In-operation calibration of clock-bias and intrinsic parameters for pan-tilt-zoom cameras based on keypoint tracking

Larsen, Martin Vonheim, Haavardsholm, Trym, Mathiassen, Kim

Martin Vonheim Larsen, Trym Haavardsholm, Kim Mathiassen, "In-operation calibration of clock-bias and intrinsic parameters for pan-tilt-zoom cameras based on keypoint tracking," Proc. SPIE 11538, Electro-Optical Remote Sensing XIV, 115380D (20 September 2020); doi: 10.1117/12.2573611

SPIE.

Event: SPIE Security + Defence, 2020, Online Only

In-Operation Calibration of Clock-Bias and Intrinsic Parameters for Pan-Tilt-Zoom Cameras Based on Keypoint Tracking

Martin Vonheim Larsen[a,b], Trym Haavardsholm[a], and Kim Mathiassen[a,b]

[a]Norwegian Defence Research Establishment, Instituttvn 20, Kjeller, Norway
[b]Institute of Technology Systems (University of Oslo), Gunnar Randers vei 19, Kjeller, Norway

ABSTRACT

We propose a method for jointly estimating intrinsic calibration and internal clock synchronisation for a pan-tilt-zoom (PTZ) camera using only data that can be acquired in the field during normal operation. Results show that this method is a promising starting point towards using software to replace costly timing hardware in such cameras. Through experiments we provide calibration and clock synchronisation for an off-the-shelf low-cost PTZ camera, and observe a greatly improved directional accuracy, even during mild manoeuvres.

Keywords: Pan-tilt-zoom camera, in-operation calibration, clock-bias, factor graph calibration, keypoint tracking calibration

1. INTRODUCTION

Pan-tilt-zoom (PTZ) cameras offer the versatile combination of wide-area coverage and high-fidelity long-range data, at a low cost and with tractable processing requirements. As part of advanced target tracking systems, PTZ cameras can provide both precise directional measurements as well as human-level object classification and target identification, all the while enabling a sound balance between scene overview and long-range performance.

In order to use observations in contexts outside the image itself, many applications require PTZ cameras to provide the viewing direction in the platform coordinate system for any pixel. This is certainly the case for visualisation, for instance on a map or in a panorama, or for fusing information across multiple sensors in a target tracker. Ideally, the viewing direction should be known with an accuracy on the scale of the pixel size. Today, even many low-end off-the-shelf PTZ cameras have pan/tilt encoders capable of a relative pointing accuracy better than 0.1mrad*, giving FullHD-cameras with a few degrees of field-of-view (FOV) a directional accuracy corresponding to less than 5px. In order to maintain this pixel-level accuracy during camera manoeuvres, the camera and the sensors providing pan/tilt/zoom measurements must be time-synchronised. Figure 1 shows a fixed point in the platform frame that is projected into the image as the camera is manoeuvring. As the figure demonstrates, insufficient clock synchronisation between the camera and the pan/tilt-unit can produce projection errors on the scale of hundreds of pixels, even under modest manoeuvring.

Ideally, PTZ cameras should have hardware support for acquiring both timestamped images and pan/tilt/-zoom measurements. This can either be achieved by triggering both the pan/tilt measurement and image frame acquisition with an external signal, or by using some type of hardware supported protocol to synchronise the clocks in the camera unit and the pan/tilt unit. Regardless of how the timestamping is handled internally in the PTZ camera, the timestamps must be made available through an API in order to be useful. Figure 2 indicates how this affects the accuracy of many the available off-the-shelf PTZ cameras in the $500 - $5k price range† that are simply not designed to be able to provide millisecond-accurate timestamps for neither images nor pan/tilt measurements.

Further author information: (Send correspondence to Martin Vonheim Larsen)
Martin Vonheim Larsen: E-mail: martin-vonheim.larsen@ffi.no, Telephone: +47 66 93 48 22
*In section 4 we for instance find that the Axis Q6215 has σ_{pan} and σ_{tilt} of around $5 \cdot 10^{-5}$rad.
†Such as the Axis P- and Q-series, HIKVision DE- and DF-series or FLIR Quasar-series.

Electro-Optical Remote Sensing XIV, edited by Gary W. Kamerman, Ove Steinvall, Proc. of SPIE
Vol. 11538, 115380D · © 2020 SPIE · CCC code: 0277-786X/20/$21 · doi: 10.1117/12.2573611

Figure 1: A fixed point in the platform frame projected into images from a PTZ camera where the image timestamps and pan/tilt timestamps are out-of-sync by about 50ms. The camera is panning to the right, and suddenly comes to a stop. The left and centre images are taken while the camera is panning towards the right. The right image is taken after the camera has stopped panning. On a properly synchronised and calibrated system, the projected point should appear fixed against the background.

Another complication preventing pixel-level directional accuracy in PTZ cameras is the absence of accurately calibrated intrinsic camera parameters. Several existing methods aim to holistically model the intrinsic camera parameters as a function of the measured zoom level.[1] It has been observed that inaccuracies in the zoom measurements can cause significant error in the intrinsic parameter estimates, and that the zoom measurements can drift over time.[2] In low-end systems we suspect that these measurement errors in the zoom mechanism may be too severe for the holistic "calibrate once" methods to work over time. Instead, we claim these cameras may require intrinsic re-calibration every time the zoom level is changed.

In order to bring these low-cost PTZ cameras to their full potential we need a calibration routine that can provide current camera intrinsics, as well as the current clock-offset between images and pan/tilt measurements. We here propose a method that estimates these parameters based on observations of keypoint tracks during normal camera manoeuvres together with pan/tilt-measurements. For this routine to be usable in the field, it must be quick, and should not require specific calibration targets. At a high level, we follow the method proposed by Furgale et al.[3] by constructing a full continuous-time bundle adjustment problem over the pan/tilt-measurements with both the camera intrinsics, the clock-offset and all camera poses and landmark positions as variables. For a stationary PTZ camera operating with narrow FOV the translation of the optical centre is usually negligible, making the distance to landmarks unobservable. Several existing methods apply considerable efforts to constrain the underdetermined dimensions and explicitly solve for the camera parameters.[4–6] By formulating the bundle adjustment problem as a factor graph using rotation- and direction-only manifold representations of image poses and landmark positions, our problem is fully determined. This formulation has several benefits. Firstly, having fewer parameters should yield more robust estimates, as well as faster convergence. Secondly, the flexible factor graph framework paves the way for including other sensors such as IMUs, or using incremental optimisation techniques for real-time per-frame estimation of parameters. Through the full continuous-time bundle adjustment where *all* available information is included, our maximum a posteriori (MAP) estimates should be better than estimating these parameters separately, as suggested by Furgale et al.[3]

The primary contribution of this paper is our method for jointly estimating intrinsic parameters and clock-offset in PTZ cameras. To our knowledge no similar method for time calibration has been published that does not require external apparatus or calibration targets. Secondly, our method offers a vastly simplified procedure for obtaining intrinsic calibration. Existing methods either require capturing a full 360° panorama[1] or capturing images at multiple zoom levels.[2] Finally, we believe the application of factor graph techniques is a step forwards in terms of simplicity and flexibility over existing methods.

