2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Robust Stereo Visual Odometry Using Iterative Closest Multiple Lines

Jonas Witt and Uwe Weltin

***Abstract*— This work is concerned with the matching of straight lines between two stereo image pairs by reprojection. While we will focus on visual odometry in the realm of simultaneous mapping and localization, the techniques are also relevant to monocular and stereo 3D object detection and tracking. Our first contribution is an adaptation of the Iterative Closest Point (ICP) algorithm to the domain of lines. We argue that a naive "Iterative Closest Line" derivation cannot deliver similar performance. In contrast, our novel Iterative Closest Multiple Lines (ICML) algorithm allows efficient line matching while even reducing the amount of local minima during iterative optimization with its consideration of several weighted matches. The second contribution is a fast and robust hypothesize-and-test algorithm which can act as a fallback for challenging frame pairs where pure gradient-based optimization fails. In several differently textured scenes, we demonstrate robust performance, even in very sparse cases where proven feature point based methods fail. In comparison to edge-point ICP, we see speed improvements of more than an order of a magnitude and reduced susceptibility for local minima.**

I. Introduction

Visual navigation for mobile robots is an intensively researched topic which can be arbitrarily complex. Ideally, a system should be able to recognize features and places independent of environmental influences. Dramatic changes in appearance due to lighting, partial scene reconfiguration or even seasonal changes in the environment are challenging in this respect. In addition, a system must be able to cope with dynamic objects in its field of view and work equally well in textured and untextured environments. However, to the knowledge of the authors, such a complete solution has not been proposed yet.

Many real-time capable visual navigation systems rely on point features. They have favorable properties like being easily detectable, locatable and matchable. In many environments, these systems function fast and robustly. However, reliable point features are not always available in sufficient number. Untextured three-dimensional objects and environments pose problems to these kind of systems. Edges on the other hand are interesting image features for a host of robotic applications where textures are sparse. Many man-made structures lack texture while edges are usually still abundantly available (see Figure 1). In [1], it was exemplarily shown how an edge-based system can outperform a feature point-based algorithm like the popular KLT tracker [2]. We will go one step further and provide quantitative results of the state-of-the-art stereo odometry algorithm VISO2 [3] for all our test sequences. In the KITTI benchmark [4], this method has proven to be very accurate and one of the fastest feature point based odometry algorithms.

J. Witt and U. Weltin are with the Institute for Reliability Engineering, Hamburg University of Technology, 21073 Hamburg, Germany, email: (jonas.witt, weltin)@tuhh.de

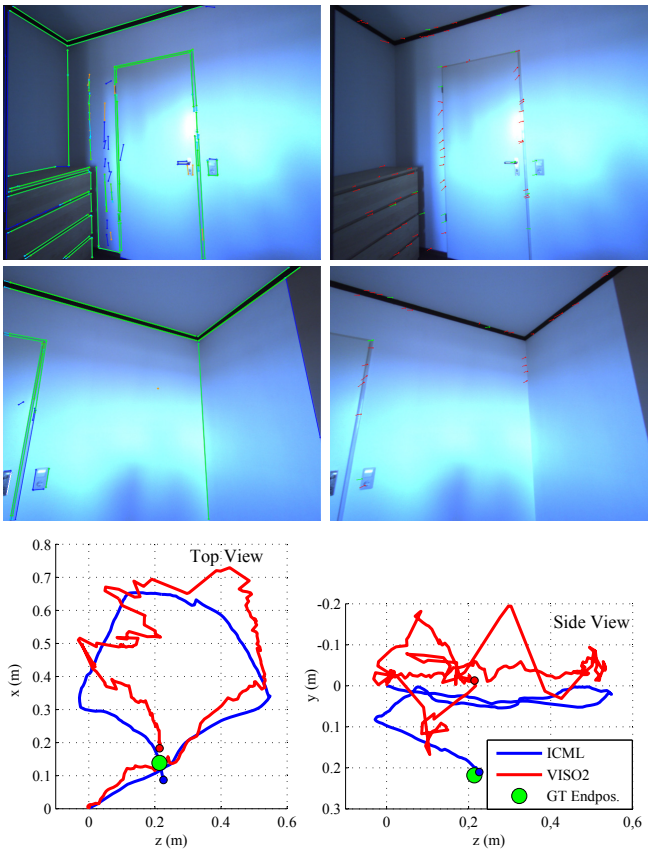


Fig. 1. Non-textured scenes can be a challenge for feature point based SLAM systems. The images show frames of an indoor Room sequence for which ICML (left column) successfully computes odometry, while the state-of-the-art feature point based VISO2 [3] (right column) fails to recover the camera motion in many frames. Even the matched feature points (green) are often inaccurate, when they are not located on corners. The result is a zig-zag trajectory with large jumps. ICML achieves an accuracy of 2.5% on this short trajectory of 2.1m length (note the video link on the last page).

Concerning the use of edges in vision-based navigation, inherent difficulties exist. By definition, edges divide regions of homogeneous intensity. Accordingly, their distinctiveness is limited. Finding an individual edge correspondence between two images without known camera motion is not simple and sometimes not even possible in a reliable manner [5]. However, with a calibrated stereo camera, within a stereo image set, this task is a lot simpler. This is because stereo matching searches a constrained solution space. With prior stereo matching, in order to match edges between two distinct stereo camera locations, we are left with the

problem of finding the rigid transformation that aligns the largest number of lines between the two stereo frames. This can make an appearance-based inter-frame edge matching unnecessary.

