Automated Camera Stabilization and Calibration for Intelligent Transportation Systems

Marcel Bruckner
Technische Universität München
Boltzmannstraße 15, 85748 Garching
marcel.bruckner@tum.de

Abstract

1. Introduction

Within the PROVIDENTIA project, a section of the highway A9 between Munich and Nuremberg was converted to a testing site for autonomous driving. As part of this, a large sensor network system has been set up along the highway to allow monitoring and steering of traffic as well as to improve the coordination between autonomous and traditional cars. The primary task of the intelligent system is to create a digital traffic twin that accurately represents the physical road situation in real-time. Based on this digital twin, the smart infrastructure can provide a far-reaching and comprehensive view to the drivers and autonomous cars in order to improve their situational awareness within the current traffic environment. (Taken from the description)

A key challenge of ITS lies in the reliable and accurate calibration of the different sensors. The calibration is especially challenging when the sensor is subject to real-life disturbances like vibration of its mounting pole caused by wind or displacements due to temperature expansion. These disturbances introduce noise to the measurements. We focus on the implications of this noise for the vision system built upon the cameras.

The focus of the project is not to accurately view and measure the vehicles in the sensor ranges, but rather to build upon these measurements. The backend tracking the vehicles are sensible noise in the measurements despite the strong efforts of to extend the robustness using extended filtering techniques. These filters already mitigate a broad range of noise but nonetheless cannot remove all.

The projects main focus is to provide accurate inter-vehicle information and to predict the future traffic development. The digital twin modeling the state of the traffic and environment drifts over time and the calibration between the nodes and of the nodes gets less accurate.

To tackle real-life disturbances and to remove noise from the system we provide two vision based frameworks. The problems arising from disturbances can be roughly grouped into problems concerning the Dynamic Stabilization of the video feed and the Static Calibration of the camera pose.

Dynamic stabilization The dynamic shaky motions of the camera due to wind and vibrations from passing vehicles are stabilized using a digital image stabilization approach. The DIS approach is based on visual image features that are matched between the current and a keyframe. Using the feature matching the homographic transformation between the frames is computed and the current frame is warped to minimize the pixel distances between the static background scene. This enables us to mitigate the real-world motions of the camera in the image space.

Static calibration Within the project high definition maps (HD maps (2.3)) of the enclosed environment are heavily used. These HD maps (2.3) offer the real-world positions of the highway lanes, the gantry bridges, objects like poles and permanent delineators and traffic signals like speed limits or exit markers.

Due to the environmental influences on the mounting constructions as well as the gantry bridges (Explain these earlier) and the natural wear of the materials the cameras positional and rotational parameters are changing over time.

Using the spatial information of the HD maps (2.3) and a mapping to pixels in the video frame the reprojection error is minimized to find the optimal camera pose for the observations.

The code for the dynamic stabilization, static calibration and object position retrieval from the HD maps (2.3) are accessible via the two GitHub repositories: https://github.com/Brucknem/GuidedResearch and https:

//github.com/Brucknem/OpenDRIVE.

2. Terms and Definitions

This section provides explanations of extensively used terms and definitions.

2.1. Digital Twin

2.2. Dynamic foreground and static scene

We group parts of the world seen by the cameras into the two groups of dynamic foreground and static scene. Dynamic foreground describes all pixels that represent objects that are meant to be moving, *e.g.* vehicles on the roads. On the other hand the static scene are all pixels that are not dynamic foreground, *e.g.* the road, guardrails or bridges.

2.3. High definition road maps (HD maps (2.3))

Several HD maps are used to generate the Digital Twin (2.1). The HD maps adhere to the OpenDRIVE standard V1.4. They contain spatial and relational information between the world, roads, the road coordinate frames and objects within these frames.

3. Related Work

There are many Intelligent Transportation System projects emerging, *e.g.* Koster *et al.* at the *Testfeld für Autonomes und vernetztes Fahren in Niedersachsen* [11]. See *e.g.* this and that ... Also the cooperative perception using infrastructure sensors is an emerging field [4].

None of these provide public information about their camera calibration and stabilization approaches.

Within the *Test Area Autonomous Driving Baden-Württemberg* [9] project one of the other projects provided information about their calibration techniques. They are too facing the problems of dynamic stabilization and static calibration. Among their test area near Karlsruhe, Germany they have built a system mostly consisting of cameras. Multiple optical camera sensors are attached to large poles with overlapping fields ov view [9].

They too use a high-precision map to calibrate the cameras. They differ by assuming the calibration to be static, whereas we present the foundations for an automated approach to self-calibration. Our approach differs that we are able to recalibrate at runtime, opposed to the a-priori approach they presented.