2. METHOD DEVELOPMENT

The goal of this section is to provide the theoretical foundation needed to formulate the continuous-time bundle adjustment problem that contains the intrinsic calibration and clock offset as parameters. Since we do not

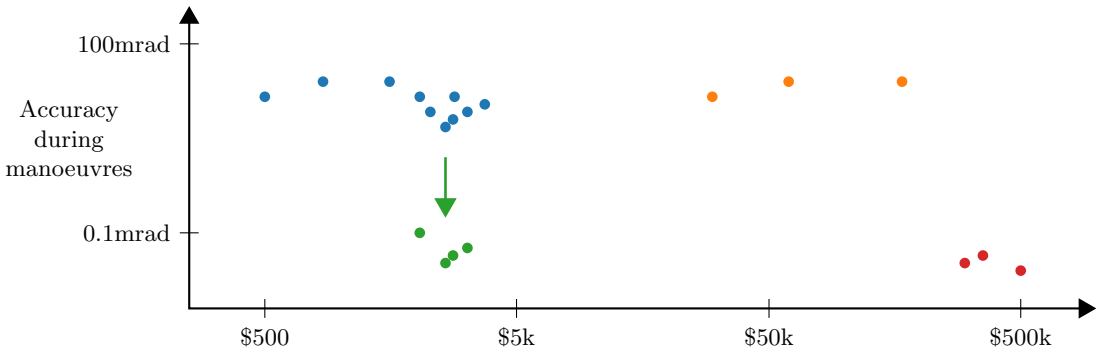

Figure 2: The pointing accuracy during manoeuvres vs price for different classes of PTZ cameras. The plot is not meant to accurately depict the accuracy or price of specific camera models. PTZ cameras that lack synchronised timestamps for pan/tilt measurements and images observe severely degraded pointing accuracy during manoeuvres. This is generally the case for low-end PTZ cameras (blue), and in our experience even for several mid-range actively stabilised cameras (orange). We presume that high-end ISTAR cameras (red) maintain good accuracy overall, but these are in a completely different price range. Our method is able to provide clock synchronisation for some of the low-end PTZ cameras, resulting in greatly increased performance during manoeuvres (green).

know the true camera orientation for each image nor the true landmark positions, our problem resembles Visual Odometry (VO),[7] albeit with rotation-only camera poses and direction-only landmarks. Thus, we choose to formulate the problem as a MAP estimation problem, which is standard in current VO and Visual Simultaneous Localisation And Mapping (VSLAM) methods.[8]

We begin this section by giving an overview of how we can model a nonlinear MAP inference problem using *factor graphs* in section 2.1. This framework will allow us to express probabilistic constraints on and between state variables, based directly on uncertain measurements and corresponding (possibly) nonlinear *measurement models*. Since we will need to linearise measurement models over orientations and directions, we cover how to do this using *manifold representations* in section 2.2. We then proceed to present our underlying models, including the clock model (section 2.3), the continuous-time pan/tilt motion model (section 2.4), and our camera model (section 2.5). Finally, we assemble the measurement models and present the complete factor graph for solving our MAP problem in section 2.6.

2.1 MAP Estimation using Factor Graphs

This section is based on the comprehensive overview of factor graphs for robotics by Dellaert and Kaess.[9]

Let $X = \{x_j\}$ denote a set of unknown state variables and $Z = \{z_k\}$ be a set of measurements involving X. Assume that the measurements can be modelled with a generative probabilistic model $z_k = h_k(X_k) + \boldsymbol{\eta}_k$, where h_k is a known measurement prediction function, X_k is the subset of X involved in this model, and $\boldsymbol{\eta}_k$ is a random variable with known distribution. The MAP estimate of X is then given as the X^* that maximises the posterior distribution

$$X^* = \arg\max_X p(X \mid Z) = \arg\max_X p(Z \mid X)p(X), \tag{1}$$

where the last equality in eq. (1) follows from Bayes' law and the fact that $p(Z)$ is constant over X. The distribution $p(X)$ is the prior distribution of X. When no prior information is known about X, $p(X)$ is uniform, and does not affect the estimate of X^*.

If we assume that all $p(x_j)$ and $\boldsymbol{\eta}_k$ are stochastically independent, we can write eq. (1) as

$$X^* = \arg\max_X \prod_j p(x_j) \prod_k p(z_k \mid X_k). \tag{2}$$

Since the arg max does not depend on the posterior being normalised, we can instead maximise over any likelihood functions $l(X_k;\, z_k) \propto p(z_k \,|\, X_k)$, which means that eq. (2) can be re-written as

$$X^* = \arg\max_X \prod_j p(x_j) \prod_k l(X_k;\, z_k). \tag{3}$$

We use the notation $l(X_k;\, z_k)$ to emphasise the fact that z_k here is constant and that $p(z_k \,|\, X_k)$ is just a function of a subset of state variables X_k. In fact, the above represents a factored posterior probability density on the unknown state variables only, albeit unnormalised.

We can represent this factorisation explicitly using a *factor graph*. This is a graphical model consisting of two sets of nodes: A set of *factor nodes* $\phi_i \in \mathcal{U}$, and a set of *variable nodes* $x_j \in \mathcal{V}$ representing each of the state variables. Edges $e_{ij} \in \mathcal{E}$ are always between factor nodes and variable nodes, and encode independence relationships, with each factor ϕ_i a function of only the variables X_i in its adjacency set. A factor graph $F = \{\mathcal{U}, \mathcal{V}, \mathcal{E}\}$ then defines the factorisation of a function $\phi(X)$ as

$$\phi(X) = \prod_i \phi_i(X_i). \tag{4}$$

We can represent the unnormalised posterior with a factor graph by letting each factor ϕ_i correspond to either the prior factors $p(x_j)$ or the measurement factors $l(X_k;\, z_k)$. MAP inference then simply comes down to maximising the product of all factor graph potentials:

$$X^* = \arg\max_X \prod_i \phi_i(X_i). \tag{5}$$

If we assume Gaussian priors and likelihood functions derived from measurements corrupted by zero-mean normally distributed noise, all factors in eq. (5) are on the form

$$\phi_i(X_i) \propto \exp\left(-\frac{1}{2}\|h_i(X_i) - z_i\|^2_{\mathbf{\Sigma}_i}\right), \tag{6}$$

where $\|\mathbf{e}\|^2_{\mathbf{\Sigma}} \triangleq \mathbf{e}^\top \mathbf{\Sigma}^{-1} \mathbf{e}$ denotes the squared Mahalanobis distance over the residual errors $\mathbf{e}$ with covariance matrix $\mathbf{\Sigma}$. By taking the negative log of eq. (5) and dropping the constant factor, we can instead minimise a sum of nonlinear least-squares:

$$X^* = \arg\min_X \sum_i \|h_i(X_i) - z_i\|^2_{\mathbf{\Sigma}_i}. \tag{7}$$

MAP inference in this situation is therefore equivalent to a nonlinear least-squares problem, which can be solved with iterative methods such as Gauss-Newton or Levenberg-Marquardt. These methods iteratively solve a linear approximation to the nonlinear problem by linearising about the current estimate in each step.

We can linearise each of the residual errors $e_i(X_i) = h_i(X_i) - z_i$ with a first order Taylor expansion

$$\begin{aligned} e_i(X_i) &= e_i(X_i^0 + \Delta_i) \\ &\approx e_i(X_i^0) + \mathbf{J}^{e_i}_{X_i}(X_i^0)\Delta_i, \end{aligned} \tag{8}$$

where $\mathbf{J}^{e_i}_{X_i}$ is the Jacobian of e_i w.r.t. X_i, and

$$\Delta_i \triangleq X_i - X_i^0. \tag{9}$$

By substituting eq. (8) into eq. (7), we obtain a *linear* least squares problem in the state update vector Δ:

$$\begin{aligned}
\Delta^* &= \arg\min_{\Delta} \sum_i \|\mathbf{J}^{e_i}_{X_i}(X_i^0)\Delta_i + e_i(X_i^0)\|^2_{\mathbf{\Sigma}_i} \\
&= \arg\min_{\Delta} \sum_i \|\mathbf{J}^{h_i}_{X_i}(X_i^0)\Delta_i - (z_i - h_i(X_i))\|^2_{\mathbf{\Sigma}_i} \\
&= \arg\min_{\Delta} \sum_i \|\mathbf{\Sigma}_i^{-\frac{1}{2}}\mathbf{J}^{h_i}_{X_i}(X_i^0)\Delta_i - \mathbf{\Sigma}_i^{-\frac{1}{2}}(z_i - h_i(X_i))\|^2_2 \\
&= \arg\min_{\Delta} \sum_i \|\mathbf{A}_i\Delta_i - \mathbf{b}_i\|^2_2 \\
&= \arg\min_{\Delta} \|\mathbf{A}\Delta - \mathbf{b}\|^2_2,
\end{aligned} \tag{10}$$

where we have appropriately organised all $\mathbf{A}_i$ and $\mathbf{b}_i$ into a large matrix $\mathbf{A}$ and vector $\mathbf{b}$, respectively. The solution to eq. (10) can be found using standard linear least-squares solvers.