Finding the six motion parameters between two stereo edge frames is still challenging, since the search space contains local minima. This registration problem can be approached with the well-known Iterative Closest Point (ICP) algorithm in a 2D-3D variant [1], [6]. The possibility of registration failures requires a robust fallback method, though. While ICP has been shown to work, point sequences are not a very compact representation for intensity edges, especially when it is expected that many edges are partially or completely straight. This leads to significantly higher computation times than for most feature point based methods. For example, a square is fully described with its four corners (which are good feature points) or the lines that make up the contour. On the other hand, the number of edge points in image space is much larger, while not necessarily carrying additional information.

The deduction of an "Iterative Closest Line" algorithm is not as straight forward as one might think at first, though. In contrast to points, for lines, no definitive scalar metric for closeness exists. Additionally, the detection of line endpoints is usually unreliable. An edge that was detected as one long line in a given image might be split into several shorter segments in another image. Thus, a 1-to-1 matching would have to discard all of those segments but one.

We propose a method called Iterative Closest Multiple Lines (ICML) to efficiently register lines with a one-to-many matching. This remedies the problems associated with the selection of a single best matching line. The registration rate even improves over the much more costly ICP. For robustness, we use measures for automatic registration failure detection and propose a sample consensus based solution as fallback. In contrast to RANSAC, the hypothesis selection is deterministic to make efficient use of computational resources.

II. Related Work

Several monocular SLAM systems that utilize edge segments as features have been proposed in recent years [7], [8], [9]. They try to establish an appearance-based matching between frames. However, only a subset of all possible edges is considered to allow for successful matching. In very sparse environments, this can lead to problems.

In [6], known three-dimensional models are matched to edges that are detected in monocular images. The ICP algorithm is employed. It iteratively aligns the reprojected model edges with the detected image edges.

An extension of this technique to 3D SLAM using stereo edge points was presented in [1]. Here, edge points are first matched within the stereo frame and afterwards the rigid transformation between two frames is iteratively computed by optimizing the reprojection error. In distinction to this work, the author uses individual edge points and employs a variant of the ICP algorithm to align the stereo frames. This direct method can suffer from local minima during the optimization, which is why the author introduces SIFT (Scale-Invariant Feature Transform) descriptors along edge points in [10] for failure recovery.

A system using straight lines was proposed in [11]. Dense stereo matching for intra-frame and multi-level Lucas-Kanade optical flow for inter-frame matching are employed. With the significantly reduced search space, RANSAC is used to find the best transformation by computing a rigid transformation hypothesis, built from two or three matched lines. Accordingly, the line matching and motion recovery are separate steps in this approach. A consequence of this appearance-based matching is that the success depends on the performance of the optical flow algorithm.

III. Stereo Reconstruction

For stereo edge matching, we employ an adapted version of [12]. Briefly, edges are detected with a Canny detector [13] in the left and right image of a stereo frame. Subsequently, straight segments are extracted with the Douglas-Peucker algorithm [14] and lines are fitted for each. Slightly curved edges are included as multiple straight lines and are not left out. All found lines are binned, depending on their angle and sorted from left to right within each angular line group (per image). Finally, dynamic programming finds the matches based on pixel support regions and overlap. Note that the line order of the sorted lines can not be reversed during dynamic programming. Horizontal lines can not easily be matched and are not included in this step. They are recovered during post-processing, when horizontal line disparities are interpolated from adjacent diagonal and vertical matched lines.

Although the more general edge matching technique in [15] finds even more correct matches and works very well, the line matching is usually about 5-10 times faster. The technique proposed in [16] was also considered for stereo edge matching, but for horizontal edges the matching rates are poor. This was found to be problematic in many instances, since the environments that we are interested in mainly consist of long and featureless vertical and horizontal edges.

We represent lines by their endpoints, which is convenient for reconstruction and reprojection. The 3D reconstruction of one point requires two matched image points (u_l, v_l) and (u_r, v_r). The vertical coordinate $v = v_l = v_r$ is the same for both image points, since we only consider rectified images where matches lie on the same scanline. The reconstruction in camera coordinates is obtained by

$$\mathbf{c} = \left(\frac{bu_l}{u_l - u_r}, \frac{bv}{u_l - u_r}, \frac{bf}{u_l - u_r} \right)^T \tag{1}$$

with known intrinsic stereo camera parameters (we assume subtracted principal points). The baseline is denoted with b and f corresponds to the focal length. The difference in horizontal point coordinates $u_l - u_r$ is the disparity, corresponding to depth. World point coordinates $\mathbf{p}$ are computed with the camera rotation $\mathbf{R}$ and translation $\mathbf{t}$.

$$\mathbf{p} = \mathbf{R}^{-1}\mathbf{c} + \mathbf{t} \tag{2}$$

IV. Motion Reconstruction Using Lines

Motion reconstruction is achieved through registration of the previously reconstructed three-dimensional lines with the currently detected (yet two-dimensional) image lines. Although it would be possible to do 3D-3D matching by reconstructing the current image lines prior to registration, a naive approach would introduce unnecessary errors as lines with large Euclidean errors would dominate the optimization (i.e. lines that are far away). Optimizing the reprojection error (3D-2D optimization) naturally accounts for the type of uncertainty that is inherent to image space measurements. In addition, the registration of monocular imagery with a 3D model is equally possible, as done in [6]. The minimum number of line matches is three in this case, while stereo matching requires only two nonparallel lines to recover the 6-DOF motion.

A. Line Reprojection Error

The reprojection for 3D points is defined by the intrinsic and extrinsic camera parameters. While the rotation matrix $\mathbf{R}_k$ and the translation vector $\mathbf{t}_k$ make up the unknown six degrees of freedom of the camera that we would like to recover at frame k, the intrinsic parameters, again, are known from prior calibration. First we transform from world to camera coordinates, then we project to the image plane.

$$\mathbf{c} = \mathbf{R}_k(\mathbf{p} - \mathbf{t}_k) \tag{3}$$

$$(u, v)^T = \left(\frac{fc_x}{c_z}, \frac{fc_y}{c_z} \right)^T \tag{4}$$

While it is rather straightforward to define the reprojection error between two points (or a point and a line) as their minimum Euclidean distance, such a clear statement for the error between two lines is not possible. This stems from the fact that the error between 2D lines is two-dimensional (if endpoint locations are not considered). One possibility to parameterize this error is by representing one of the lines by its end points and computing the perpendicular distance to the other line for each.

