Online Stereo Camera Calibration From Scratch

Eike Rehder[1], Christian Kinzig[1], Philipp Bender[2] and Martin Lauer[1]

***Abstract*— Stereo cameras are among the most promising sensors for automated driving. For their deployment, however, calibration should be automated and possible *in-situ*. We propose a restructuring of bundle adjustment into an incremental online calibration system. It allows us to estimate all observable camera parameters on the fly. Both simulations and experiments with real world cameras show its capability to calibrate stereo rigs in real time while driving. With this method, cameras can be employed with almost no calibration overhead. Only the non-observable parameter of scale has to be defined in advance.**

I. Introduction

Along the road towards automated vehicles, stereo cameras have always played a significant role. To this day, they are widely applied in experimental vehicles for automated driving. Also, quite recently, they have been introduced into series production cars as sensors for driver assistance functions.

While stereo cameras provide rich information for scene understanding, they are extremely sensitive to calibration. Even smallest errors in calibration degrade performance significantly. These errors may be due to faulty initial calibration but also occur over time, e.g. due to vibrations, thermal stress or aging of materials. Thus, special attention must be paid to calibration of stereo camera systems.

For accurate calibration, the pattern based method of Zhang *et al.* [1] lays the foundation of most available tools. They require recordings of known calibration targets which can be identified in the images later. With the known structure of the targets, the calibration parameters can be inferred. A great diversity of different toolboxes is available [3], [4], [5], including the well-known MATLAB toolbox by Bouguet *et al.* [2]. All these processes require time consuming recording and processing steps as well as expert knowledge for handling the tools. Also, the camera parameters may change over time, making frequent recalibration necessary.

In order to estimate calibration parameters from unknown structure, camera self-calibration has been proposed. For the case of simple, undistorted cameras, self-calibration was introduced in [6], [7]. However, this fails if lens distortions affect the imaging process. To overcome this, joint optimization of scene structure, camera motion and camera parameters was proposed as the so-called bundle-adjustment [8], [9]. Since bundle adjustment requires a joint global optimization of all parameters, it is not feasible to apply these methods with real time constraints.

The works of Dang *et al.* aim to tackle these problems [10], [11]. They propose a reduced order bundle adjustment as well as parameter tracking using an Extended Kalman Filter. While this method can even cope with active stereo cameras, it requires a precalibration of all distortion parameters, making it unsuitable for calibration from scratch.

For the deployment of stereo cameras, it would be desirable to perform full stereo camera calibration *in-situ* and in real time. For production line setups, this enables the system to track the calibration parameters, even if they change over its lifetime. Also, in-factory calibration becomes obsolete. In the context of experimental vehicles, frequent reconfiguration of stereo camera setups would no longer entail tedious recalibration.

In this work, we propose a restructuring of the bundle adjustment problem into an incremental calibration process. From *in-situ* observations in stereo camera images, we iteratively improve the calibration in both intrinsic and extrinsic parameters. For this, we break down the problem in sequentially executed, quickly computable tasks and then only perform part-wise optimization for the full bundle adjustment per small image sequences, resulting in an ever-improving calibration in real time. For a system that performs motion estimation from camera images, i.e. *visual odometry*, the overhead of simultaneous calibration is neglectable.

II. Online Camera Calibration

A. Projection Model

As basis for future derivations, we would like to review the standard pinhole camera model with radial distortions as used in previous works [2].

A three-dimensional point under observation is denoted $\vec{X}_W = (x_W, y_W, z_W)^\top$. At a pose $T \in SE(3)$, we place our camera in this three-dimensional space. In a first step, the point $\vec{X}_W$ is transformed from some world fixed coordinate frame into the camera frame with

$$\vec{X}_C = T^{-1}\vec{X}_W. \tag{1}$$

Note here that the transform T may be augmented by a series of individual transformations (see Fig. 1).