The approach of using landmarks like lane markings and poles is similar to ours. They also do a manual selection of landmarks beforehand.

The calibration procedure itself differs as they assume to have the exact world positions of their landmarks. We optimize for the world positions using the approximation of objects by their center lines. A common approach in this field is to minimize the squared distances between objects in the camera stream and their real world objects projections.

A drawback we face currently is that we only optimize the camera pose per pose, opposed to an global optimization strategy. This is mostly due to small overlaps in the fields of view [12] between the cameras. Additionally we face the problem of data association within the multi-view setup. Matching pixels over multiple views between the cameras is inherently hard problem [1], assigning samples along the center line of the objects to the exact corresponding pixel over multiple views is not feasible.

Additionally solving the camera calibration problem to radar sensors has been solved previously by Schöller *et al.* within our *Providentia++* project [12].

4. Approach

In this paper we propose two algorithms to solve distinct problems that arise from the real-world disturbances that act upon the Intelligent Transportation System.

4.1. Dynamic stabilization

The gantry bridges to which the cameras are mounted are prone to environmental influences. These influences include, but not exclusively, wind and vibrations from passing vehicles. These external influences bring the cameras into a swinging state which introduces jittery motion in the video feeds.

Although the displacements of the camera only span a small range around its resting position, the influences in on the images amplify by the huge distances the cameras overview.

To mitigate the noise added by the jittery motion we propose the pipeline displayed in Figure 1.

Removing dynamic foreground scene We only want to find features on the non-moving static scene (2.2) objects, *e.g.* the road, poles, guardrails and bridges. As these features are expected to not move between frames, aligning these during image warping ensures that the scene stays static. Moving vehicles are thus excluded from the detection and do not contribute in the algorithm.

Hence, we mask the current video frame and the stable reference frame using a background segmentation based on the *Improved Adaptive Gaussian Mixture Model for Background Subtraction* proposed by Zoran Zivkovic *et al.* [18, 19, 6].

Additionally it speeds up the algorithm.

Maybe append more about warmup for MOG2.

Feature detection We are looking for specific patterns or specific features which are unique, can be easily tracked and can be easily compared. (Taken from OpenCV)

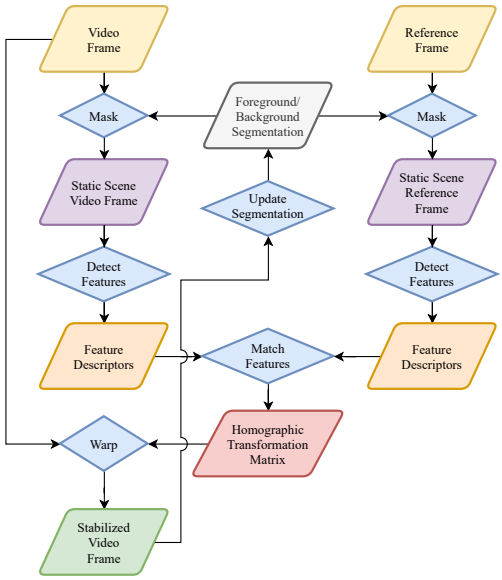

Figure 1

Hence, on the static scene (2.2) of the video frame and reference frame we perform feature detection to retrieve the descriptors and pixel locations of prominent image parts.

There exists many different implementations of feature detectors and descriptors as described by Kumar *et al.* [13]. We mainly focus on the implementations of SIFT [14], SURF [5] and ORB [17] feature detectors and descriptors, Fast [10] feature detector with FREAK [2] feature descriptors and Star [1] feature detector with BRIEF [7] feature descriptors coming from the OpenCV library [6].

Feature matching Feature matching compares the features of one frame to the features of the other frame. A match is reported if the feature descriptors of two compared features surpass a specific dynamic threshold [14] regarding some feature dependent metric [13]. These feature matches establish a spatial relationship in pixel space between the two frames.

Based on this spatial pixel relationship we can now compute the homographic transformation between the two frames. The transformation describes a perspective transformation H between $(x_i', y_i', 1)^T$ and $(x_i, y_i, 1)^T$ so that

$$s_i * (x_i', y_i', 1)^T \sim H * (x_i, y_i, 1)^T \tag{1}$$

and minimizes the back-projection error.

The implementation is included in the OpenCV library [6].

Image alignment With the given homographic transformation the current video frame is now warped so that the matches align in pixel space. As a consequence the static scene is now aligned perfectly between the two frames. As the keyframe is stable and does not change over time the current video frame is now also stabilized as the motion of the static scene is minimized.