We characterize the match between the j^{th} reprojected line $\mathbf{l}^j(\mathbf{x})$ and the i^{th} image line $\hat{\mathbf{l}}^i$ by their overlap $l(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)$ and mean distance $d_{ij}(\mathbf{x})$ in image space.

$$d_{ij}(\mathbf{x}) = 0.5 \left(|d_{P_1}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)| + |d_{P_2}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)| \right) \tag{5}$$

The terms $d_{P_1}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)$ and $d_{P_2}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)$ denote the perpendicular distances of the end points of image line $\hat{\mathbf{l}}^i$ to the reprojected line $\mathbf{l}^j(\mathbf{x})$. We will omit the dependency of the overlap $l(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)$ on the transformation $\mathbf{x}$ by writing l_{ij} to convey that this value is kept as a weighting constant instead of participating in the optimization (although it is updated in between iterations). The product of $d_{ij}(\mathbf{x})$ and l_{ij} equals the overlapping area between the two lines. Accordingly, the mean pixel error (ME) for all matching lines can be formulated in the following way:

$$\mathrm{ME}(\mathbf{x}) = \frac{\sum_{i,j} d_{ij}(\mathbf{x}) l_{ij}}{\sum_{i,j} l_{ij}} \tag{6}$$

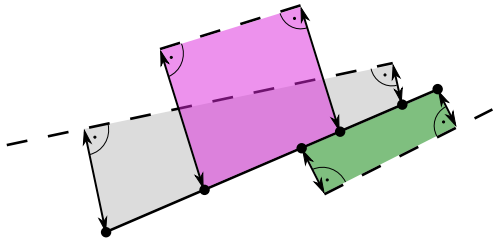


Fig. 2. Multiple reprojected lines (dashed) near a line in the current frame (solid). Which one is the "best" match? Since the matching error is multidimensional, a scalar "best" metric as for the distance between points does not exist.

Another natural choice would be the weighted mean squared error (MSE):

$$\mathrm{MSE}(\mathbf{x}) = 0.5 \frac{\sum_{i,j} \left(d_{P_1}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)^2 + d_{P_2}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i)^2 \right) l_{ij}}{\sum_{i,j} l_{ij}} \tag{7}$$

The weighting with the overlap l_{ij} is introduced to reflect the significance of a match in the cost function. Accordingly, a number of small matching lines have the same influence on the result as one larger match with equal cumulative overlap.

While the MSE is a common and usually sound choice in optimization problems, the mean pixel error includes absolute value functions in $d_{ij}(\mathbf{x})$, which are problematic for optimization. However, if the absolute value terms are replaced by Huber functions $h(z)$, the equation becomes differentiable while keeping its linear behavior for larger errors.

$$h(z) = \begin{cases} 0.5z^2 & |z| < k \\ k|z| - 0.5k^2 & else \end{cases} \tag{8}$$

Using values like $k = 1\mathrm{px}$ limits the region with quadratic behavior to small errors.

B. Iterative Closest Multiple Lines (ICML) Algorithm

The Iterative Closest Points (ICP) algorithm [17] functions by finding the minimum distance to a model (which would be lines or edge point sequences in this case) for each point and improving this distance with gradient-based optimization. This is repeated until convergence. The publications [6] and [1] use this approach for the monocular and stereo case, respectively.

Due to the two-dimensional reprojection error it is not clear what the closest line to another line is, as shown in Figure 2. Of course we can define a matching score and utilize e.g. the mean pixel distance and overlap between two lines as a criterion, but we will see why this is unfavorable.

Instead of finding a "best" match among the model lines (1-to-1 matching), the ICML algorithm considers all lines in the neighborhood (1-to-N matching). For a line to be considered in the neighborhood it has to fulfill the following criteria: $d_{ij} < d_{\max}$, $l_{ij} > 0$ and $\alpha_{ij} < \alpha_{\max}$. The term α_{ij} refers to the angle between both lines in the image plane,

while $\alpha_{\max}$ and $d_{\max}$ are free parameters. During all stages we use $\alpha_{\max} = 12°$.

The resulting optimization problem has two rows for each line match per view. For stereo optimization, the matches in the right image are added analogously. With the weighting matrix $\mathbf{W} = diag(l_{ij})$ we minimize the following quadratic cost function in the case of MSE:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{e}(\mathbf{x})^T\mathbf{W}\mathbf{e}(\mathbf{x}) \tag{9}$$

$$\mathbf{e}(\mathbf{x}) = \left(..., \; d_{P_1}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i), \; d_{P_2}(\mathbf{l}^j(\mathbf{x}), \hat{\mathbf{l}}^i), \; ...\right)^T \tag{10}$$

To minimize the Huber ME we use $h(\mathbf{z})$ as an element-wise Huber function and yield:

$$f_h(\mathbf{x}) = \frac{1}{2}\mathbf{W}\mathbf{e}_h(\mathbf{x}) \tag{11}$$

$$\mathbf{e}_h(\mathbf{x}) = h(\mathbf{e}(\mathbf{x})) \tag{12}$$

These functions can finally be minimized with Levenberg-Marquardt optimization when the incremental motion update $\mathbf{x}$ and the error Jacobian $\mathbf{J}$ are defined:

$$\mathbf{x} = (t_x, t_y, t_z, \phi_x, \phi_y, \phi_z)^T \tag{13}$$

$$\mathbf{J} = \frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}} \tag{14}$$

Note, that $f(\mathbf{x})$ and $f_h(\mathbf{x})$ are scaled versions of Eq. 6 and 7 respectively. Since the denominator with the cumulative overlap is constant during one iteration, we omit it for simplicity.

