction in model space for which the blocks error $E(\mathbf{m}_k)$ would decrease. Without outliers and image noise, the values of all $\mathbf{m}_k$ should be similar. Blocks that contain a significant amount of outliers have a different gradient direction compared to blocks that correctly describe the vehicle's motion. Based on the estimated model parameters $\mathbf{m}_k$, a clustering in model space is performed by comparing the model space position of each block to all other blocks. For clustering a weighting function $W(v, \varepsilon)$ based on the Tukey weighting function [17] is used:

$$W(v,\varepsilon) = \begin{cases} 0 & \text{if } |v| > \varepsilon \\ (1 - (\frac{v}{\varepsilon})^2)^2 & \text{otherwise.} \end{cases} \quad (16)$$

Each model parameter c_x, θ has a separate weighting parameter $\varepsilon_x, \varepsilon_\theta$. The cluster weight of each block is:

$$\Phi(\mathbf{m}_k) = \sum_{j=0}^{k} W(m_{kx} - m_{jx}, \varepsilon_x) W(m_{k\theta} - m_{j\theta}, \varepsilon_\theta) \quad (17)$$

The parameters of the block with the highest $\Phi(\mathbf{m}_s)$ are selected as $\mathbf{m}_f$ and used as center for the cluster membership test. To get the new motion estimate, the weighted sums over the Jacobians and image differences are given by:

$$\mathbf{J}_f = \sum_{j=0}^{k} \mathbf{J}_j W(m_{fx} - m_{jx}, \varepsilon_x) W(m_{f\theta} - m_{j\theta}, \varepsilon_x) \quad (18)$$

and

$$\mathbf{d}_f = \sum_{j=0}^{k} \mathbf{d}_j W(m_{fx} - m_{jx}, \varepsilon_x) W(m_{f\theta} - m_{j\theta}, \varepsilon_x). \quad (19)$$

From $\mathbf{J}_f$ and $\mathbf{d}_f$ a new estimate for $\mathbf{m}_r$ is calculated as described in (15). The process can be repeated several times. This might be required if the initial estimate was too far from the optimum.

VI. Evaluation

A. Compared Methods

The proposed method was compared against five publicly available visual odometry or SLAM systems and its predecessor:

1) DoF3OR, the predecessor of the proposed method [14] without kinematic model constraints.

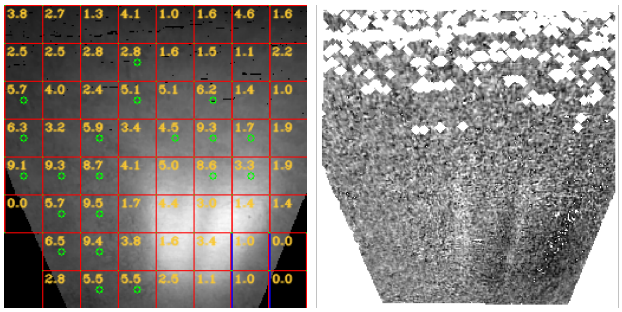

Fig. 4. Left: Visualization of the blocks used for outlier rejection, the number is $\phi(\mathbf{m}_k)$, green dots mark blocks that contribute to the selected cluster. Right: Residual image after performing the global image alignment. The white spots in the upper third of the image are pixels that were removed due to their proximity to invalid pixels.

2) DVO, a dense 6DoF visual odometry [9].
3) EDVO, a semi-dense 6DoF direct visual odometry method [7].
4) RTAB, a versatile feature-based SLAM system that includes loop closure and pose graph optimization [3]. The method was used with 3DoF localisation and non-holonomic constraints for motion estimation.
5) ORBSLAM2, a feature-based 6DoF SLAM system [2]. Two modes were tested: The full SLAM system (ORBSLAM) and the visual odometry mode with mapping disabled (ORBLOC).
6) DEMO, a feature based 6DoF visual odometry [20].

B. Data Acquisition

To compare the performance of the proposed method with other approaches, nine sequences in different indoor environments were recorded. Fig. 8 shows one of these sequences with ground truth trajectory and the resulting trajectories of all compared visual odometry systems. The total length of these nine trajectories is 527m and the total recorded time 17min. They were selected to cover different surface materials under different lighting conditions, see Fig. 5.