Foreground/Background segmentation upate As a last step the foreground/background segmentation is updated using the new stabilized frame. This ensures that for the next frame the segmentation distinguishes the static scene and dynamic foreground.

This reduces the search space for the feature detectors and prohibits matches between static scene and dynamic foreground objects. Hence, the algorithm is robustified and speed up.

4.2. Static calibration

Intelligent Transportation System are inherently dependent on the calibration of the different sensors. To track and predict traffic the system has to know the poses of the different sensors relative to some reference coordinate system. This enables the ITS to accurately measure the position of vehicles within the single sensor ranges and at the overlapping boundaries.

Previous experiments have shown that a calibration process based on an IMU is not feasible in our case. Instead we focus on a calibration procedure based on visual landmarks in the video feed. The landmarks are mapped to their partially known world positions from high definition road maps.

Retrive objects from the HD maps (2.3) In this work we focus on the permanent delineator objects that are easily visible in the video feeds.

The world position of the objects can be retrieved using the mathematical operations defined in the OpenDRIVE standard.

This gives us the the base origin point $o = (x, y, z)^T$ of the object in the transverse mercator projection [16]. The base origin point is the world position of the lower end of the object where it ends in the ground or another object.

Additionally we retrieve a directional heading axis $d = (x, y, z)^T$ and the height h of the object.

These values enable us to approximate the real-world objects by sampling points $s \in S$ in world position along the center line of the object, where

$$S = \{s | s = o + \lambda * d : \quad \lambda \in [0, h]\} \tag{2}$$

Mapping objects to pixels To calibrate the camera we need to establish a mapping

$$s_c \mapsto p_c \Leftrightarrow o_c, d_c, \lambda_c \mapsto p_c \tag{3}$$

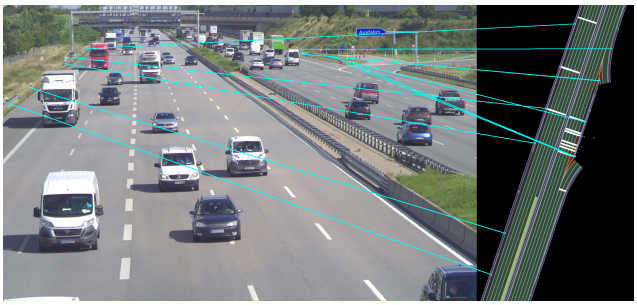

Figure 2: Left: The current camera frame. Right: A part of the HD maps (2.3). Cyan: The mapping $s_c \mapsto p_c$ from objects (right) to their corresponding pixels (left).

from pixels $p_c = (u,v)^T \in P$ from the video stream to the corresponding sampled points s_c from the object they belong to.

This leaves us with a set C of correspondences $\{p_c, s_c\}$.

This mapping is currently done by human interaction and not fully automated. To minimize the work an aiding system to mark pixels was implemented that outputs a list of pixels that can easily be mapped to the list of objects.

Relaxation of problem by line approximation Approximating the objects by lines removes the need for exactly known 3D correspondences as usually needed in calibration problems. Nonetheless it does not require to jointly estimate the full world position of the objects and camera pose jointly as in Bundle-Adjustment problems. The assumptions we made are:

- Objects are symmetric around their directional heading axis.
- Pixels of the objects are also symmetric around the projected directional axis.
- The pixels in the mapping is equally distributed around the directional axis as the deviations from tangential offset points cancel out then.

This models that the objects collapse into their center line in 2D and 3D and that tangentially deviating points on the surface in one direction equally cancels out by the opposite point mirrored at the center line.

Calibration procedure Our camera is modelled using the pinhole camera model. This uses the known intrinsic camera parameters and their corresponding projection π from camera to image space.

The extrinsic camera parameters that map from world space to camera space are defined by the camera translation T and the camera rotation R. The translation and rotation are unknown, for them we optimize.

We estimate the camera pose by minimizing a modified version of the reprojection error

$$\min_{T,R,\Lambda,W} E(P,S,T,R,\Lambda,W) \tag{4}$$

formulated as

$$\begin{aligned} s_c =& o_c + \lambda_c * h_c \\ E(P,S,T,R,\Lambda,W) =& \sum_{c\in C} \|w_c * [p_c - \pi(T * R * s_c)]\|_2^2 \\ &+ \sum_{c\in C} \|penalize(\lambda_i, h_i)\|_2^2 \\ &+ \sum_{c\in C} \|\alpha * (1 - w_i)\|_2^2 \end{aligned} \tag{5}$$

where P is the set of mapped pixels in the image, S is the set of mapped corresponding sampled points from the objects and Λ is the set of λ values associated with the sampled points. This formulation allows the optimization over the line approximations of the objects and jointly optimizes for the camera parameters T, R and the $\lambda \in \Lambda$ parameters of the line objects.