The solution vector $\mathbf{x}$ constitutes the incremental transformation that we want to calculate. Besides a translation vector $\Delta\mathbf{t} = (t_x, t_y, t_z)^T$ it contains an incremental rotation which relates to the following linearized rotation matrix.

$$\Delta\mathbf{R} = \begin{bmatrix} 1 & -\phi_z & \phi_y \\ \phi_z & 1 & -\phi_x \\ -\phi_y & \phi_x & 1 \end{bmatrix} \tag{15}$$

The pose update equations from frame $k-1$ to k are

$$\mathbf{R}_k = \Delta\mathbf{R}\mathbf{R}_{k-1}\,, \tag{16}$$

$$\mathbf{t}_k = \mathbf{t}_{k-1} - (\Delta\mathbf{R}\mathbf{R}_{k-1})^{-1}\Delta\mathbf{t}\,. \tag{17}$$

While for a given $d_{\max}$, the scheme of alternate matching and optimization is similar to regular ICP, we overlay another loop to control this matching parameter and implement a coarse-to-fine strategy. We iteratively reduce $d_{\max}$ from a large value d_{init} down to a fine value d_{final} to achieve convergence from a wider range of configurations while not sacrificing accuracy when the lines are correctly aligned. While d_{init} and d_{final} are both free parameters, we kept $d_{\text{final}} = 1\text{px}$ during all experiments. The parameter d_{init} should be chosen large enough, so that the correctly matching lines are among the matches. The algorithm is listed in the following:

1) Set $d_{\max} = d_{\text{init}}$.
2) For each image line, find all model lines with $d_{ij} < d_{\max}$, $l_{ij} > 0$ and $\alpha_{ij} < \alpha_{\max}$.
3) With all found matches, do Levenberg-Marquardt until convergence.
4) If $d_{\max} \leq d_{\text{final}}$ exit, otherwise $d_{\max} = d_{\max}/2$ and go to step 2.

Optionally, one can add an offset in the order of $d_{\max}$ to each l_{ij} to allow lines to influence the optimization that do not actually overlap, but are within the axial range of this offset. Accordingly, we need to count the axial gap between lines as negative overlap. This can be of importance for fast simultaneous movements in horizontal and vertical image direction when no large lines are detected. However, in our trials this was not necessary.

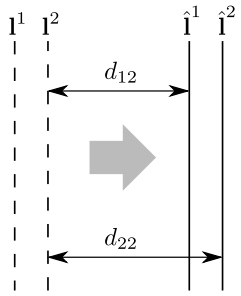
Due to the potentially large number of matched lines (especially for sizeable d_{init}), ICML can be biased towards one-sided groupings of parallel lines within match range. A typical configuration of this kind can be found when looking down corridors (see top left image of Figure 4). To remedy this affinity, we sort all matches of each line by their mean distance d_{ij}. Subsequently, the matches are removed starting from the largest d_{ij} until the cumulative overlap of the remaining matched lines is smaller than $\mu_{\text{OM}}|\hat{\mathbf{l}}^i|$. While $|\hat{\mathbf{l}}^i|$ is the length of the image line that is matched, μ_{OM} is a tuning parameter that controls the amount of "overmatching". For example, when $\mu_{\text{OM}} = 3$ the cumulative overlap of all matched lines is restricted to three times the image line length. Accordingly, three lines would be matched if the overlap was 100% each. In the general case, a number $N \geq 3$ is matched for $\mu_{\text{OM}} = 3$ if enough lines are found in the $d_{\max}$ neighborhood. The introduction of sorting usually has no measurable influence on the optimization time.

C. Evaluation of Common Configurations

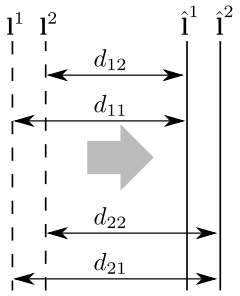
This section will investigate the behavior of ICP and ICML in common configurations. In the presented cases with infinite parallel lines, ICP and ICML with 1-to-1 matching (i.e. $\mu_{\text{OM}} = 0$) are equivalent. We will refer to this special case of ICML where only the line with the smallest d_{ij} is matched as Iterative Closest Line (ICL). For simplicity, we look at one dimension of the optimization problem, which is the horizontal displacement in this case.

Figure 3 shows a general configuration with parallel lines for which both 1-to-1 and 1-to-N matching are applied. One can see, that 1-to-1 matching is asymmetrical. So, if we iterated over the reprojected lines to find the best matches instead, we would get a different cost function. With 1-to-N matching, this becomes symmetrical (for $\mu_{\text{OM}} \to \infty$), since all mutual distances are included.

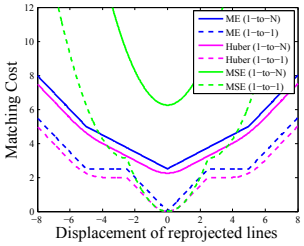
1) Matching with Successful Line Detection: We will first review the ideal case, when all lines have successfully been detected. The corresponding cost function is shown in Figure 3(c). First, we find that all cost functions are strictly monotonic decreasing within a displacement of less than half the line spacing. While with 1-to-N matching, the function stays strictly monotonic, with 1-to-1 matching we can locate plateaus on either side of the global minimum for ME and Huber. This section corresponds to the displacement



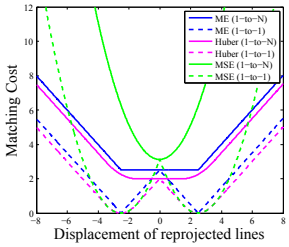
(a) The closest line is matched, which is equivalent to ICP/ICL for infinite parallel lines (1-to-1 matching).