Fig. 5. Three different surface types were recorded: Tiles, wood and PVC (shown in Fig. 2). The images depict challenging situations the proposed method can handle: low light conditions and reflections, shadows and non planar environment.

The ground truth trajectories were created using a Sick TiM561 LIDAR for data acquisition and the Hector SLAM system [18] to integrate the LIDAR data into a global occupancy map with $0.05m$ resolution used for localisation. Every ground truth sequence was checked for map and

trajectory inconsistencies and discarded if any were found, leaving 9 out of 18 originally recorded sequences.

Each tested method was run on all nine sequences, recording the pose estimate after each frame. This yields a set $\mathcal{F}$ of poses for each method and each sequence.

C. Evaluation Method

For performance evaluation, the visual odometry evaluation method described in [19] was used. This evaluation method extracts sub paths of different lengths and calculates two error measures individually for each sub path: the rotation error and the translation error. Each error measure is normalized by the path length to be able to compare the results of sub paths of different lengths. The error measures, as given in [19], are:

$$E_{rot}(\mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{(i,j \in \mathcal{F})} \angle[(\hat{p}_j \ominus \hat{p}_i]) \ominus (p_j \ominus p_i)] \quad (20)$$

$$E_{trans}(\mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{(i,j \in \mathcal{F})} ||(\hat{p}_j \ominus \hat{p}_i]) \ominus (p_j \ominus p_i)||_2 \quad (21)$$

where i and j are the start and end indices of a subset of $\mathcal{F}$ for a given length l_p, $\hat{p}$ and p are the estimated resp. ground truth poses, $\ominus$ is the inverse compositional operator and $\angle$ denotes the rotation angle.

The path lengths for evaluation were l_p=(1m, 2m, 5m, 10m, 15m, 20m, 25m, 30m, 35m, 40m). For each path length, a sub path was extracted at every tenth ground truth frame.

D. Results

The mean translation error per path length over all sequences is shown in Fig. 6, the mean rotation error in Fig. 7. Table I shows the mean error weighted by the number of sub paths per path length. One major source of error are rotations, as seen in Fig. 8. Only the proposed method provides a proper estimate of all rotations contained in this sequence. Another problem are reflections, causing most methods to underestimate the distance travelled. This can be observed when turning to the right after approximately 16m.

Two methods are not suited to work with the provided data: DEMO has problems establishing correct feature correspondences. For EDVO the reason is not as clear, it may be caused by the information selection scheme which selects data with strongest Jacobians. In environments with low contrast texture, image noise and reflections tend to give larger gradients than the floor itself. Regardless, as errors in setting up theses systems cannot entirely be ruled out these results should be regarded with care.

ORBSLAM provides good results as long as the tracking remains intact. In cases where loop closure could be applied, it outperformed most other methods. When the tracking is lost, no further pose estimates are provided, causing the translation and rotation error to rise to maximum for the remaining trajectory. This happened in four out of nine sequences e.g. Fig. 8 and was caused either by fast rotation or heavy motion blur. To be comparable, it was required to use the visual odometry only version ORBLOC. The results of ORBSLAM are only shown for completeness.

As shown in Fig. 7, DVO provides reasonable rotation estimates for most sequences, but tends to underestimate the travelled distance. Several sequences contain reflections that cover significant parts of the image. As described in [9], this high outlier ratio can cause a drift.

RTAB was configured to perform 3DoF SLAM and use non-holonomic constraints, i.e. not allowing strife motions. Although the approach is lacking some features of ORB-SLAM, like local bundle adjustment, the 3DoF SLAM combined with model based constraints results in better performance on the given data.

DoF3OR without non-holonomic constraints gives results comparable to RTAB. The basic structure, performing image alignment on the orthogonal projection of the sensor data and performing block-wise outlier rejection, is similar to the proposed method.

KMVO performs best across all tests, especially rotation estimation accuracy could be improved compared to its predecessor and it is over two times better than RTAB, which also uses 3DoF and model based constraints.