The calculation of the exact position of $s_c \sim \lambda$ allows the optimizer to search the whole space of real numbers for λ. Nonetheless we penalize values for λ that exceed the physical height of the object by

$$penalize(\lambda, h) = \begin{cases} \lambda - h, & \text{if } \lambda > h \\ \lambda, & \text{if } \lambda < 0 \\ 0, & \text{else} \end{cases} \tag{6}$$

This regularization enables a robust estimation procedure that can flexibly adjust to the missing exact world positions.

Initialization In contrast to most pose estimation problems our approach drops the need for good initialization. By regularization of the λ values enough flexibility is given to optimize over an infinite space of values, but enforces the solution of the λ to lie within the interval of $\lambda \in [0, h]$.

$$\bar{s} = \frac{1}{|C|} \sum_{c\in C} o_c \tag{7}$$

$$T_0 = \begin{pmatrix} 1,0,0, & \bar{s}_x \\ 0,1,0, & \bar{s}_y \\ 0,0,1, & \bar{s}_z + 1000 \\ 0,0,0, & 1 \end{pmatrix} \tag{8}$$

$$R_0 = \mathbb{I}^{4\times 4} \tag{9}$$

It is sufficient to initialize with $\lambda_c = 0$ for all correspondences. The camera rotation is defined to be zero with the

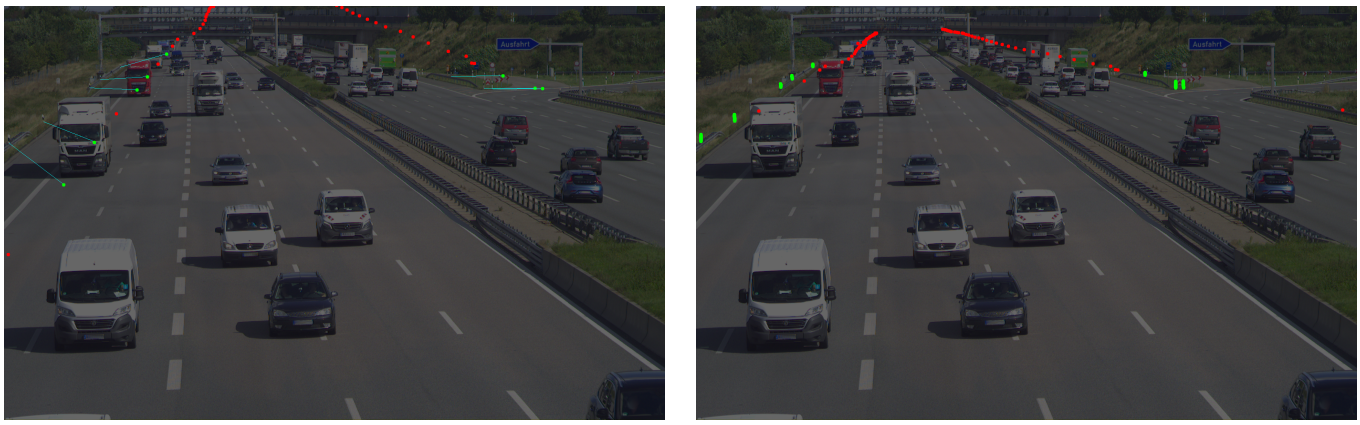

Figure 3: Left: Sampled points of objects that are mapped to pixel locations (green) and sampled points without known corresponding pixels (red) rendered by a poorly calibrated camera model. The mapping from the points to their expected pixels is drawn in cyan. Right: The same sampled points after the calibration procedure. The rendered positions of the sampled points align with the pixels of the objects they are mapped to and the drawn mapping disappears as the distance approaches 0.

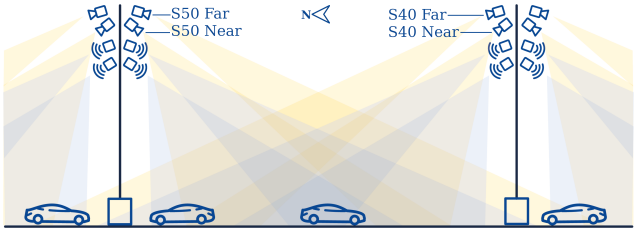

Figure 4: The schematic camera setup along the highway A9. The cameras S40 Far and S40 Near are facing north, the cameras S50 Far and S50 Near are facing south.

camera facing in negative world z axis. By placing the camera at some distance over the mean of the known object positions with zero rotation the optimization always converges to the desired minimum.