(b) All close lines are matched, resulting in a symmetrical matching (1-to-N matching).



(c) Cost function comparison when all lines were successfully detected.



(d) Cost function comparison for the case when $\hat{\mathbf{l}}^2$ is not detected.

Fig. 3. These figures depict the 1-to-1 and 1-to-N matching methods. The dashed lines in (a) and (b) illustrate reprojected lines, while the solid lines are image lines of the current frame. The lines are parallel and have a horizontal spacing of five units. In (c) and (d), the mean error (ME), the Huber mean error and the mean squared error (MSE) are given for two scenarios. One can see that 1-to-N matching (solid lines) has fewer local minima and plateaus for these common cases.

for which one reprojected line is matched to both image lines and the distances have opposite signs. If no other edge matches can drag the optimization over such a plateau, the optimization is stuck. So for the optimization to converge, the maximal tolerable displacement is half the line spacing (both in image space). In many practical situations, this does not hinder optimization, though.

2) Matching with Partial Line Detection: When lines are only detected partially, the cost function changes. Consider the same configuration, except that $\hat{\mathbf{l}}^2$ is not detected. The resulting cost functions are plotted in Figure 3(d). With 1-to-1 matching we see two minima. This actively hinders successful optimization even if other lines have been matched correctly. In the case of 1-to-N matching on the other hand, a plateau permits effortless transition between both possibilities in this ambiguous setup for ME and Huber. Of course, other matches are needed to drive the optimization in the right direction, but even a correct line match between small line segments can provide the correct direction. When $d_{\max}$ is iteratively reduced, the line is finally only associated with one of the reprojected lines. When this is the case, the cost function has a unique minimum which supports an accurate alignment. In some scenarios, the MSE can be problematic for this case, because it tries to push the solution to the middle of both matches. However, in the sequences that we tested, line detection was reliable so that those situations were rare. When the line detector or the stereo matching algorithm perform unreliably, this has been found to make a difference.

D. Registration Failure Detection

Although optimizations most often converge to the global minimum with 1-to-N matching, as with any gradient based technique, local minima and other failures cannot be ruled out. In order for other more robust and costly techniques to take over, a measure for failure detection is required. In [10], the ratio of matched and detected edge points is used for failure detection. We use a similar measure by taking the ratio of the cumulative matched line length that was calculated with $d_{\max} = d_{\text{final}}$ and the minimum of the cumulative lengths of all detected image lines and reprojected lines. This ratio must be larger than $M_{\min}$.

$$M_k = \frac{\sum_{i,j} l_{ij}}{\min\left(\sum_i |\hat{\mathbf{l}}^i|, \sum_j |\mathbf{l}^j(\mathbf{x})|\right)} > M_{\min} \qquad (18)$$

We further test whether the mean error $\mathrm{ME}(\mathbf{x})$ exceeds the threshold $G_{\max}$. However, sometimes these criteria alone are not a sufficient indicator whether we can trust the solution. We found it necessary to include a measure which indicates whether the solution is stable. Imagine a scene with mostly parallel lines. In such a case, we can find a transformation which easily fulfills both of the other criteria. However, this solution is not unique. No reliable line match constrains the camera motion to not slide in the direction of those parallel lines. The measure that we propose here is an indicator for the stability of the solution. For this, we determine the directional cumulative matched line lengths l_h, l_{diag1}, l_{diag2} and l_v for horizontal ($|\alpha_i| < 22.5°$), diagonal ($22.5° \leq \alpha_i < 67.5°$ and $-22.5° \geq \alpha_i > -67.5°$) and vertical ($|\alpha_i| \geq 67.5°$) lines to gain knowledge of how the line directions are distributed. The angle α_i is the line orientation in image space. If three of these lengths are small we have an unstable solution. We desire a minimum diversity in line orientations to trust the solution. For this, l_h, l_{diag1}, l_{diag2} and l_v are sorted and assigned to $l_1 \leq l_2 \leq l_3 \leq l_4$. The stability criterion that we impose is the following.

$$L_k = l_1 + l_2 + l_3 > L_{\min} \qquad (19)$$

Accordingly, the sum of the lesser three directional cumulative line lengths must be at least $L_{\min}$ long. This does not constrain the ratio of the lengths, but ensures an absolute minimum orientation diversity among the matched lines.

E. Robust Sample Consensus Matching

When a registration failure was detected, a more robust algorithm is needed to circumvent a possible local minimum. In contrast to edge point matching, with lines, the geometry information is significantly more compact which makes a hypothesize-and-test scheme tractable. Since we are not building a map at this point, we are interested in a fallback solution for incremental motion that efficiently provides good starting values for a subsequent refinement with ICML.

Instead of assigning line match hypotheses purely random (like RANSAC), for incremental motion we can restrict the matching possibilities to the coarse initial line matching with $d_{\max} = d_{\text{init}}$. Further, we cluster the line matches in horizontal and vertical ones (again with the image space orientation α) and sort them by their overlap. In the last section, we already discussed the necessity of diverse line orientations to yield a robust solution.

