Automated Camera Stabilization and Calibration for Intelligent Transportation Systems

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

0.1. 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 (??) 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 [1]. 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

$$S = \{s | s = o + \lambda * d : \quad \lambda \in [0,h]\} \tag{1}$$

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

$$s_c \mapsto p_c \Leftrightarrow o_c, d_c, \lambda_c \mapsto p_c \tag{2}$$

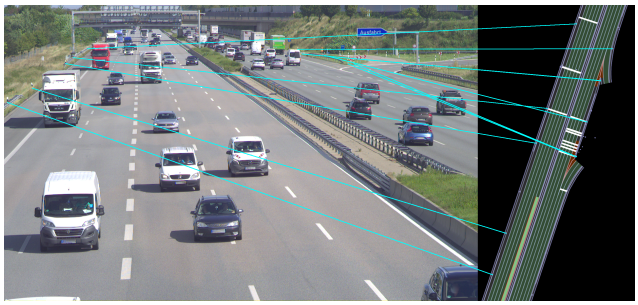

Figure 1: Left: The current camera frame. Right: A part of the HD maps (**??**). Cyan: The mapping $s_c \mapsto p_c$ from objects (right) to their corresponding pixels (left).

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 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.

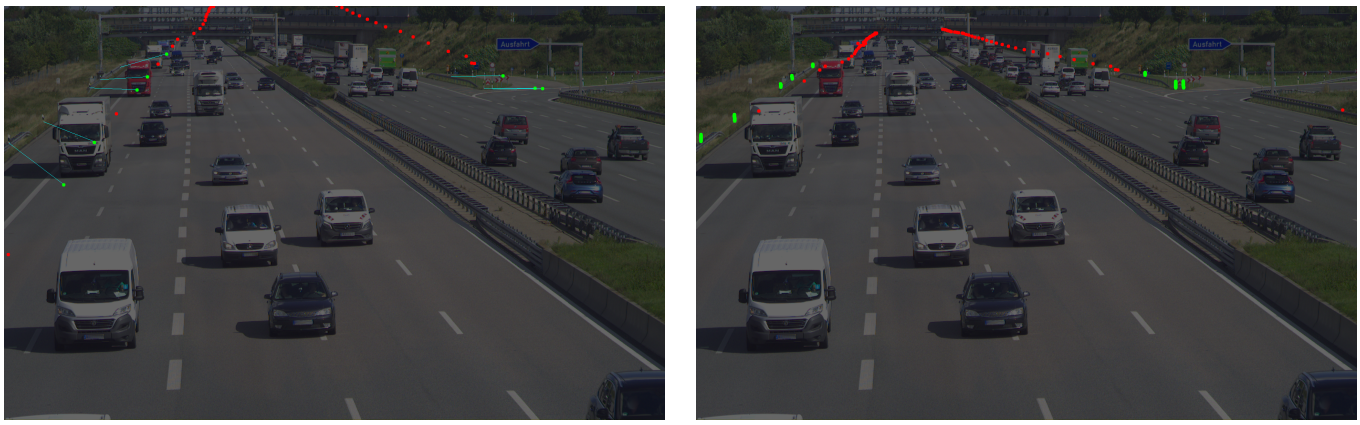

Figure 2: Left: Sampled points of objects that are mapped to pixel locations (green) and sampled points without known corresponding pixels (red) rendered by a poorly calibrated camera model. The mapping from the points to their expected pixels is drawn in cyan. Right: The same sampled points after the 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.

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

$$\min_{T,R,\Lambda} E(P,S,T,R,\Lambda) \tag{3}$$

formulated as

$$\begin{aligned} s_c &= o_c + \lambda_c * h_c \\ E(P,S,T,R,\Lambda) &= \sum_c \left\| \begin{pmatrix} p_c - \pi(T * R * s_c) \\ penalize(\lambda_c, h_c) \end{pmatrix} \right\|_2^2 \end{aligned} \tag{4}$$

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

$$penalize(\lambda, h) = \begin{cases} \lambda - h, & \text{if } \lambda > h \\ \lambda, & \text{if } \lambda < 0 \\ 0, & \text{else} \end{cases} \tag{5}$$

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 move over an infinite space of values, but enforces the solution of the λ to lie within the interval of $\lambda \in [0, h]$.

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

translation can be and initializing the camera at the some distance over the mean

References

[1] PROJ contributors. *PROJ coordinate transformation software library*. Open Source Geospatial Foundation, 2021. 1