5. Evaluation

We assure the correctness and quantify the visual improvements resulting from the algorithms by an empirical study on the video streams of the highway cameras.

5.1. Study objects

We use video recordings from the four cameras mounted to the two gantry bridges internally named S40 and S50. The schematic camera setup is displayed in Figure 4.

The dataset consists of four recordings, each with of a length of 1495 frames over $\sim$ 60 seconds at 25 frames per second.

The recordings are taken on a day with strong winds to ensure high jitter in the video feed to optimally test the Dynamic Stabilization algorithm described in subsection 4.1.

5.2. Dynamic stabilization

To evaluate the Dynamic Stabilization algorithm described in subsection 4.1 we have come up with two metrics to quantify the environmental influences.

5.2.1 Optical Flow

The Optical Flow describes the apparent motion of image objects between two consecutive frames. It is a 2D vector field where each vector is a displacement vector showing the movement of points between the frames caused by movement of the objects or cameras.

We use the dense Optical Flow estimation algorithm proposed by Farnebäck [8, 6] to measure the displacement of each pixel between the frames. We calculate the mean displacement over the whole image to get the overall displacement. We see that the dynamically stabilized video feed exhibit a substantially lower mean displacement between frames as the displacement from camera jitter is removed. This leaves us with only the expected displacement resulting from moving objects, *e.g.* vehicles on the highway or shaking trees.

Figure 5 displays the mean pixel displacement per frame. It shows that the displacement with especially the SURF [5, 6] feature detector is lowered greatly. The damping approximates the removed jitter and also displays a huge lowering in mean displacement.

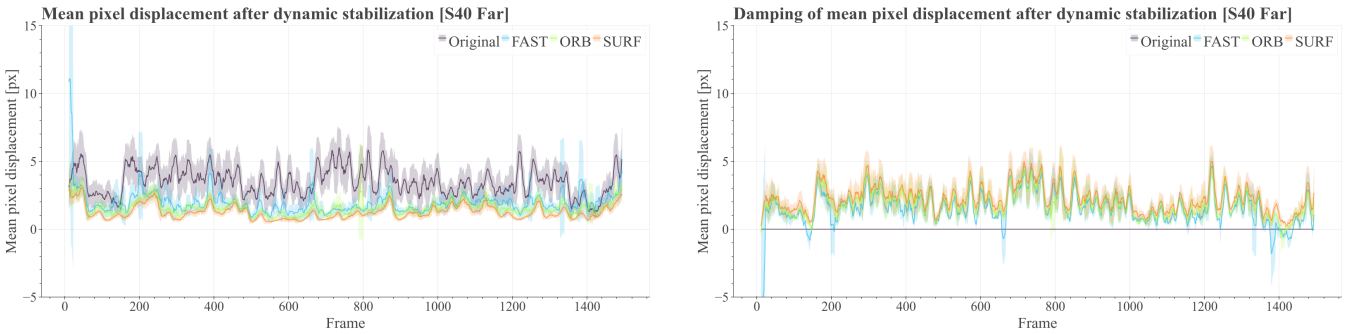

Figure 5: Left: Comparison of the three implemented dynamic stabilizers and the original not stabilized video feed using Optical Flow as metric (lower is better). The stabilizers are based on the FAST [10, 6] feature detector with FREAK [2, 6] feature descriptors, SURF [5, 6] feature detector and ORB [17, 6] feature detector. The graphs display the mean pixel shift at each frame. Right: The damping capabilities of the same three stabilizers (higher is better). The graphs approximate the removed jitter in the mean pixel shift between the original video and the stabilizer at each frame.
For visualization the values are filtered using the rolling mean over 12 frames. The light areas display the standard deviation within the window.