Of both directional clusters, only $N_{\text{SAC}}/2$ line matches with the longest overlap are considered to maximize the line orientation diversity. Then, we iterate over all N_{SAC} matches and form hypotheses of two line matches for all possible combinations. Accordingly, up to

$$N_H = \sum_{i=1}^{N_{\text{SAC}}} i = \frac{N_{\text{SAC}}(N_{\text{SAC}}+1)}{2} \tag{20}$$

hypotheses are formed. Recall, that two nonparallel line matches are sufficient to calculate the 6 degrees of freedom (DOF) in the stereo case. Consequently, we disregard hypotheses whose line orientations are too similar. We employ a threshold of $45°$ by which orientations have to differ at least. For the remaining hypotheses we do the following:

1) Calculate 6 DOF transformation $\mathbf{x}_H$ for the two lines of the hypothesis with gradient descent (fixed matching).
2) If $\text{ME}(\mathbf{x}_H) > G_{\text{SAC}}$ drop hypothesis and go to 1).
3) Compute matching with all lines and $d_{\max} = d_{\text{SAC}}$ to measure consensus. Save cumulative line overlap.
4) Optional: Do an early-out test with the measures from Section IV-D to speed up the algorithm in some cases.

The hypothesis with the largest cumulative line overlap is chosen as the winner. Afterwards, the solution is refined with ICML and a comparably small $d_{\text{init}} = 2d_{\text{SAC}}$. We parameterized the sample consensus algorithm with $N_{\text{SAC}} = 60,\ G_{\text{SAC}} = 0.2\text{px},\ d_{\text{SAC}} = 4\text{px}$.

V. Experimental Results

To benchmark the presented algorithm in realistic environments, we recorded image sequences containing different visual complexity with a hand-held stereo camera. The Corridor sequence consists of 532 stereo frames over a total travel distance of approximately 48m. The sequence is noisy, due to low lighting and reflections in the ceiling and has a medium amount of visual features. A visually complex environment is tested with the Big Room sequence. It contains many three-dimensional objects and texture. Finally, in the Room sequence, the camera is moved through a mostly non-textured scene with only very few visual corners, which makes it specifically hard for feature point based systems. To evaluate the loop closure accuracy, the groundtruth end positions were computed with bundle adjustment by matching several frames in the beginning and end of the sequences by hand.

The stereo camera that was used for recording has a baseline of 16 cm, a resolution of 1280×1024 and a field-of-view of about $100° \times 80°$. The high resolution was used to preserve the sharpness during rectification. The algorithms were run on rectified sets with 640×512 pixels. For all sequences, the parameterization was left unchanged after being determined empirically. The maximum disparity for stereo matching was set to 140px. The optimization used $\mu_{\text{OM}} = 2.5$, $d_{\text{init}} = 64\text{px}$ and $d_{\text{final}} = 1\text{px}$. The registration failure detection was parameterized with $L_{\min} = 100\text{px}$, $G_{\max} = 0.7$ and $M_{\min} = 0.4$. The experiments were run on a single core of an Intel Core i7 with 2.8 GHz. No GPU acceleration was used.

The ICP implementation uses the same coarse-to-fine scheme and parameterization as ICML. The edge points for ICP are extracted from the same refined lines that ICML uses in these experiments. Recall, that ICL refers to a special case of ICML where $\mu_{\text{OM}} = 0$ and only the lines with the smallest d_{ij} are used to build a 1-to-1 matching. In Table I, ICML$_h$ denotes the use of the Huber cost function $f_h(\mathbf{e})$ instead of the MSE. Since the Huber function becomes quadratic near its minimum, the final accuracy is the same as for MSE, if the same lines are matched. Accordingly, we only compare the registration failure rates between the two.

Figure 1 shows two characteristic frames from the Room sequence along with a trajectory plot. Very few point features are found, which leads to numerous failures and an unusable trajectory with VISO2. While it generally performs very well for its algorithm category, it cannot succeed in these very sparse cases (we used the full resolution mode to maximize the chance to find features). ICML on the other hand successfully recovers the motion. Since this scene is generally easy to register for edge-based systems, ICP and ICL also complete without failures and achieve a similar accuracy as ICML (they are omitted in the plot for clarity). However, the stereo matching is very challenging for a number of frames, since horizontal lines have to be recovered in 3D to succeed. As this can be virtually infeasible for lines that do not have locatable features at both ends, we opted for the following strategy: for small numbers of detected lines, we do not only match stereo lines with ICML, but a mixture with all remaining image lines - thus simultaneous stereo and monocular optimization. Additionally, we remember 3D lines from previous frames, as long as they get matched to at least one image line in consecutive frames. For easy cases the horizontal line recovery step can directly compute the desired disparities.

Table I lists the registration failure rates for different frame increments to test for robustness with increasing distances between frames. Without sample consensus matching (the results of which are bracketed) the pure ICML convergence rates can be directly compared to ICP. It is also interesting to compare ICML with ICL in this way, as the registration failures have been approximately halved without sacrificing speed. In comparison to ICP, the registration failures are often reduced by about 30%. The comparison of the different cost functions for ICML reveals no overall winner. While the Huber function theoretically seems more robust than the MSE, the results of ICML$_h$ in Table I show no clear indication for the tested sequences. Often, ICML$_h$ beats ICML by just one successful registration. On the other hand, ICML can be superior in some cases like for the Corridor

TABLE I

REGISTRATION FAILURE RATES FOR THE CORRIDOR AND ROOM SEQUENCES. SAMPLE CONSENSUS FALLBACK RATES ARE BRACKETED.