Some of the benefits of representing a MAP estimation problem using a factor graph is that it makes the problem simple to express, visualise and extend with new measurements and variables. Furthermore, since the independence relationships are explicitly modelled, sparsity and conditional independencies can be directly exploited to optimise and solve the problem efficiently.[10]

2.2 Manifold Representations of Rotations and Directions

We will in the following consider two special kinds of state variables: Camera orientations, which we will represent as rotation matrices $\mathbf{R} \in SO(3)$, and landmark directions, which we will represent as direction vectors $\mathbf{d} \in S^2$. These variables are not defined on vector spaces, but live on smooth manifolds in higher dimensional space. In order to apply the estimation framework above on these manifolds, we take the standard approach of working in the tangent space to the manifold at the current estimate, which locally behaves as a Euclidean space.

The *special orthogonal group* in 3D is the set of valid rotation matrices

$$SO(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3\times 3} \;\middle|\; \mathbf{R}\mathbf{R}^\top = \mathbf{I}, \det \mathbf{R} = 1 \right\}, \tag{11}$$

which is a *Lie group* closed under matrix multiplication, with identity $\mathbf{I}$ and inverse $\mathbf{R}^{-1} = \mathbf{R}^\top$. *Lie theory* allows us to map exactly the tangent space to/from the manifold using the *exponential* and *logarithmic* maps. We will use the *capitalised* exponential and logarithmic map notation, which maps tangent space vectors $\boldsymbol{\theta} \in \mathbb{R}^3$ directly to rotation matrices $\mathbf{R} \in SO(3)$ via the tangent space at the identity, and vice versa. These are given by

$$\mathrm{Exp} : \mathbb{R}^3 \to SO(3); \qquad \mathbf{R} = \mathrm{Exp}(\boldsymbol{\theta}) \tag{12}$$

$$\mathrm{Log} : SO(3) \to \mathbb{R}^3; \qquad \boldsymbol{\theta} = \mathrm{Log}(\mathbf{R}). \tag{13}$$

We can work in the tangent space at the current estimate $\hat{\mathbf{R}} \in SO(3)$ by composing the current estimate with the maps at identity. For convenience we express these compositions using the plus and minus operators

$$\mathbf{R} = \hat{\mathbf{R}} \oplus \boldsymbol{\theta} \triangleq \hat{\mathbf{R}}\,\mathrm{Exp}(\boldsymbol{\theta}) \quad \in SO(3) \tag{14}$$

$$\boldsymbol{\theta} = \mathbf{R} \ominus \hat{\mathbf{R}} \triangleq \mathrm{Log}(\hat{\mathbf{R}}^\top \mathbf{R}) \in \mathbb{R}^3. \tag{15}$$

The plus operator lets us increment a rotation matrix $\hat{\mathbf{R}}$ with a tangent space vector $\boldsymbol{\theta}$, while the minus operator lets us compute the corresponding tangent space vector between the two rotation matrices $\hat{\mathbf{R}}$ and $\mathbf{R}$. These operators can also be used to define derivatives on the manifold. We refer to Solà et al.[11] for further details.

Directions lie as points on the unit sphere S^2, which is a manifold defined as

$$S^2 = \left\{ \mathbf{d} \in \mathbb{R}^3 \;\middle|\; \|\mathbf{d}\| = 1 \right\}. \tag{16}$$

Since S^2 is not a Lie group, the theory in the previous paragraph does not apply for directions. It is however possible to define a tangent vector space in the local tangent plane at a point on S^2. We follow this procedure as outlined in Dellaert et al.[9] to define an efficient mapping, which enables us to define plus and minus operators for S^2 similar to the ones above.

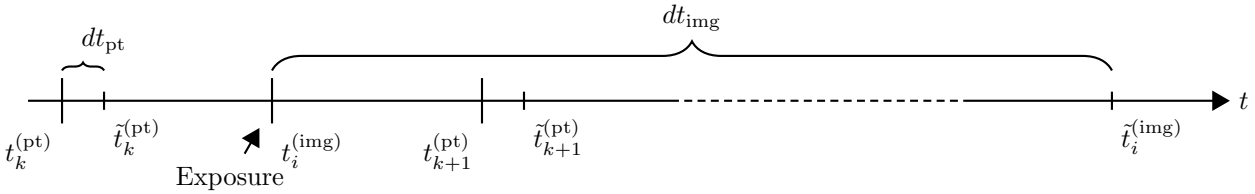

Figure 3: Conceptual timeline for acquisition of pan/tilt measurements and images in a pan-tilt camera. In low-cost systems, we expect that pan/tilt measurements acquired at intervals $t_k^{(\text{pt})}$ have their timestamps recorded at a slightly later time $\tilde{t}_k^{(\text{pt})}$. Some such systems record the image timestamps after the image has passed through the video encoder. Other low-cost systems do not provide image timestamps at all, in which case the time of reception on the client computer may be the only viable timestamp. In either case we expect the timestamp $\tilde{t}_i^{(\text{img})}$ to be recorded significantly later than the time of exposure $t_i^{(\text{img})}$. We denote the time-offsets between the timestamps and the actual measurements dt_{pt} and dt_{img}.

2.3 A Clock Model for Pan-Tilt Cameras

As demonstrated in fig. 1, a PTZ camera can exhibit aggravated pointing accuracy during manoeuvres if the internal clocks in the system are not properly synchronised. Figure 3 shows one possible way to model this. We consider both pan-tilt measurements and images to be acquired at given points of time $t_k^{(\text{pt})}$ and $t_i^{(\text{img})}$ along a "true" timeline, but due to shortcomings in the system design the *reported* timestamps are recorded at some other point of time $\tilde{t}_k^{(\text{pt})}$ and $\tilde{t}_i^{(\text{img})}$ along the same timeline. We are interested in the offsets between the time of measurement and the recorded timestamps

$$\begin{aligned} dt_{\text{pt}} &= \tilde{t}_k^{(\text{pt})} - t_k^{(\text{pt})} \\ dt_{\text{img}} &= \tilde{t}_i^{(\text{img})} - t_i^{(\text{img})}. \end{aligned} \tag{17}$$

In most cases dt_{pt} should be relatively small, and is not possible to observe without some external (accurately timed) device which measures the direction in which the platform is pointing. dt_{img}, on the other hand, can vary greatly depending on how the imaging pipeline is set up. For typical computer vision cameras designed to provide accurate timestamps, dt_{img} should be less than 10μs. For low-cost cameras, however, dt_{img} can easily be as large as 100ms[‡], which is significant when the camera is moving.

Although this "true" timeline is useful as a mental model, it must be defined in terms of some synchronised external clock, which low-cost PTZ cameras do not provide. However, we only need our clock model to be able to translate between pan-tilt timestamps and image timestamps. So instead of working with dt_{pt} and dt_{img} directly, we shall consider the relative offset between pan/tilt timestamps and image timestamps

$$dt = dt_{\text{img}} - dt_{\text{pt}}. \tag{18}$$

This relative offset is observable as discrepancies between the observed motion of the camera in the images and the motion corresponding to the pan/tilt measurements.

2.4 Continuous-Time Representations of Discrete-Time Pan-Tilt Measurements

Unless our PTZ is constructed with special hardware that triggers a pan/tilt measurement at each exposure, we need a model that can predict the camera orientation measurement at the time of exposure. This model associates each image, via its timestamp $t^{(\text{img})}$, with a predicted camera orientation measurement. In order to be able to evaluate the model for any offset between the image timestamps and pan/tilt measurement timestamps, we follow

[‡]Many low-cost cameras do not really provide timestamps per frame at all. In this case the NTP timestamps in the RTCP stream is often the best source of high resolution timing. These timestamps are often set *after* the frame has been encoded, adding a significant delay.

the method proposed by Furgale et al.[12] which uses the discrete-time pan/tilt measurements to parameterise a continuous-time measurement function. In the interest of simplicity, we use piecewise linear interpolation of the measurements.