Stabilizer	Camera	Disp.	Better	Better [%]
Original	S40 Far	3.515	0	0.0
FAST	S40 Far	1.999	1279	85.552
ORB	S40 Far	1.632	1396	93.378
SURF	S40 Far	1.273	1478	**98.863**
Original	S40 Near	5.084	0	0.0
FAST	S40 Near	4.371	983	65.753
ORB	S40 Near	4.222	1024	68.495
SURF	S40 Near	2.495	1403	**93.846**
Original	S50 Far	2.624	0	0.0
FAST	S50 Far	2.749	983	65.753
ORB	S50 Far	1.510	1258	84.147
SURF	S50 Far	1.550	1273	**85.151**
Original	S50 Near	2.212	0	0.0
FAST	S50 Near	1.847	1028	68.993
ORB	S50 Near	1.671	1234	82.819
SURF	S50 Near	1.543	1290	**86.577**

Table 1: Comparison of the implemented dynamic stabilizers and the original not stabilized video feed using Optical Flow as metric. It shows that most of the stabilizers exhibit a lower mean displacement (Disp.) after dynamic stabilization. In terms of the number of frames the stabilizers all show a high number of total (Better) and relative (Better %) frames where they have a lower mean displacement. Especially for the SURF [5, 6] feature detector (bold) the improvement is substantially.

5.2.2 Track features and calculate path smoothness

5.3. Static calibration

- Compare to GPS position
- Compare to Google Maps/Open Street Map
- Compare to Gyro Rotation
- Compare by hand calibration
- Calculate min needed points
- Calculate expectable error

6. Future Work

The project leaves us with the opportunities to research in a broad range of directions.

6.1. Test on different weather/lighting conditions

The implementations are currently only tested on good weather and lighting conditions. A next step is to test the implementations in bad weather and lighting conditions, *e.g.* by night, rain and snow. As far as we can tell by now the feature based dynamic stabilization approach will suffer in performance as the detected features will be compromised. The static calibration nonetheless wont be impacted as we have the human-in-the-loop approach for correspondence mapping. When we switch to an automatic landmark detection algorithm the weather conditions will impact the detection and thus the calibration.

6.2. Dynamic stabilization

In the subfield of the dynamic stabilization we have until now implemented the visual feature based approach.

6.2.1 Dynamically pick best feature detector based on optical flow performance

Currently we empirically picked the SURF feature detector to have the best stabilization performance. We do measure the performance of the feature based dynamic stabilization approach based on the mean pixel deviation of the optical flow. As multiple feature detectors are already implemented together with the optical flow to measure the performance. To test would be to run all detectors in parallel and use the one with the most stabilize video output.

6.2.2 IMU based sensor fusion

Previous by-hand calibrations have shown that an inertia measurement unit (IMU) can be used to initially find a camera pose, but the lack of performance and huge time needed makes this not feasible. This is just empirical knowledge, not backed up by research. This should be looked into further.

6.2.3 Warp field stabilization based on Optical Flow

Currently the optical flow is only used to measure performance but could be used as an alternative approach to the dynamic stabilization problem. The optical flow is modelled as a vector field over the image and contains the pixel movement as direction vector and distance per pixel. By inverting this vector field and thus applying the inverse of the jitter the image could be stabilized using image warping.

6.2.4 Deep Learning based approaches

With the recent emerge of deep learning based approaches in computer vision, especially with convolutional neural networks (CNN), a self-learning stabilization procedure might be developed. This might speed up the procedure and would inherently add a measure for the uncertainty of the results when finding the homographic transform. Additionally the feature detection, matching and warping steps could be fused into one step and optimized jointly.

As far as we can tell this can be broken down into two approaches based on different loss functions.

Image MSE for calibration (unsupervised) Input the current frame and reference frame, output the homographic transformation. Learning is done by minimizing the mean squared error of pixel distances between the frames with the current transformation.

Logistic regression on pose (supervised) Input the current frame and reference frame, output the homographic transformation. Learning is done by logistic regression over the expected and predicted transform.

6.3. Static calibration

For the static calibration there are several improvements possible.

6.3.1 Extend to global refinement of multiple cameras

Currently the reprojection error as described in Eq. 5 optimizes the pose of one camera w.r.t. the HD maps (2.3). By extending the optimization to jointly optimize over all cameras and their mappings a global calibration procedure can be established. The procedure might be formulated as formulated as

$$\begin{aligned} E(P,S,T,R,\Lambda,W) = &\sum_{i\in Cams}\sum_{c\in C_i} \|w_c * [p_{i,c} - \pi(T_i * R_i * s_c)]\|_2^2 \\ &+ \sum_{i\in Cams}\sum_{c\in C_i} \|penalize(\lambda_c, h_c)\|_2^2 \\ &+ \sum_{i\in Cams}\sum_{c\in C_i} \|\alpha * (1 - w_c)\|_2^2 \end{aligned} \tag{10}$$

where one should optimize over all cameras $i \in$ Cameras and the seen seen correspondences $c \in C_i$. This would finally give a per camera transformation T_i and rotation R_i that would be globally optimal also in relation to the other cameras.