Over all evaluated sequences the average processing time of the proposed method was 15.2ms per frame on a single thread of an Intel i5 4300U Mobile CPU with 1.9GHz. This makes the proposed method suitable for real time applications on a wide range of mobile robot hardware.

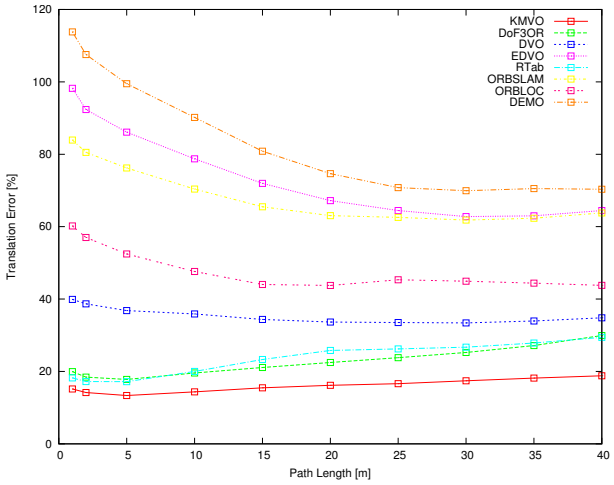

Fig. 6. Mean translation error by path length.

TABLE I

MEAN TRANSLATION AND ROTATION ERROR OVER ALL SUB PATHS.

Method	Trans (%)	Rot ($^\circ/m$)
KMVO	**15.48**	**1.69**
DoF3OR	21.44	2.80
DVO	36.00	4.10
EDVO	78.40	15.66
RTab	21.83	3.67
ORBSLAM	71.10	8.97
ORBLOC	49.80	7.20
DEMO	89.39	22.03

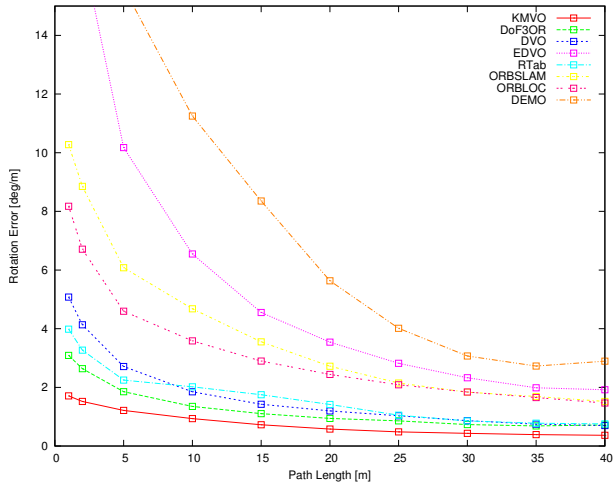

Fig. 7. Mean rotation error by path length.

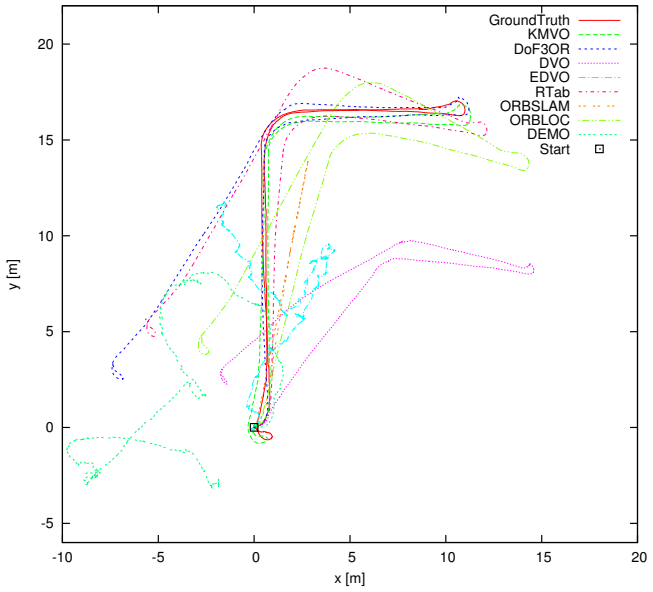

Fig. 8. Example Trajectory recorded in the environment shown as left image in Fig. 2.