For a 3D point $(x_C, y_C, z_C)^\top$, we obtain the normalized direction of its line of sight as

$$\vec{p}_u = \frac{1}{z_C}\begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix} = \begin{pmatrix} \chi_u \\ \eta_u \\ 1 \end{pmatrix}. \tag{2}$$

[1]Eike Rehder, Christian Kinzig and Martin Lauer are with the Institute of Measurement and Control Systems, Karlsruhe Institute of Technology, Karlsruhe, Germany {eike.rehder, martin.lauer}@kit.edu

[2]Philipp Bender is with the FZI Research Center for Information Technology, Karlsruhe, Germany pbender@fzi.de

Lens imperfections introduce distortions to the lines of sight, i.e. they shift the perceived line of sight depending on their position in the image. A common approach to model radial distortion is based on even polynomials. The distorted line of sight $\vec{p}_d$ is derived from $\vec{p}_u$ as

$$\vec{p}_d = \begin{pmatrix} \chi_d \\ \eta_d \end{pmatrix} = \begin{pmatrix} \chi_u \\ \eta_u \end{pmatrix} \left(1 + \tau_0 r^2 + \tau_1 r^4\right), \tag{3}$$

where r is the norm of the undistorted image position $r = \sqrt{\chi_u^2 + \eta_u^2}$. In our experience, we found tangential and higher order distortions as modeled in [1] to be of little relevance. Instead, the additional degrees of freedom may lead to overfitting and are thus omitted.

As the final stage, the distorted normalized image coordinates are transformed into pixel coordinates by

$$\vec{p}_C = \underbrace{\begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{=K} \begin{pmatrix} \chi_d \\ \eta_d \\ 1 \end{pmatrix} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}, \tag{4}$$

where f_u, f_v represent the focal length of the lens in horizontal and vertical direction respectively and $(c_u, c_v)^\top$ represents the principal point.

Depending on the parameters of the perspective mapping, different camera properties can be modeled, three of which are of greater relevance in this work:

1) **Real Camera:** the full model as described above,
2) **Pinhole Camera:** while projection and transforms stay the same, distortion has been corrected for, thus setting τ_1 and τ_2 equal to zero,
3) **Rectified Camera:** for this setup, two pinhole cameras have been virtually aligned by rotation and scaling such that the transform between the two is only a shift in horizontal direction.

B. Calibration

The goal of this work is to calibrate a stereo camera setup. That is, we would like to estimate all projection parameters of both cameras (*instrinsic calibration*) and their respective pose (*extrinsic calibration*). For this purpose, we utilize a set of observed landmarks $\hat{\vec{p}}_{Ci}$ and their reprojections p_{Ci} from their estimated positions $\hat{\vec{X}}_C$. These two measures together make up the well known reprojection error

$$\hat{\vec{e}}_i = \hat{\vec{p}}_{Ci} - p_{Ci}(\hat{\vec{X}}_C), \tag{5}$$

the error that is made from projecting an uncertain landmark into an image using an uncertain camera model and comparing it to an uncertain observation.

The calibration task is to find the set of intrinsic and extrinsic camera parameters as well as landmark positions that minimize the total reprojection error of all observed landmarks

$$(\vec{X}_W^N, \tau, T^M, T_c, K) = \arg\min \sum_{i=0}^{N} \hat{\vec{e}}_i^\top \hat{\vec{e}}_i, \tag{6}$$

where X_W^N is a series of N 3D-points, T^M a series of M individual camera motion steps and (τ, T_c, K) represent all stereo camera parameters, i.e. two independent sets of projection parameters for the two cameras as well as the relative pose. Solely the length of the translation vector between the two cameras is unobservable and thus held constant. This is known as the windowed bundle adjustment problem. It is solved using non-linear optimization.

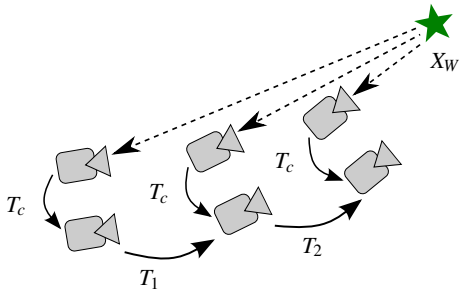


Fig. 1: Observation of a landmark in multiple camera frames. All camera parameters, transformations and landmark positions have to be estimated online.