6.3.2 Automatic detection of more landmarks after initial calibration

Currently the detection and mapping of landmarks is done by hand using a watershed algorithm [15, 6] based external tool. This tool relies on a by hand marking of landmarks as the markers used to find the segmentation. By applying template, color or gradient matching approaches the detection of the markers might be automated. This automated detection would speed up the mapping procedure.

6.3.3 Automatic mapping of pixels to objects

To establish the mapping of the correspondences a human has to look up the ids of the landmarks in the HD maps (2.3). After an initial calibration an image region based approach might be used to automate this mapping. This positions of the known objects from the HD maps (2.3) can be projected into the camera image with the current camera pose. Starting from the resulting pixel locations one could search in a defined enclosing region to search for pixels that clearly correspond to the objects. This automatic detection of more landmarks could further improve the robustness an can be used to mitigate drift in the cameras.

6.3.4 Machine learning based pose estimation

A machine learning based approach for the bundle adjustment problem we faced and the resulting camera pose estimation problem can be formulated. As Aravkin *et al*. [3] have shown a Student's-t distribution based approach can be used to estimate and robustify the procedure against outliers.

6.3.5 New HD map

The newer OpenDRIVE standards also provide the possibility to include lane markings. These lane markings are easily detectable and can then be used for the calibration procedure in conjunction with the object landmarks. This would greatly simplify the automatic detection and mapping procedures as described in Sec. 6.3.2 and Sec. 6.3.3 as they are spatially more extend and thus easier to detect. Additionally with the color information, as the lane markings are always white (Germany) and yellow (USA) the detection is simplified a lot.

7. Conclusion

References

[1] Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. Censure: Center surround extremas for realtime feature detection and matching. In *European Conference on Computer Vision*, pages 102–115. Springer, 2008. 2, 3

[2] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, 2012. 3, 6, 10

[3] Aleksandr Aravkin, Michael Styer, Zachary Moratto, Ara Nefian, and Michael Broxton. Student's t robust bundle adjustment algorithm. *Proceedings / ICIP ... International Conference on Image Processing*, 11 2011. 8

[4] Eduardo Arnold, Mehrdad Dianati, Robert de Temple, and Saber Fallah. Cooperative perception for 3d object detection in driving scenarios using infrastructure sensors. *IEEE Transactions on Intelligent Transportation Systems*, 2020. 2

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 3, 5, 6, 10

[6] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 2, 3, 5, 6, 7, 10

[7] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, page 778–792, Berlin, Heidelberg, 2010. Springer-Verlag. 3

[8] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In Josef Bigun and Tomas Gustavsson, editors, *Image Analysis*, pages 363–370, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. 5

[9] Tobias Fleck, Karam Daaboul, Michael Weber, Philip Schörner, Marek Wehmer, Jens Doll, Stefan Orf, Nico Sußmann, Christian Hubschneider, Marc René Zofka, et al. Towards large scale urban traffic reference data: Smart infrastructure in the test area autonomous driving baden-württemberg. In *International Conference on Intelligent Autonomous Systems*, pages 964–982. Springer, 2018. 2

[10] Morteza Ghahremani, Yonghuai Liu, and Bernard Tiddeman. Ffd: Fast feature detector. *IEEE Transactions on Image Processing*, 30:1153–1168, 2021. 3, 6, 10

[11] Frank Köster and Mathias Höhne. Testfeld für autonomes und vernetztes fahren in niedersachsen. 2017. 2

[12] Annkathrin Krämmer, Christoph Schöller, Dhiraj Gulati, Venkatnarayanan Lakshminarasimhan, Franz Kurz, Dominik Rosenbaum, Claus Lenz, and Alois Knoll. Providentia - a large-scale sensor system for the assistance of autonomous vehicles and its evaluation, 2020. 2

[13] Rekhil M Kumar and K Sreekumar. A survey on image feature descriptors. *Int J Comput Sci Inf Technol*, 5:7668–7673, 2014. 3

[14] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 3

[15] Fernand Meyer. Color image segmentation. In *1992 international conference on image processing and its applications*, pages 303–306. IET, 1992. 7

[16] PROJ contributors. *PROJ coordinate transformation software library*. Open Source Geospatial Foundation, 2021. 3

[17] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, Nov 2011. 3, 6, 10

[18] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2 - Volume 02*, ICPR '04, page 28–31, USA, 2004. IEEE Computer Society. 2

[19] Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recogn. Lett.*, 27(7):773–780, May 2006. 2

8. Appendix

To declutter the paper we moved some of the images here.