VII. Conclusion

This work presented KMVO, a method for estimating motion parameters of a differential drive vehicle directly from ground plane images. The reduction from three to two parameters in the optimization process significantly improves the localization accuracy, especially with regard to the rotational part. Furthermore, an outlier rejection scheme was presented that can handle large outlier areas and is computationally efficient. The complete system was evaluated on nine real world data sets and compared to six other methods. In the tested scenarios, it outperformed all six methods it was compared to. It is also shown that the system is fast enough to run in real time on a single thread of a mobile CPU. Future work includes porting the system to monocular cameras and the inclusion of a pose graph optimization to further improve the accuracy.

Including the kinematic model into the visual odometry system is not constrained to differential drive vehicles. It can be similarly applied to other non-holonomic vehicles with two degrees of freedom, e.g. Ackermann steered vehicles.

ACKNOWLEDGMENT

This work was supported by the BMBF project MobilAssist under grant number 01IS15049A.

References

[1] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2564–2571.

[2] R. Mur-Artal and J. D. Tardos, "Orb-slam2: an open-source slam system for monocular, stereo and rgb-d cameras," *arXiv preprint arXiv:1610.06475*, 2016.

[3] M. Labbe and F. Michaud, "Online global loop closure detection for large-scale multi-session graph-based slam," in *Intelligent Robots and Systems (IROS), 2014*, Sept 2014, pp. 2661–2666.

[4] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," 1981, pp. 674–679.

[5] E. Malis, "Improving vision-based control using efficient second-order minimization techniques," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 2. IEEE, 2004, pp. 1843–1848.

[6] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[7] S. Klose, P. Heise, and A. Knoll, "Efficient compositional approaches for real-time robust direct visual odometry from rgb-d data," in *Intelligent Robots and Systems (IROS), 2013*. IEEE, 2013, pp. 1100–1106.

[8] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," in *arXiv:1607.02565*, July 2016.

[9] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for rgb-d cameras," in *Robotics and Automation (ICRA), 2013*. IEEE, 2013, pp. 3748–3754.

[10] J. Zienkiewicz and A. Davison, "Extrinsics autocalibration for dense planar visual odometry," *Journal of Field Robotics*, pp. 803–825, 2014.

[11] S. Lovegrove, A. Davison, and J. Ibanez-Guzman, "Accurate visual odometry from a rear parking camera." IEEE, 2011, pp. 788–793.

[12] B. M. Kitt, J. Rehder, A. D. Chambers, M. Schonbein, H. Lategahn, and S. Singh, "Monocular visual odometry using a planar road model to solve scale ambiguity," in *Proc. European Conference on Mobile Robots*, September 2011.

[13] D. Scaramuzza, F. Fraundorfer, and R. Siegwart, "Real-time monocular visual odometry for on-road vehicles with 1-point ransac," in *Robotics and Automation (ICRA), 2009.* IEEE, 2009, pp. 4293–4299.

[14] J. Jordan and A. Zell, "Ground plane based visual odometry for rgbd-cameras using orthogonal projection," *IFAC-PapersOnLine*, vol. 49, no. 15, pp. 108–113, 2016.

[15] A. Tarantola, *Inverse problem theory and methods for model parameter estimation*. SIAM, 2005.

[16] S. Benhimane and E. Malis, "Real-time image-based tracking of planes using efficient second-order minimization," in *Intelligent Robots and Systems, 2004. (IROS).*, vol. 1, Sept 2004, pp. 943–948.

[17] P. Huber, *Robust Statistical Procedures*. Society for Industrial and Applied Mathematics, 1996.

[18] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011.

[19] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR), 2012*, 2012.

[20] J. Zhang, M. Kaess, and S. Singh, "Real-time depth enhanced monocular odometry," in *Intelligent Robots and Systems (IROS), 2014*. IEEE, 2014, pp. 4973–4980.