Unfortunately, the bundle adjustment problem is computationally expensive. Thus, a full calibration in an online application is hardly possible. To overcome this problem, we break it into several parts that can each be solved in real time individually. The following steps are executed to perform full online calibration:

1) **Compute scene structure:** with the assumption of a rectified setup, compute 3D landmarks from triangulation of observations.
2) **Guess camera motion:** from the initial scene structure, estimate the camera motion between the last two consecutive frames.
3) **Select relevant landmarks:** from the motion and scene structure, find the best landmarks for further use.
4) **Improve scene structure and motion chain:** re-optimize the entire chain of motion transformations and the 3D landmark positions. The following steps can only be executed if sufficient motion is detected.
5) **Improve projection:** again, with the assumption of a pinhole camera, reestimate the projection parameters.
6) **Improve entire calibration:** with all knowledge from Steps 1)-5) as initialization, optimize all real camera parameters.

While Step 1) can be solved entirely linear, the initial guess of the camera motion can be approximated with only one iteration of non-linear optimization if the camera motion is small [12]. Step 4) is solved to full convergence with [13].

For runtime reasons, only few optimization iterations of both 5) and 6) are executed last. Then, we use the result as initial guess as well as for feature remapping for the next time instance.

C. Feature Re-Mapping

For efficient optimization of the processing steps, we have to switch between different camera models in the course of estimation.

1) From Rectified to Pinhole Camera: Rectification is achieved by rotation of lines of sight with a 3D rotation matrix R_R. If the rectified camera has projection matrix K_R and the pinhole camera K_P, then observations may be remapped with

$$\vec{p}_P = \lambda K_P R_R^{-1} K_R^{-1} \vec{p}_R, \tag{7}$$

where λ is a scaling parameter to normalize to homogeneous coordinates.

2) From Pinhole to Real Camera: The distortion function (3) is defined in forward manner, i.e. from the undistorted to the distorted case. Again, we utilize the two projection matrices of pinhole camera K_P and of real camera K_r. The distorted image point can be computed as

$$\vec{p}_r = K_r * d_r(K_P^{-1} \vec{p}_P), \tag{8}$$

where $d_r(\cdot)$ represents the distortion function (3).

3) From Real to Rectified Camera: The view from a real camera can be transformed into that of a rectified camera by substituting (7) into (8) and solving for $\vec{p}_R$,

$$\vec{p}_R = \lambda K_R R_R * d_r^{-1}(K_r^{-1} \vec{p}_r), \tag{9}$$

where, again, λ is needed for normalization. The inverse distortion $d_r^{-1}(\cdot)$ takes the same functional form as (3), where only the parameters differ. In order to obtain the parameter set that undoes distortion, we can simply distort known points and then solve the system of linear equations for the undistortion parameters.

Obviously, (7), (8) and (9) can be concatenated for arbitrary conversions.

D. Motion Estimation

The problem of estimating camera motion from subsequent images is known as visual odometry. Since we observe a sequence of motion steps, we may assume that we have obtained all but the last transformation before. For the motion from time $t-1$ to t, we minimize the reprojection error of tracked landmarks into the images w.r.t. the motion transform. We assume little change of motion in between consecutive frames, thus we only solve the linearized problem in one single optimization step. We evaluate the reprojection error for outlier detection. If the error is too large, we run the RANSAC algorithm for a new initial guess of motion.

E. Feature Selection

The result of the calibration highly depends on the quality of the landmark observations. We therefore have multiple criteria for landmark evaluation. First of all, in the visual odometry step, we have already computed reprojection errors per landmark or even performed RANSAC. Thus, we use the current motion estimate for outlier removal.

As a second step, we perform bucketing of landmark observations over the image. This ensures that the landmarks are distributed equally over the image. Without this step, the landmark density would be cumulated in the center of the image, leading to a biased result that preferably minimizes errors in the center.

The third quality measure is the duration of landmark visibility. Landmarks that can be tracked for a long period of time have the property of being highly discriminative. Thus, they can be located accurately throughout an image sequence.