Pan/tilt measurements directly observe the orientation of the *camera coordinate frame* $\mathcal{F}_c$ relative to the *platform coordinate frame* $\mathcal{F}_p$. The platform frame $\mathcal{F}_p$ defines the nominal directions of the base of the PTZ camera with a right-down-forward coordinate system. The camera frame $\mathcal{F}_c$ has its origin at the optical centre of the camera, which is also aligned with the origin of the platform frame. The x-axis points to the right, the y-axis points down and the z-axis is aligned with the optical axis of the camera.

Although it is possible to interpolate the pan/tilt measurements directly, we instead convert them to rotations $\mathbf{R}_{pc} \in SO(3)$ and perform the interpolations on the manifold. In this paper we use the notation $\mathbf{R}_{ab}$ for a rotation matrix rotating from frame $\mathcal{F}_a$ to $\mathcal{F}_b$. The primary benefit of this is that it avoids wraparound-issues in the implementation. We can convert the pan/tilt measurements to rotations using principal rotations about the involved axes. In our case with right-down-front axes an orientation with pan θ and tilt ϕ becomes

$$\mathbf{R}_{pc} = \mathbf{R}_y(\theta)\mathbf{R}_x(\phi). \tag{19}$$

In the remainder of the paper we will adopt the notation $\mathbf{R}^{(\mathrm{pt})} = \mathbf{R}_{cp} = \mathbf{R}_{pc}^\top$ for any rotation that stems from pan/tilt *measurements*, and use $\mathbf{R}_i$ to denote rotation *variables* in the factor graph. In all cases it is a rotation that maps points from the platform frame to the camera frame.

We will now order pan/tilt measurements by their timestamp t_i, and convert them to orientations $\mathbf{R}_i^{(\mathrm{pt})}$. The piecewise linear interpolation providing the continuous-time representation of the orientation measurements is then given as

$$\begin{aligned}\mathbf{R}^{(\mathrm{pt})}(t) \triangleq \mathbf{R}_i^{(\mathrm{pt})} \oplus \left(\frac{t - t_i}{t_{i+1} - t_i}(\mathbf{R}_{i+1}^{(\mathrm{pt})} \ominus \mathbf{R}_i^{(\mathrm{pt})})\right) \\ \text{where } i \text{ i.s.t. } t \in [t_i, t_{i+1}).\end{aligned} \tag{20}$$

2.5 Camera Model

The camera model describes the relationship between the direction of a landmark $\mathbf{d} \in S^2$ and its observed pixel position $\mathbf{u} \in \mathbb{R}^2$ in the image. Our landmarks are static directions $\mathbf{d}^p$ in the platform frame $\mathcal{F}_p$, which correspond to directions $\mathbf{d}^c = \mathbf{R}_{cp}\mathbf{d}^p$ in the camera frame $\mathcal{F}_c$. For the intrinsic camera model we will use a standard perspective camera model with quadratic radial distortion. We denote the projection of a landmark direction $\mathbf{d}^c$ in the camera frame to pixel coordinates $\mathbf{u} \in \mathbb{R}^2$ by

$$\mathbf{u} = \pi(\mathbf{d}^c; f_u, f_v, c_u, c_v, k). \tag{21}$$

This mapping first projects the direction $\mathbf{d}^c$ to the corresponding point on the (undistorted) normalised image plane:

$$\mathbf{x}^u = \frac{1}{d_3^c}\begin{bmatrix} d_1^c \\ d_2^c \end{bmatrix} \tag{22}$$

This point is then radially distorted as

$$\mathbf{x} = \mathbf{x}^u \left(1 + k \left\|\mathbf{x}^u\right\|_2^2\right), \tag{23}$$

and finally mapped to image pixels

$$\mathbf{u} = \begin{bmatrix} f_u x_1 + c_u \\ f_v x_2 + c_v \end{bmatrix}. \tag{24}$$

Some existing methods for calibrating PTZ cameras model the focal lengths and distortion parameter as functions of a measured zoom-parameter z.[2] As discussed in section 1, we are targeting cameras where we do not expect the zoom measurements to be sufficiently consistent to be usable. Instead, our plan is to re-estimate a fixed set of intrinsic parameters each time we change the zoom level during operation.

For spherical images the principal point (c_u, c_v) is not observable under pan/tilt manoeuvres, as panning and tilting in this case is equivalent to translating the principal point. For narrow FOVs, planar images are approximately spherical, and in datasets with narrow manoeuvre swaths, this approximate equivalence can make our optimisation problem approximately underdetermined. For this reason we will exclude c_u and c_v from the state variable set, and rather use their nominal values as constants.

2.6 Constructing the Factor Graph

All that remains in order to define the factor graph that represents our MAP problem is to construct the factors that express the probabilistic constraints imposed by our measurements. Since we have two types of measurements, we will have to create two types of factors: $\phi^{(\text{proj})}$ representing landmark observations, and $\phi^{(\text{pt})}$ representing the continuous-time pan/tilt measurements. Sections 2.6.1 and 2.6.2 define the measurement models and error functions needed to create these factors. We can now construct the factor graph as shown in fig. 4, with the following contents:

- A variable $dt \in \mathbb{R}$, representing the clock offset.
- A variable $C = \left(f_u, f_v, k_1\right) \in \mathbb{R}^3$, representing the intrinsic camera parameters.
- Variables $\mathbf{R}_i \in SO(3)$ for each image used in the estimation, representing the orientation of the camera.
- Variables $\mathbf{d}_j \in S^2$ for each landmark track observed, representing the direction of the landmark.
- Factors $\phi_i^{(\text{pt})}$ for each image, connected to dt and the corresponding $\mathbf{R}_i$. This factor contains the image timestamp $t_i^{(\text{img})}$, and encodes the information of the pan/tilt measurements.
- Factors $\phi_{ij}^{(\text{proj})}$ for each observation of a landmark in every image, connected to C and the corresponding $\mathbf{R}_i$ and $\mathbf{d}_j$. This factor contains the measured position $\mathbf{u}_{ij}$ of the landmark $\mathbf{d}_j$ in image i.

2.6.1 Modelling landmark observations

Since we have the orientation $\mathbf{R}_i$ for each image as a variable in the graph, creating a factor for the landmark observations is essentially the same as in standard VO/SLAM formulations that include camera calibration. The only minor difference is the use of orientations instead of poses, and directions instead of positions for landmarks. Thanks to the notation we have introduced in the previous sections, this difference disappears as an implementation detail.

We will use the following measurement model for an observation $\mathbf{u}_{ij}$ of a landmark with direction $\mathbf{d}_j$ in an image taken with camera orientation $\mathbf{R}_i$ and calibration C:

$$\mathbf{u}_{ij} = \pi(\mathbf{R}_i \mathbf{d}_j;\, C) + \boldsymbol{\eta}_{ij},\, \boldsymbol{\eta}_{ij} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{ij}) \tag{25}$$

The covariance matrix $\boldsymbol{\Sigma}_{ij}$ represents the measurement noise of the observation $\mathbf{u}_{ij}$, and depends on the camera resolution and keypoint detection method used.