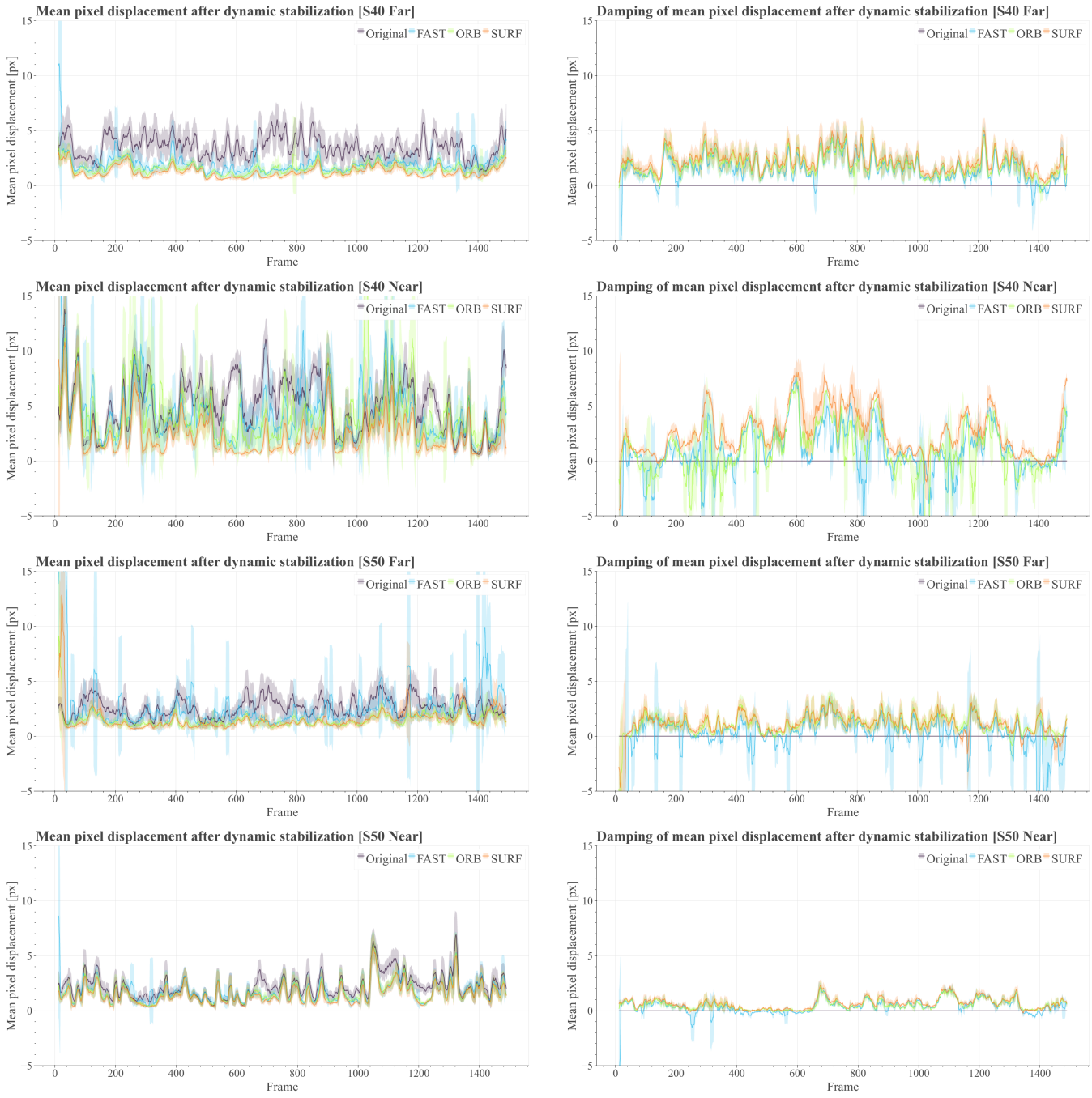

Figure 6: Left: Comparison of the three implemented dynamic stabilizers and the original not stabilized video feed using Optical Flow as metric (lower is better). The stabilizers are based on the FAST [10, 6] feature detector with FREAK [2, 6] feature descriptors, SURF [5, 6] feature detector and ORB [17, 6] feature detector. The graphs display the mean pixel shift at each frame. Right: The damping capabilities of the same three stabilizers (higher is better). The graphs approximate the removed jitter in the mean pixel shift between the original video and the stabilizer at each frame.
For visualization the values are filtered using the rolling mean over 12 frames. The light areas display the standard deviation within the window.