F. Robustness and Regularization

The detection of landmarks in images is prone to a multitude of errors, among which are random noise and occasional outliers. These errors may degrade the performance of the optimization or, in the case of outliers, even fully break it. In order to cope with the given conditions, we introduce countermeasures to the optimization.

The first countermeasure is a robust loss function that is applied to the reprojection error. We employ a Cauchy Loss Function that reduces the impact of large outliers on the optimization result. Even with the previous outlier rejection, this improves performance greatly.

As a second measure for robustness, we desire that the supposedly static calibration parameters only change slowly over time. For this, we use a regularization term in the error function. The change of each parameter with respect to its last value is introduced to the error. Let, for example, f_u^- be the estimated focal length from the last full optimization step. Then we introduce

$$e_{\Delta f_u} = \lambda_{f_u}(f_u - f_u^-)^2 \tag{10}$$

as a new error term, where λ_{f_u} is a non-negative weighting constant. Larger values for λ_{f_u} lead to slower change of parameters and more robustness. Smaller values lead to faster calibration but less robustness.

Due to the special field of application in a driving vehicle with wide horizontal field of view, we found it beneficial to also regularize for isotropic projection, i.e. the focal length in horizontal as well as vertical direction should be close to equal. This is achieved by introducing

$$e_{\Delta f} = \lambda_{\Delta f}(f_u - f_v)^2. \tag{11}$$

to the error function. The reason for this is that due to the wide horizontal field of view, errors are more pronounced in horizontal direction.

Since the basewidth of the stereo setup, i.e. scale, cannot be observed, we hold the length of the translation vector between the two cameras constant at all times.

G. Threshold Decay

Over time, the calibration parameters improve and thus, they may need less change but instead can be used for stricter outlier detection. Therefore, we adapt the thresholds for outlier detection over time.

The outlier removal threshold at initialization ε_{max} should be fairly high since at this time, reprojection may not be computed accurately. With better reprojection estimation, the outlier threshold may decline to a lower level ε_{min} to still

allow for some tolerance. This is achieved with a threshold ε subject to

$$\varepsilon = (\varepsilon_{max} - \varepsilon_{min}) \exp\left(-\frac{t}{T_\varepsilon}\right) + \varepsilon_{min}, \tag{12}$$

where t is the run time and T_ε is the time constant governing the duration of the threshold decline. We choose an exponential function since the improvement of reprojection error is large in the beginning where extrinsics are corrected quickly. Later, the influence of calibration declines and so should the threshold decay.

H. Parameter Initialization

The parameters have to be initialized for the optimization to find a valid solution. We have two sets of intrinsic parameters per camera to initialize, namely the distortion and the projection parameters, as well as the extrinsic transformation between the two.

We initialize the principal point as the center of the image and the focal length for a horizontal field of view of 90°, i.e. half the width of the image in pixels. Distortion is initialized with all-zero values. If no information on the basewidth is available, $b = 1$ can be assumed.

In some cases, the user may know specifications of the camera setup. If this is the case, these parameters can be passed to the calibration in advance. This is especially relevant for the baseline that determines scale. In most cases, it can be measured by hand to give a close approximate of the real value.

III. Experiments

Since a ground truth calibration cannot be obtained for real camera system, we perform evaluation in two different ways: first, we use a simulated stereo camera setup to evaluate the accuracy of parameter estimation. For real world calibration, we quantify quality based on results of algorithms that require calibration, namely visual odometry [12] and disparity computation [14]. All real-world calibrations use the point feature matching from [12].

A. Simulated Camera Rig

Ground truth calibration parameters do not exist for real camera systems. Thus, we simulate a projection with known parameters to evaluate our online calibration. For the simulation, we use projection of randomly created 3D-landmarks with a sequence of motion taken from the KITTI dataset. As calibration parameters, we also employ the parameters of the rectified KITTI cameras but introduce decalibration manually. We added displacement of the cameras together with mustache distortion (see Fig. 2).