We define our projection factor as

$$\begin{aligned} \phi_{ij}^{(\text{proj})}(\mathbf{R}_i, \mathbf{d}_j, C; \mathbf{u}_{ij}) &\triangleq \left\| \pi(\mathbf{R}_i \mathbf{d}_j;\, C) - \mathbf{u}_{ij} \right\|^2_{\boldsymbol{\Sigma}_{ij}} \\ &= \left\| \mathbf{e}_{ij}^{(\text{proj})}(\mathbf{R}_i, \mathbf{d}_j, C) \right\|^2_{\boldsymbol{\Sigma}_{ij}} \end{aligned} \tag{26}$$

The Jacobians of $\mathbf{e}_{ij}^{(\text{proj})}$ with respect to each of the variables are thus

$$\begin{aligned} \mathbf{J}_{\mathbf{R}_i}^{\mathbf{e}_{ij}^{(\text{proj})}} &= \mathbf{J}_{\mathbf{R}_i \mathbf{d}_j}^{\pi} \mathbf{J}_{\mathbf{R}_i}^{\mathbf{R}_i \mathbf{d}_j} \\ \mathbf{J}_{\mathbf{d}_j}^{\mathbf{e}_{ij}^{(\text{proj})}} &= \mathbf{J}_{\mathbf{R}_i \mathbf{d}_j}^{\pi} \mathbf{J}_{\mathbf{d}_j}^{\mathbf{R}_i \mathbf{d}_j} \\ \mathbf{J}_{C}^{\mathbf{e}_{ij}^{(\text{proj})}} &= \mathbf{J}_{C}^{\pi}. \end{aligned} \tag{27}$$

Where $J_{\mathbf{R}_i \mathbf{d}_j}^{\pi}$ is the Jacobian of π w.r.t. the direction $\mathbf{d}_j^c = \mathbf{R}_i \mathbf{d}_j$ in the camera frame, and J_C^{π} is the Jacobian of π w.r.t. the intrinsic parameters C.

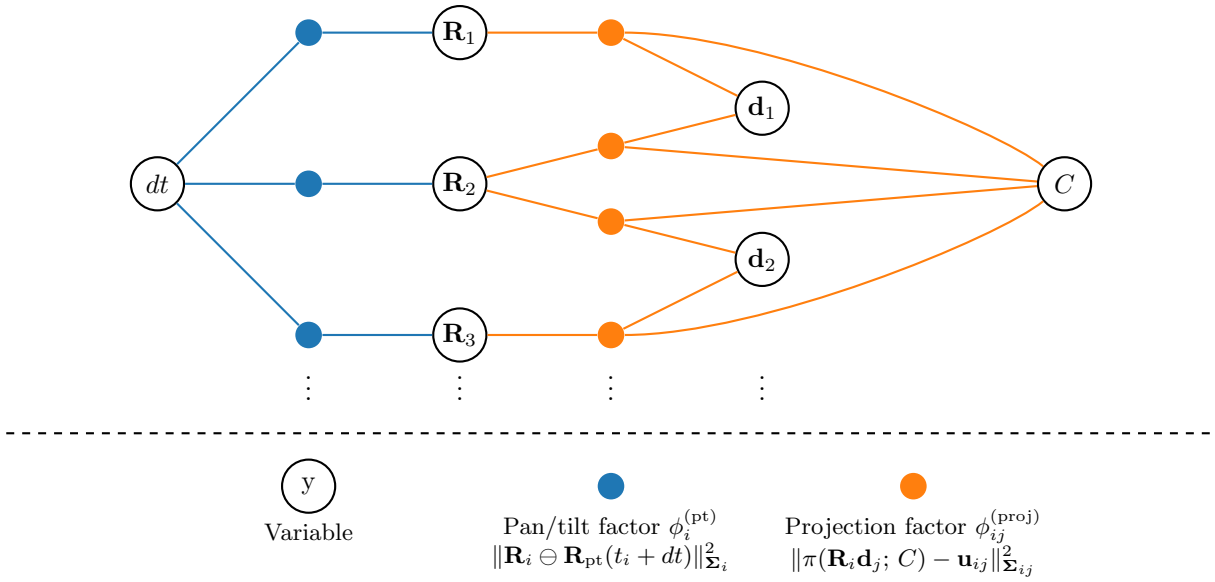

Figure 4: Overview of the factor graph representing our estimation problem. The graph contains a single common variable for the clock offset $dt \in \mathbb{R}$ and single common variable for the intrinsic parameters $C = \{f_u, f_v, k_1\} \in \mathbb{R}^3$. For each image we estimate a variable for the camera orientation $\mathbf{R}_i \in SO(3)$, and for each landmark track we estimate the landmark direction $\mathbf{d}_j \in S^2$. Pan/tilt measurements correspond to factors $\phi_i^{(\mathrm{pt})}$ (blue) on $\mathbf{R}_i$ and dt, while landmark observations adds factors $\phi_{ij}^{(\mathrm{proj})}$ (orange) on the corresponding $\mathbf{R}_i$, $\mathbf{d}_j$ and C.

2.6.2 Modelling orientation measurements

We define the probabilistic *measurement model* for the continuous-time orientation measurements $\mathbf{R}^{(\mathrm{pt})}(t^{(\mathrm{pt})})$ only at discrete timesteps $t_i^{(\mathrm{img})} = t_i^{(\mathrm{pt})} - dt$ where an image with camera orientation $\mathbf{R}_i$ was taken. We assume the measurements to be generated as

$$\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})} + dt) = \mathbf{R}_i + \boldsymbol{\eta}_i,\ \boldsymbol{\eta}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_i), \tag{28}$$

where the covariance matrix $\mathbf{\Sigma}_i$ represents the measurement noise.

We can then define our factor as:

$$\begin{aligned} \phi_i^{(\mathrm{pt})}(\mathbf{R}_i, dt; \mathbf{R}^{(\mathrm{pt})}, t_i^{(\mathrm{img})}) &\triangleq \left\| \mathbf{R}_i \ominus \mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})} + dt) \right\|^2_{\mathbf{\Sigma}_i} \\ &= \left\| \mathbf{e}_i^{(\mathrm{pt})}(\mathbf{R}_i, dt) \right\|^2_{\mathbf{\Sigma}_i} \end{aligned} \tag{29}$$

The Jacobians of $\mathbf{e}_i^{(\mathrm{pt})}$ with respect to the variables are then given as:

$$\begin{aligned} \mathbf{J}^{\mathbf{e}_i^{(\mathrm{pt})}}_{\mathbf{R}_i} &= \mathbf{J}^{\mathbf{R}_i \ominus \mathbf{R}_{\mathrm{pt}}}_{\mathbf{R}_i} \\ \mathbf{J}^{\mathbf{e}_i^{(\mathrm{pt})}}_{dt} &= \mathbf{J}^{\mathbf{R}_i \ominus \mathbf{R}_{\mathrm{pt}}}_{\mathbf{R}_{\mathrm{pt}}} \mathbf{J}^{\mathbf{R}_{\mathrm{pt}}}_{dt}, \end{aligned} \tag{30}$$

where $\mathbf{J}^{\mathbf{R}^{(\mathrm{pt})}}_{dt}$ follows from eq. (20) with $\boldsymbol{\theta} = \mathbf{R}_{i+1}^{(\mathrm{pt})} \ominus \mathbf{R}_i^{(\mathrm{pt})}$:

$$\mathbf{J}^{\mathbf{R}_{\mathrm{pt}}}_{dt} = \frac{1}{t_{i+1} - t_i} \mathbf{J}^{\mathbf{R}_i^{(\mathrm{pt})} \oplus \boldsymbol{\theta}}_{\boldsymbol{\theta}} \boldsymbol{\theta} \tag{31}$$

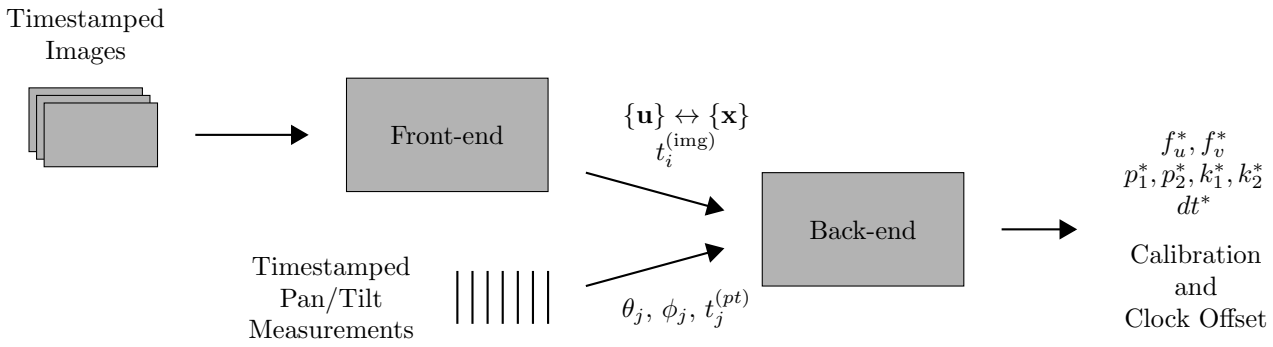

Figure 5: Overview of the calibration method. Timestamped images are fed through the front-end module, which manages a set of landmark tracks and extracts observations from each new image. The timestamped observations are fed alongside timestamped pan/tilt measurements to the back-end, which estimates the clock-offset via full bundle adjustment on the observations and measurements.