Sequence	Corridor			Big Room		Room
Frames / Inc.	532 / 1	266 / 2	177 / 3	272 / 1	90 / 3	240 / 1
ICML	**0%** (0.4%)	**0%** (4.9%)	**0%** (11.3%)	**0%** (0.0%)	**0%** (4.4%)	**0%** (0.0%)
ICML$_h$	**0%** (0.4%)	**0%** (4.5%)	**0%** (15.3%)	**0%** (0.0%)	**0%** (3.3%)	**0%** (0.0%)
ICL	**0%** (1.7%)	**0%** (10.9%)	**0%** (19.8%)	**0%** (0.0%)	**0%** (10%)	**0%** (0.0%)
ICP	**0.9%**	**7.1%**	**16.4%**	**0%**	**5.5%**	**0%**
VISO2	**0%**	**0%**	**0%**	**0%**	**0%**	**12.9%**

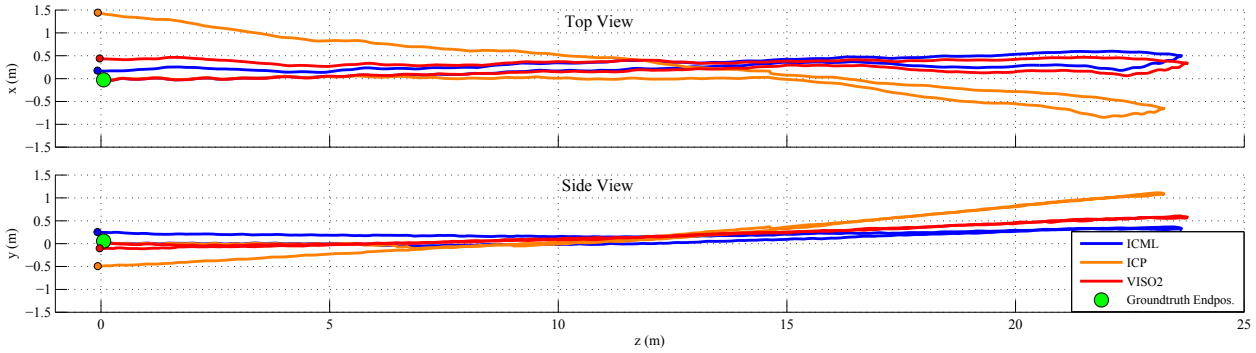


Fig. 5. In the Corridor sequence, a total distance of 48m is walked moderately fast with a hand-held stereo camera. Due to the medium feature point count, VISO2 is able to achieve an accuracy of 1%. However, with 0.65%, ICML is even more accurate since every edge pixel refines the solution, in contrast to fewer reliable corner-like features. The accuracy of ICP suffers from the lack of a registration failure recovery algorithm (3.3% accuracy) but would otherwise be comparable to ICML. ICL (not plotted) approaches the accuracy of ICML, but falls back to sample consensus matching more often.

Fig. 4. These images show initial matching configurations of the Corridor sequence. ICML was able to register the lines without sample consensus matching. ICP falls into local minima on the shown setups, because some of the closest lines are false matches. The images show the initial reprojection of the previous frame prior to registration with $d_{\max} = d_{\text{init}} = 64$px. Green depicts matched reprojected model lines, while unmatched ones are drawn in orange. Detected image lines are cyan if they were matched and dark blue for non-matched ones.

TABLE II

AVERAGE TOTAL COMPUTATION TIMES PER FRAME.

	Corridor	Big Room	Room
ICML	35+7=42ms	44+8=52ms	16+4=20ms
ICML$_h$	35+8=43ms	44+9=53ms	16+4=20ms
ICL	35+7=42ms	44+7=51ms	16+3=19ms
ICP	35+169=204ms	44+180=224ms	16+51=67ms
VISO2	91ms	103ms	63ms

sequence with large frame increment.

As the Big Room sequence is the visually most complex of the three, one would expect no big difference in accuracy between feature point and edge based methods. However, the large number of precisely detectable straight lines seems to benefit the edge based methods, see Figure 6. The trajectory of ICP is almost identical to the one of ICML, as anticipated if no registration errors occur (except for a minor misregistration of ICP near the sequence end). As ICL can only match one line, it does not always choose the best one in terms of accuracy (e.g. it could choose a small line that is closest by coincidence). While the accumulated error is not big here, it could be in other scenarios where the line detection is more unreliable.

The given time values in Table II are the mean of all measurements in a sequence. For ICML, ICL and ICP the stereo matching and line detection time (first number) and reprojection optimization time (second number) are given separately. Note, that the optimization times of ICP are more than an order of a magnitude larger than for ICML. The mean times for sample consensus (SAC) were about 50ms in the case of the Corridor sequence, with its extrema between 2ms (with early out detection) and 130ms for $N_{\text{SAC}} = 60$. Figure 5 shows the trajectories for the Corridor sequence with a frame increment of one. Due to the medium feature count of the scene, VISO2 performs well. However, ICML is twice as fast and more accurate. ICP suffers from the lack of a registration failure recovery technique to compete in such challenging scenes. While a method was proposed in [18], a significant increase of the performance gap to ICML is expected due to the use of expensive SIFT features. Furthermore, in very sparse cases, falling back to a feature point based technique may not be optimal.

VI. FUTURE WORK

The next logical step is to build a consistent map with bundle adjustment and automatic line matching. In [19], it was shown how bundle adjustment can be parameterized with lines to yield a maximum likelihood estimator without