Since the parameters have a diverse range of values, we normalize all errors w.r.t. to the ground truth (GT) value and evaluate the deviation from that, e.g. for focal length

$$e = \frac{f_{Est} - f_{GT}}{f_{GT}}. \tag{13}$$

Results for the calibration can be seen in Figure 3. For clarity of presentation, we only show parameters of the left camera and also omit the principal point. The later experiments will demonstrate the validity for both cameras and pose estimation.

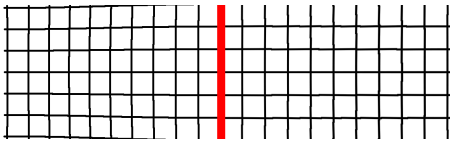


Fig. 2: The simulated mustache distortion of straight lines (left) and after online calibration (right).

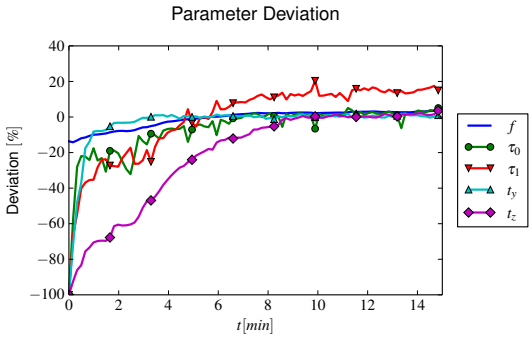


Fig. 3: Deviation of instrinsic parameters of left camera from the simulated values. The value 0 means our calibration and the simulated values coincide.

Most parameters converge to their actual simulated value. Only the parameter of distortion of 4^{th} degree is slightly overestimated due to simulated noise. However, this error is barely noticeable in the undistortion (see Fig. 2).

B. Real-World Stereo Cameras

For real world evaluation, we tested three different stereo camera setups.

1) **KITTI:** the unrectified and unsynchronized raw data from the KITTI dataset [15] together with the supplied calibration parameters.
2) **Low Distortion:** a stereo setup with Zeiss Distagon T* 15mm f/2.8 ZF.2-I lenses, designed for distortion as low as 0.3%.
3) **High Distortion:** a stereo setup with Lensagon BM4018S118 lenses with pronounced distortion and a field of view of more than 120°.

Table I gives an overview over the relevant parameters of the stereo systems. All setups have been calibrated using offline calibration with checkerboard targets, using either [3] or [5].

TABLE I: Parameters of the systems under test

Setup	f [mm]	b [m]	Dist. [%]
KITTI [15]	4.5	0.53	~ 2
Low Distortion	15	0.57	0.3
High Distortion	4	0.3	> 5

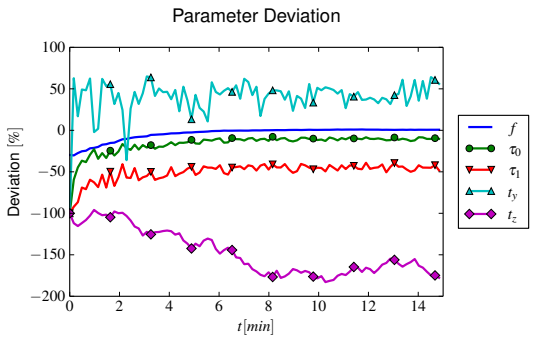


Fig. 4: Deviation of instrinsic parameters of left camera from those obtained from offline calibration [12]. The value 0 means our calibration and the offline results coincide.

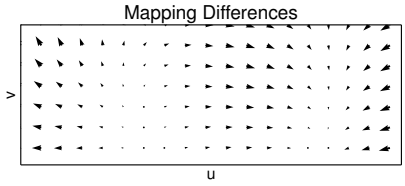


Fig. 5: Remapping difference map. The maximum shift is 10.7 pixels.

C. Parameter Accuracy

In a first attempt, we compare the results of our calibration to a calibration performed offline with checkerboard patterns [3]. We use the calibration provided with KITTI raw sequences.

Figure 4 shows how the deviation of parameters changes over time. While the focal length is estimated to have the same value, the rest of the parameters converge to quite different values. The impact of this can be seen when we look at the difference of the remapping that is applied to the image for rectification. Figure 5 shows the magnitude and orientation of these differences. Since we can add an arbitrary offset to remapping, we have compensated for the mean of differences.