3. IMPLEMENTATION

This section aims to add the missing pieces required to turn the theory described in section 2 into a procedure that can be used in the field during camera operation. The MAP estimation procedure described in section 2.1 is the primary focus of this paper. Therefore we have separated our method into a *front-end* component and a *back-end* component, as shown in fig. 5. The front-end is responsible for producing observations of landmarks from each image, while the back-end is responsible for estimating the parameters. The separation allows us to easily to a better frontend without making changes to the backend, and is somewhat analogous to how current SLAM methods are organised.[8]

3.1 The System Front-end – Extracting Landmark Observations

Since the estimation method in the back-end is the main focus of this paper, we attempt to keep the front-end as simple as possible. We have chosen to detect and track *corner keypoints*, which should be a good solution for tracking landmarks in static and highly structured scenes. For the keypoint tracking itself, we employ the OpenCV[13] implementation of the pyramidal version of the Lucas-Kanade tracking algorithm due to Bouguet.[14] New keypoint tracks are initialised using the Shi-Tomasi corner detector,[15] which is also implemented in OpenCV. The front-end iterates through the images in the dataset in the order they where taken, maintaining a fixed number of keypoint tracks from one image to the next. As we move along, any keypoint that disappears from view is replaced with a new Shi-Tomasi corner from the current image. The result is a set of keypoint tracks that span two or more images. In the interest of avoiding drift, we want to have at least some keypoint tracks that span most of the dataset. Due to the simplistic keypoint tracking used, this means that the camera must be manoeuvred in a way that keeps a section of the scene visible in all the images used.

3.2 System Back-end – Finding the Optimal Parameters

The back-end is implemented by constructing the factor graph described in section 2.6 using the factor graph library GTSAM,[16] and then optimising it using Gauss-Newton. GTSAM provides implementations for the $\oplus$ and $\ominus$ operators for both rotations[§] and directions[¶]. It also has implementations of the camera projection function π[‖]. The Jacobians of the various operators needed are also implemented in GTSAM, and can easily be assembled as described in eqs. (27) and (30).

Optimising the graph with Gauss-Newton provides MAP estimates for all state variables, as well as first-order estimates of their covariance. The estimates for $\mathbf{R}_i$ are more accurate than the $\mathbf{R}^{(\text{pt})}$ measurements

[§]See gtsam::Rot3 in the GTSAM docs.
[¶]See gtsam::Unit3 in the GTSAM docs.
[‖]See gtsam::PinholeCamera and gtsam::Cal3DS2 in the GTSAM docs.

themselves, since the estimates incorporate information from the landmark observations as well. In our case, the high-accuracy orientation estimates are primarily interesting for analysing the pointing accuracy of the pan/tilt measurements. However, in a real-time implementation the $\mathbf{R}_i$ estimates should be the preferred source of image orientations for downstream processing.

4. EXPERIMENTAL RESULTS

Experiments were conducted by applying the method to datasets recorded using an Axis Q6215-LE, which is an uncalibrated low-cost camera (∼$3000) without special timing hardware. Two datasets were recorded: One for calibration, and one for validation. Most of the analysis is conducted on the calibration dataset, as this contains more data, whereas the validation set is primarily meant to reveal shortcomings of this analysis.

4.1 Experiment Setup

The Axis Q6215-LE has a rolling shutter camera capturing 1920x1080 images at 50fps, and a nominal *preset accuracy* of 0.1°. The accuracy of the pan/tilt *measurements* is not specified, but appears to be much better. The camera provides no API-support for extracting per-frame timestamps, so this requires a little extra work.

4.1.1 Acquisition of timestamped pan/tilt measurements

The easiest way to obtain pan/tilt measurements from the Q6215-LE is through the HTTP-based VAPIX API.[17] These measurements are not timestamped, and HTTP introduces too much latency and jitter for client-side timestamping to be usable. Instead we use the ACAP API[**18] to write a driver that is run onboard the PTZ, and transmits timestamped pan/tilt measurements over UDP to the client computer. The ACAP API provides a method for reading timestamped pan/tilt measurements. At the time of writing this paper, this method had a bug which caused it to return incorrect timestamps. Therefore, we instead use our on-board driver to poll the pan/tilt position at 100hz, and send the measured pan/tilt position whenever it changes. The timestamp is recorded when the measurement is received from the ACAP API, which means that these timestamps are accurate only on the order of ±5ms.

4.1.2 Acquisition of timestamped images

We access the images as an RTSP session with the video encoded as H.264 and transmitted using RTP[††] over UDP. Each frame in the H.264 stream has a presentation timestamp (PTS), which gives the time elapsed since the start of the stream with a resolution of $1/90000s$. The association between PTS timestamps and NTP timestamps in the RTCP[‡‡] sender-reports is then used to transform the PTS timestamps to epoch timestamps, which can then be compared with pan/tilt timestamps. The PTS/NTP association in the sender reports happens on the camera side, and is outside of our control. Normally, this association is established *after* the encoder step of the image pipeline. This means that we should expect the epoch timestamps to have a bias on the size of the duration from a frame is exposed until it has been encoded. Any jitter on the duration of the exposure-to-encoder pipeline will also propagate to our epoch timestamps, but since the encoder is implemented in hardware, we assume this jitter not to be larger than 1ms. The benefit of this method is that the relative timestamps of two images are fairly accurate, and any clock-drift or acquisition hick-ups in the camera are automatically corrected. To our knowledge no other bias-free or more accurate method for acquiring per-frame timestamps from this camera exists. For many other low-cost PTZ cameras the image timestamps must instead be set on the client computer, introducing additional network or bus latency, which can easily bring the total clock bias up to and beyond 200ms.

[**]The Axis ACAP API is a C-API for writing applications designed to run on-board the cameras.
[††]The Real-time Transport Protocol is a commonly used streaming protocol with low latency.
[‡‡]The RTP Control Protocol (RTCP) is commonly used to control the RTP stream of an RTSP session.

Table 1: Estimated values of dt, f_u, f_v and k, along with the approximated stddev. of each estimate.

	Estimate	Approx. stddev. of estimate
dt	-39.2ms	1.7ms
f_u	47365	73.4
f_v	46533	100.2
k	17.4	0.13

4.2 Calibration Dataset

We recorded a dataset for calibration consisting of a list of timestamped pan/tilt measurements and a list of timestamped images, using the techniques described above. The weather conditions during the recording were close to ideal: Good lighting conditions with the sun behind the camera, essentially no wind or mirage, and no visible movement in vegetation. The camera was kept at the maximum zoom setting during the recording, which corresponds to a nominal FOV of 2.2°x1.2°. In order to keep a section of the scene present in all images, we manoeuvred the camera in a circular motion with a diameter of approximately 0.5°. We also attempted to minimise the effects of rolling shutter by manoeuvring slowly, completing three full circles in about 22s. During the circular motion, the camera is being directed at a residential area approximately 550m away. The images were recorded at 16fps, for a total of 350 images.

4.3 Estimating Parameters using the Calibration Dataset

We evaluate the full calibration dataset of 350 images with the method running on a 2016 HP ZBook G3 with an 8-thread Intel i7-6820hq CPU. Operating with 60 simultaneous keypoint tracks, the front-end uses on average 28.58ms per frame. Over 10 runs the back-end used on average 786.1ms to process the whole dataset with 350 images and a total of 127 keypoint tracks. This means that the back-end processing takes on average 2.2ms per frame, which is a strong indication that a real-time implementation using iSAM2[10] is feasible.