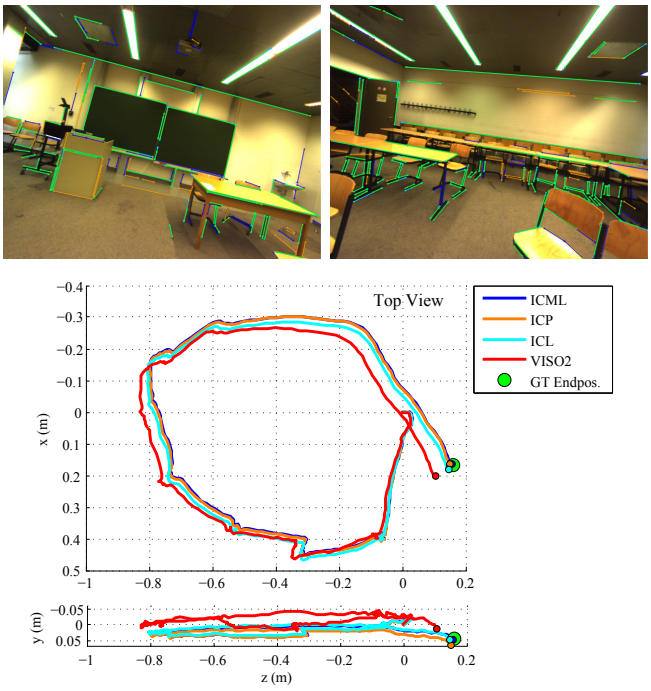


Fig. 6. The Big Room sequence contains many three dimensional objects and texture, as visible in the two frames above the trajectory plot (matched lines drawn green). With 0.2%, ICML achieves the highest accuracy on this short loop of 3m length, followed by ICL with 0.7% and ICP with 0.8%. VISO2 only achieves 2.4%, despite numerous feature points (Video is available online[1]).

overparameterization. However, since the lines were hand-matched, the relevance to real problems was limited. If ICML is used to establish the matches, a solid SLAM system could be created. With such a consistent line map, an interesting idea would be to do only monocular tracking between less frequent stereo reconstructions. Judging from the computation times, a pure monocular tracking with line detection and ICML should be feasible in under 25ms at 640×512. Saving some CPU resources for map building, one could run the tracking at 25-30Hz. With such a high frame rate, sample consensus matching should become a rare occasion. Another opportunity to cut down computation times and further increase registration success would be to use the previous delta motion as a guess for the current frame. Finally, a modified version of the sample consensus matching could be used for global pose recovery which ultimately allows for loop closure.

The source code of the developed C++ Line Vision Library is available online and will be updated as research progresses[1].

VII. Discussion

We have presented a new projective line registration algorithm, called ICML, that is substantially faster than ICP for the analogous problem. We showed that a naive "Iterative Closest Line" algorithm is inferior in terms of registration success, since it is impossible to find the ultimately best matching line in many cases. The allowance for multiple matches in ICML eliminates this hindrance and even leads to an improvement in registration performance, compared to ICP. Additionally, we proposed a robust sample consensus based bootstrapping algorithm which is automatically used once a failure criterion is fulfilled. The performance was demonstrated in an untextured and more complex, moderately textured environments. The comparison with a state-of-the-art feature point based algorithm demonstrated the potential of this technique both in accuracy and speed. However, environments without sufficient numbers of straight line segments will certainly favor traditional feature point based odometry. This suggests a hybrid system for optimal performance in diverse environments. However, a fast and self-contained edge based method can be vital for successful navigation. We already started work on automatic bundle adjustment and see great potential for real-time indoor mapping and finally the reconstruction of planar surfaces.

[1]Visit `http://www.jonaswitt.de/ICML.shtml` for the ICML source code and videos.

References

[1] M. Tomono, "Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm," in *Proc. of ICRA*. IEEE, 2009, pp. 4306–4311.

[2] J. Shi and C. Tomasi, "Good features to track," in *Proc. of CVPR*. IEEE, 1994, pp. 593–600.

[3] A. Geiger, J. Ziegler, and C. Stiller, "StereoScan : Dense 3d Reconstruction in Real-time," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 963–968.

[4] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. of CVPR*. IEEE, 2012, pp. 3354–3361.

[5] J. Meltzer and S. Soatto, "Edge Descriptors for Robust Wide-Baseline Correspondence," in *Proc. of CVPR*. IEEE, 2008, pp. 1–8.

[6] D. G. Lowe, "Fitting parameterized three-dimensional models to images," *PAMI*, vol. 13, no. 5, pp. 441–450, 1991.

[7] E. Eade and T. Drummond, "Edge landmarks in monocular slam," *Image and Vision Computing*, vol. 27, no. 5, pp. 588–596, 2009.

[8] P. Smith, I. Reid, and A. Davison, "Real-time monocular SLAM with straight lines," in *Proc. of BMVC*. IEEE, 2006, vol. 1, pp. 17–26.

[9] G. Klein and D. Murray, "Improving the Agility of Keyframe-Based SLAM," in *Proc. of ECCV*. IEEE, 2008, vol. 2, pp. 802–815.

[10] M. Tomono, "3D localization based on visual odometry and landmark recognition using image edge points," in *Proc. of IROS*. IEEE, 2010, pp. 5953–5959.

[11] M. Chandraker, J. Lim, and D. Kriegman, "Moving in stereo: Efficient structure and motion using lines," in *Proc. of ICCV*. IEEE, 2009, pp. 1741–1748.

[12] Z. N. Li, "Stereo correspondence based on line matching in Hough space using dynamic programming," *TSMC*, vol. 24, no. 1, pp. 144–152, 1994.

[13] J. Canny, "A Computational Approach to Edge Detection," *PAMI*, vol. 8, no. 6, pp. 679 – 698, 1986.

[14] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica*, vol. 10, no. 2, pp. 112–122, 1973.

[15] J. Witt and U. Weltin, "Sparse Stereo by Edge-Based Search Using Dynamic Programming," in *Proc. of ICPR*. IEEE, 2012, pp. 3631–3635.

[16] ——, "Robust Real-Time Stereo Edge Matching by Confidence-based Refinement," in *Proc. of ICIRA*. Springer, 2012, pp. 512–522.

[17] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *PAMI*, vol. 14, no. 2, pp. 239–256, 1992.

[18] M. Tomono, "Detailed 3D mapping based on image edge-point ICP and recovery from registration failure," in *Proc. of IROS*. IEEE, 2009, pp. 1164–1169.

[19] A. Bartoli and P. Sturm, "Structure-from-motion using lines: Representation, triangulation, and bundle adjustment," *Computer Vision and Image Understanding*, vol. 100, no. 3, pp. 416–441, 2005.