Due to differences in distortion, the remapping differs in the center as well as in the extreme regions. The consequences of this can be seen in stereo matching results in Figure 8. With the original calibration, the matching in the corners fails while ours allows dense disparity computation throughout the entire image. Thus we assume that pure comparison of parameters is not meaningful to evaluate quality of calibration.

D. Stereo Matching Density

Disparity computation highly depends on the quality of calibration, as seen in the previous section. Since a known calibration is the prerequisite for disparity evaluation, we only evaluate stereo matching in terms of disparity density. The intuition behind this is that a denser stereo image

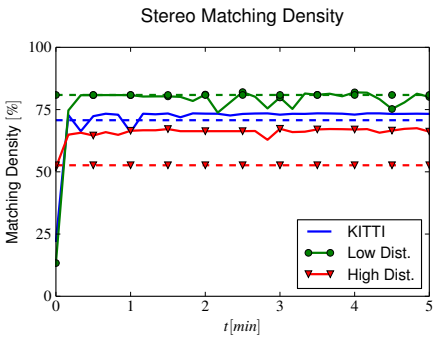


Fig. 6: Disparity image density over online calibration runtime. Dashed lines represent the density achieved with offline calibration.

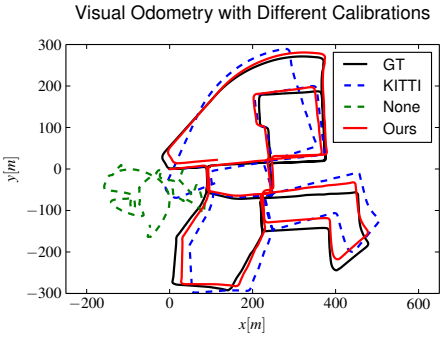


Fig. 7: Visual odometry computed for different calibrations. The black line is the ground truth (GT) obtained with DGPS.

requires a better alignment of epipolar lines and thus, a better calibration (see Fig. 8).

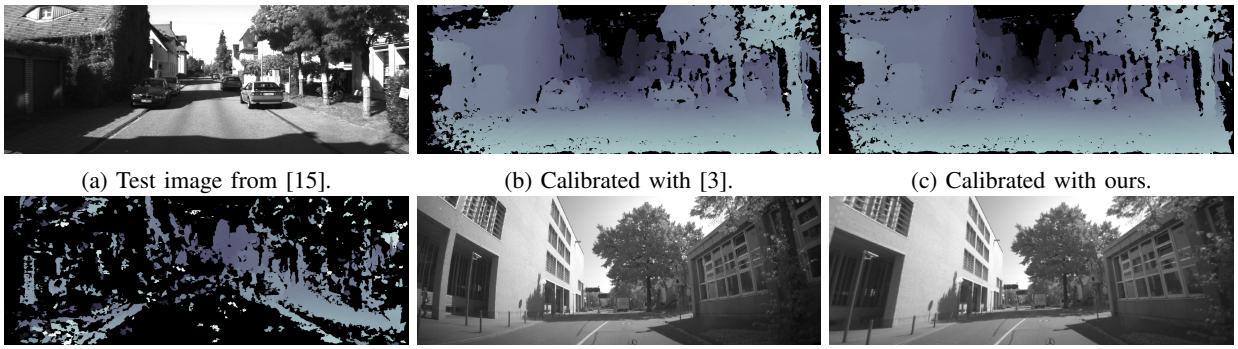
For each setup, we select a set of images from a sequence as test images. We then run online calibration on the same sequences and evaluate both, the disparity density over time as well as in comparison to an offline calibration result.

Figure 6 shows the density of disparity computation over calibration runtime. It can be seen that the density of disparity images increases over calibration. Density reaches same levels and even surpasses the result achieved with [3] and [5]. Same quality is reached after roughly 30 seconds.