Table 1 shows the estimated parameters, along with the approximated stddev. of the estimates. The estimate of dt indicates that the image timestamps are *delayed* with 39ms, which is in line with our assumptions about how the RTSP timestamps are set. Similarly, the FOV corresponding to the estimates of f_u and f_v is 2.32°x1.32°, which is plausible given the the nominal values. The estimate of $k = 17.4$ may seem high, but keep in mind that this is a camera with a narrow FOV, which means that radius in the normalised image plane $\|\mathbf{x}\|_2^2$ is very small even in the corners of the image. The approximated stddev. of the estimates might seem to indicate that the estimates are accurate with 2-3 significant figures. This approximation is intimately connected to the model used in the estimation problem, and any modelling error might greatly affect the accuracy of this approximation. If the approximated stddev. were on the scale as the estimates themselves, however, this would be a strong indication that the estimation problem is underdetermined.

4.4 Analysis of Performance on Calibration Dataset

A visual inspection of the estimation results is shown in fig. 6, where we compare different projections of directions that are static in the platform frame. In the video we see that the projections made using measured orientations have an observable jitter. The uncalibrated projections exhibit a clear circular motion as well, which to a significant extent seems to have been removed in the synchronised and calibrated projections. The projections made using the estimated orientations seem to be almost perfectly fixed to the background.

One way to quantify how well the estimated model fits the landmark observations used in the optimisation is to evaluate the average projection error of the estimated landmark directions. This quantity sets the lower bound for the expected pointing accuracy in the pixel plane that is achievable with the current model, and can be computed as:

$$\frac{1}{n}\sum_i \sum_{j\in\mathcal{O}(i)} \left\|\pi(\mathbf{R}_i^*\mathbf{d}_j^*, f_u^*, f_v^*, k^*) - \mathbf{u}_{ij}\right\|_2, \tag{32}$$

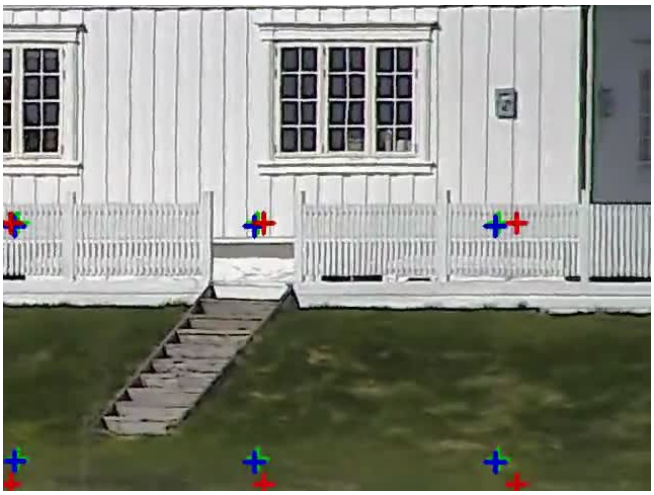

Figure 6: Video 1. Comparison of the effects of time-synchronisation and calibration. The crosses show static directions in the platform frame projected into the image. If the orientations and calibration are correct, the crosses should appear fixed to the background. Red crosses are projected using nominal calibration $f_u^{(\mathrm{nom})}, f_u^{(\mathrm{nom})}, k = 0$ and unsynchronised (raw) measured orientations $\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})})$. Estimated calibration f_u^*, f_u^*, k^* and synchronised measured orientations $\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})} + dt^*)$ is used to project the blue crosses. The green crosses are projected using estimated calibration and estimated orientations $\mathbf{R}_i^*$ i. The video plays at 3X speed and has been cropped to better illustrate the effect of orientation noise and biases in the cases with measured orientations.
http://dx.doi.org/10.1117/12.2573611.1

where i indexes the images in the dataset, $\mathcal{O}_i$ is the index set of the landmark observations in image i, and the state variables written with a superscripted asterisk are the estimated values. We can get an impression of the effect of the calibration and the time-synchronisation separately by evaluating similar averages using nominal calibration and measured orientations. Table 2 shows the average landmark projection errors for combinations of orientation and calibration sources. As we can see, both the time-synchronised measured orientations and the estimated calibration seem to give a considerable boost in projection accuracy. Here we must also keep in mind that the error due to lack of clock synchronisation is proportional to the angular velocity of the camera, and that we are manoeuvring very slowly in this dataset. Simultaneously, sub-pixel accuracy seems to be within reach when using the estimated orientations. This is consistent with our observation in fig. 6 were the $\mathbf{R}_i^*$-based projections appeared fixed against the background. However, the landmark observations are correlated with our estimates, which means that these averages cannot be viewed as a verification of our optimisation, nor do they give a proper estimate of the projection accuracy.

Table 2: The mean projection error for various combinations of orientation and calibrations sources.

Orientations	Calibration	Mean projection error
$\mathbf{R}_i^*$	estimated	0.822px
$\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})} + dt^*)$	estimated	3.28px
$\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})})$	estimated	5.72px
$\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})} + dt^*)$	nominal	14.3px
$\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})})$	nominal	15.2px

Using the calibration dataset, it is also possible to compare the measured and the estimated orientations via synchronised timestamps. Assuming that the orientations can be estimated with much greater accuracy

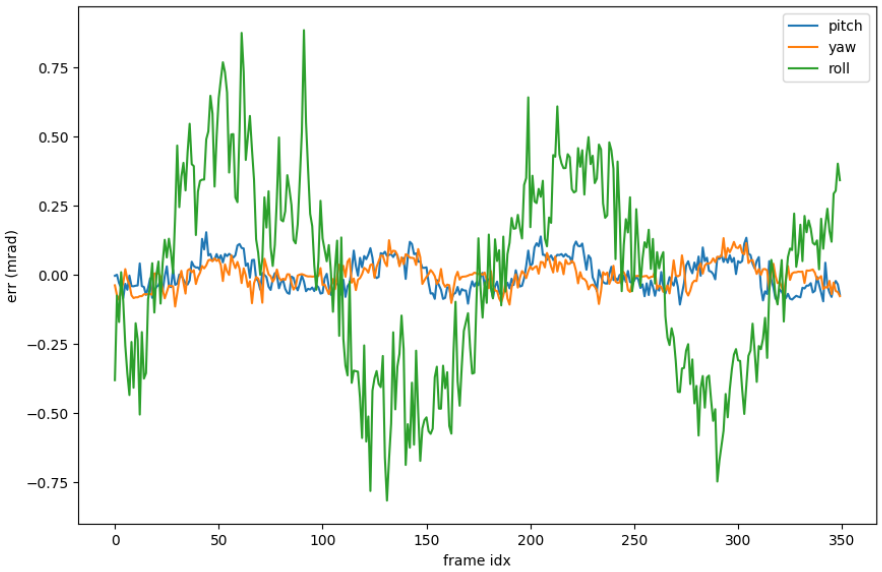

Figure 7: The difference in roll, pitch and yaw between the measured and estimated orientations of each image in the calibration dataset. The estimated orientations are here obtained in an optimisation run with relaxed constraints on the pt-factors. This difference is an estimate of the error in the pan/tilt measurements, which should resemble white noise if the model was correct. The measured orientations have 0 roll, so roll-error seen here is therefore equal to the estimated roll in the unconstrained optimisation. The RMS for the yaw- and pitch error is 0.0458mrad and 0.0571mrad, respectively.

from the landmark observations, the difference between these estimates and the measured orientations might reveal systematic errors in the pan/tilt measurements. Figure 7 shows a plot of these differences, which clearly exhibit systematic errors. The sinusoidal pattern hints that these errors are correlated with either the positions or velocity of the PTZ camera. Plotting the roll error against the PTZ yaw rate, as shown in fig. 8, exposes that the latter is certainly the case. This bias might be caused by mechanical lash in the pan/tilt mechanism. Alternatively, the bias may be due to the effects of the rolling shutter, causing the images to appear rotated as we change the pan/tilt speed. It is also possible that the roll is (wrongfully) introduced in the interpolation in eq. (20). Despite the obvious shortcomings of our model, we observe that the RMS on the yaw and pitch error is close to 0.05mrad, which is impressive for a camera in this price range.

4.5 Evaluation on an Independent Dataset

In order to provide some form of external validation that is not directly connected to the calibration data, we recorded a small dataset where the PTZ is first kept stationary, before panning mildly to the left. Similarly to fig. 6, we can compare projections of a fixed direction using raw orientations $\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})})$ to projections using synchronised orientations $\mathbf{R}^{(\mathrm{pt})}(t_i^{(\mathrm{img})} + dt^*)$. Figure 9 shows two such comparisons when using the estimated calibration. By manual inspection of the two frames shown in the figure, we see that the raw orientations exhibit a projection error of more than 100px while the projection based on synchronised orientations is at least within 5px. Overall, the video shows that the synchronised orientations seem to fit the camera motion significantly better than the raw measurements, but still with room for improvement. The error observed for the unsynchronised projections is also much larger than for the calibration dataset, in which the camera was manoeuvring slowly.

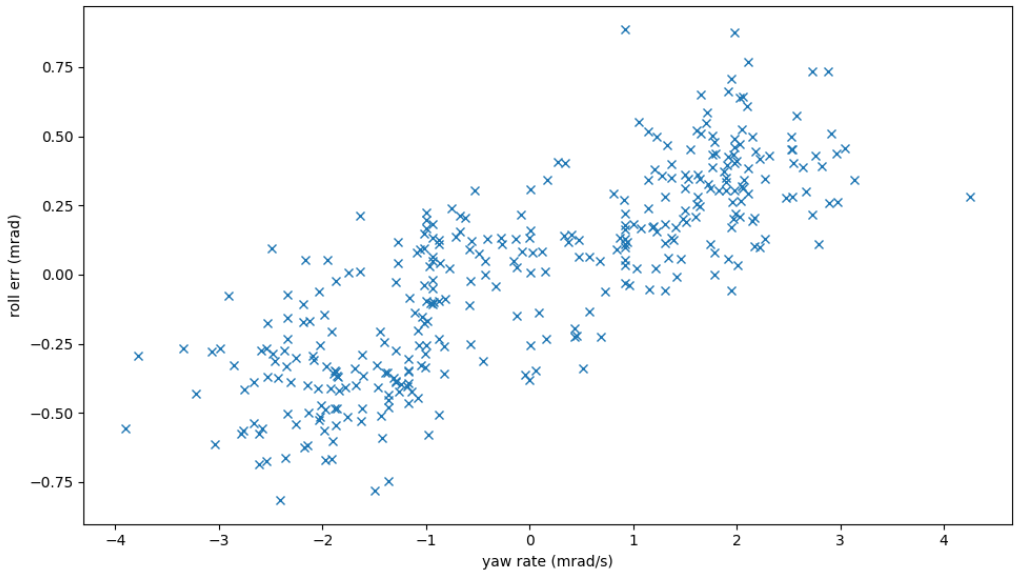

Figure 8: The roll error from fig. 7 plotted against the yaw rate of the PTZ camera. Here we see evident correlation between the two.

5. DISCUSSION AND CONCLUSIONS

In this paper we have demonstrated the feasibility of jointly estimating intrinsic calibration and clock synchronisation for a PTZ camera using only data that can be acquired in the field during normal operation. By using a factor graph to model the estimation problem, this is a promising starting point for more advanced methods that can also incorporate other measurements, such as measurements from an inertial navigation system (INS). Through experiments we have observed that the calibration and clock synchronisation can significantly improve the overall directional accuracy of a low-cost system.

Although this first implementation is not designed to be used in real-time on a per-image basis, the running times observed on the test datasets suggest that such an implementation using incremental factor graph optimisation is within reach. This seems particularly enticing if paired with a lightweight front-end, which can help bring the total processing time per frame further down. By also incorporating zoom measurements, a real-time method might also be used to provide intrinsic calibration in scenarios where the zoom is changed continuously. Using a front-end capable of performing loop closures by recognising previously disappeared landmarks would also hugely beneficial, as it would enable calibration with wide swaths. The results with estimated orientations indicate that sub-pixel directional accuracy is achievable, and serve as an extra motivation for realising such a per-image implementation.

More evaluation must be conducted to verify the effectiveness of the method. The accuracy of the estimates, and whether the constant-bias clock model is valid remain open research questions. We should be able to evaluate the overall accuracy of the method through simulations, but model verification requires testing with real-world cameras. In our datasets we see clear indications of unmodelled effects which might be caused by the rolling shutter or other mechanisms in the PTZ camera. If rolling shutter indeed causes the observed roll, it means that the shutter line duration is potentially observable and can be estimated by extending our model.

In conclusion, the proposed method shows promising results towards achieving pixel-level absolute directional accuracy in low-cost PTZ cameras. Through exploiting the rotation-only motion and direction-only observability of such cameras, the proposed method is less susceptible to noise and converges faster than existing methods

Figure 9: Video 2. Projections of a static landmark when the camera begins panning to the left. The red cross shows a static landmark projected using unsynchronised (raw) orientation measurements. The blue cross shows the same static landmark, but projected using the orientation measurements synchronised using the estimated dt from table 1. The first frame is taken while the camera is stationary, in which case the two projections coincide. The second frame is taken just after the camera has begun panning. Here we see that the unsynchronised projection (red) significantly overestimates how far the camera has panned. See the linked video for the full sequence. http://dx.doi.org/10.1117/12.2573611.2

with larger optimisation spaces. The joint estimation of calibration and time synchronisation can potentially provide more accurate and less biased estimates than estimating them separately, but this question requires further investigation.

Acknowledgements

This research was partially funded by The University Graduate Center at Kjeller (UNIK).

REFERENCES

[1] Sinha, S. N. and Pollefeys, M., "Pan-tilt-zoom camera calibration and high-resolution mosaic generation," *Computer Vision and Image Understanding* **103**(3), 170–183 (2006).

[2] Wu, Z. and Radke, R. J., "Keeping a pan-tilt-zoom camera calibrated," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**(8), 1994–2007 (2013).

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

[4] Cannelle, B., Paparoditis, N., and Tournaire, O., "Panorama-Based Camera Calibration," *Pcv* **XXXVIII**, 1–6 (2010).

[5] Frank, A., "Short papers," *History of Economics Society Bulletin* **6**(2), 16–25 (1985).

[6] Szeliski, R. and Shum, H. Y., "Creating full view panoramic image mosaics and environment maps," *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1997* , 251–258 (1997).

[7] Scaramuzza, D. and Fraundorfer, F., "Tutorial: Visual odometry," *IEEE Robotics and Automation Magazine* **18**(4), 80–92 (2011).

[8] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., and Leonard, J. J., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics* **32**(6), 1309–1332 (2016).

[9] Dellaert, F. and Kaess, M., "Factor Graphs for Robot Perception," *Foundations and Trends in Robotics* **6**(1-2), 1–139 (2017).

[10] Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., and Dellaert, F., "ISAM2: Incremental smoothing and mapping using the Bayes tree," *International Journal of Robotics Research* **31**(2), 216–235 (2012).
[11] Solà, J., Deray, J., and Atchuthan, D., "A micro Lie theory for state estimation in robotics," 1–17 (2018).
[12] Furgale, P., Tong, C. H., Barfoot, T. D., and Sibley, G., "Continuous-time batch trajectory estimation using temporal basis functions," *International Journal of Robotics Research* **34**(14), 1688–1710 (2015).
[13] Bradski, G., "The OpenCV Library," *Dr. Dobb's Journal of Software Tools* (2000).
[14] Bouguet, J.-y., "Pyramidal implementation of the Lucas Kanade feature tracker," *Intel Corporation, Microprocessor Research Labs* (2000).
[15] Shi, J. and Tomasi, C., "Good Features," *Image (Rochester, N.Y.)* , 593–600 (1994).
[16] Dellaert, F., "GTSAM." http://gtsam.org.
[17] "Axis ACAP API." https://www.axis.com/support/developer-support/axis-camera-application-platform.
[18] "Axis VAPIX API." https://www.axis.com/vapix-library/.