E. Visual Odometry

Different studies have shown the effect of calibration on the quality of visual odometry [16]. As an additional reference, we compare visual odometry performance with the result of online calibration. For this, we utilize the first sequence from the KITTI odometry raw data. We loop it for five minutes for calibration. We then hold the resulting parameters constant and run visual odometry [12] on that sequence. Figure 7 shows the estimated motion for the different calibration states together with one run for the KITTI calibration parameters.

At initialization, calibration parameters are far from accurate, thus visual odometry fails completely. After the five minutes of processing the online calibration achieves good



(a) Test image from [15]. (b) Calibrated with [3]. (c) Calibrated with ours.

(d) Uncalibrated. (e) Distorted test image. (f) Undistorted after calibration.

Fig. 8: Stereo matching results [14] for different calibrations in (a)-(d) and undistortion result of highly distorted cameras in (e) and (f).

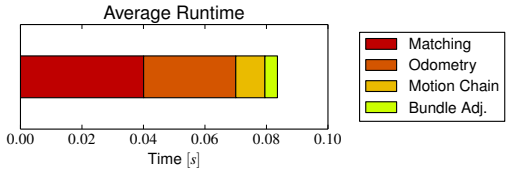


Fig. 9: Average runtime of the entire processing chain

results so that the quality of visual odometry computation even exceeds the one with the original KITTI calibration.

F. Runtime

The entire processing chain can be executed in real time. Average runtimes per process step can be seen in Figure 9. We run it on one core of a 2.9GHz INTEL I7-3520M CPU with images with a resolution of 512×1024. As expected, matching of image points and the initial odometry computation take up most of the runtime.

IV. CONCLUSION

In this work, we introduced an online camera calibration scheme based on bundle adjustment. By breaking the bundle adjustment into sequentially executable smaller tasks, real time constraints can be met. The use of observed scene structure for calibration allows us to calibrate all observable camera parameters, that is all intrinsic and extrinsic camera parameters up to scale, in real time. In comparison to other calibration methods, we do not require the use of any calibration pattern. Also, pre-calibration for an initial guess is not necessary.

Calibration is done with only little overhead over standard visual odometry. In the context of automated driving and experimental vehicles, this enables us to exchange the standard visual odometry with visual odometry plus calibration. This way, we do not require time consuming offline calibration in advance. Instead, we even can do alterations of the camera setup on the go without having to worry about sensitive camera parameter adjustments. The code is made publicly available at https://github.com/KIT-MRT.

REFERENCES

[1] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[2] J.-Y. Bouguet, "Camera calibration toolbox for matlab," 2004.

[3] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3936–3943.

[4] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[5] T. Strauß, J. Ziegler, and J. Beck, "Calibrating multiple cameras with non-overlapping views using coded checkerboard targets," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 2623–2628.

[6] O. D. Faugeras, Q.-T. Luong, and S. J. Maybank, "Camera self-calibration: Theory and experiments," in *European conference on computer vision*. Springer, 1992, pp. 321–334.

[7] A. Zisserman, P. A. Beardsley, and I. D. Reid, "Metric calibration of a stereo rig," in *Representation of Visual Scenes, 1995.(In Conjuction with ICCV'95), Proceedings IEEE Workshop on*. IEEE, 1995, pp. 93–100.

[8] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment: a modern synthesis," in *International workshop on vision algorithms*. Springer, 1999, pp. 298–372.

[9] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 72–79.

[10] T. Dang, C. Hoffmann, and C. Stiller, "Self-calibration for active automotive stereo vision," in *2006 IEEE Intelligent Vehicles Symposium*. IEEE, 2006, pp. 364–369.

[11] ——, "Continuous stereo self-calibration by camera parameter tracking," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1536–1550, 2009.

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

[13] S. Agarwal and K. Mierle, *Ceres Solver: Tutorial & Reference*, Google Inc.

[14] B. Ranft and T. Strauß, "Modeling arbitrarily oriented slanted planes for efficient stereo vision based on block matching," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 1941–1947.

[15] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[16] I. Krešo and S. Šegvic, "Improving the egomotion estimation by correcting the calibration bias," in *10th International Conference on Computer Vision Theory and Applications*, 2015.