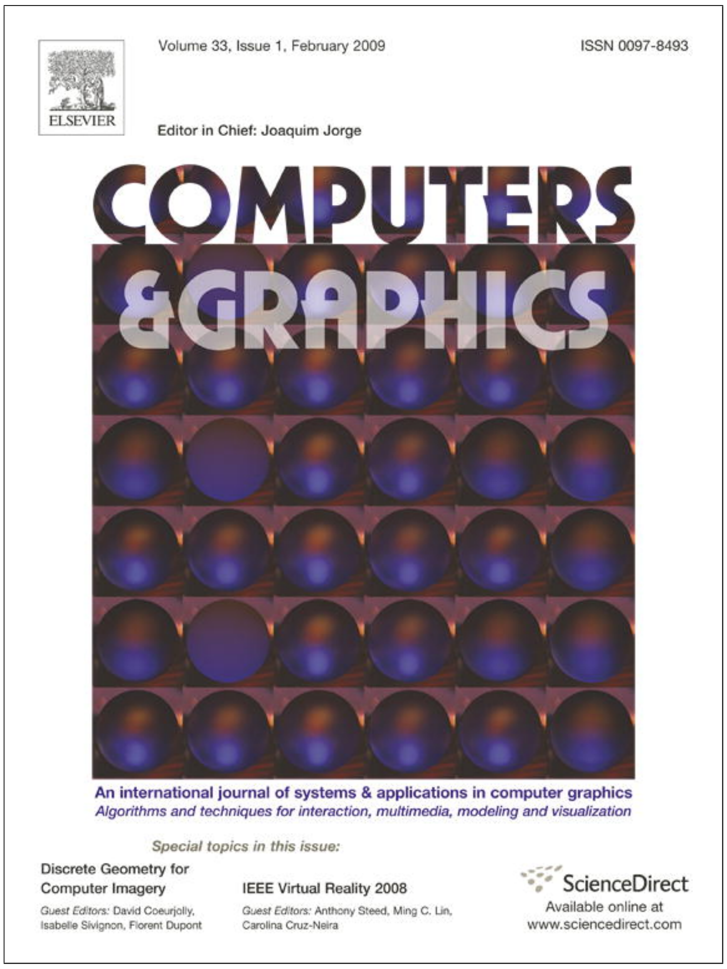
Volume 33, Issue 1, February 2009

ISSN 0097-8493

Editor in Chief: Joaquim Jorge

COMPUTERS & GRAPHICS

An international journal of systems & applications in computer graphics

Algorithms and techniques for interaction, multimedia, modeling and visualization

Special topics in this issue:

Discrete Geometry for Computer Imagery

Guest Editors: David Coeurjolly, Isabelle Sivignon, Florent Dupont

IEEE Virtual Reality 2008

Guest Editors: Anthony Steed, Ming C. Lin, Carolina Cruz-Neira

ScienceDirect
Available online at
www.sciencedirect.com

Computers & Graphics 33 (2009) 59–72

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Technical Section

Advanced tracking through efficient image processing and visual–inertial sensor fusion ☆

Gabriele Bleser *, Didier Stricker

Department of Virtual and Augmented Reality, Fraunhofer IGD, Fraunhoferstr. 5, D-64283 Darmstadt, Germany

A R T I C L E I N F O

Article history:
Received 13 June 2008
Received in revised form
2 October 2008
Accepted 10 November 2008

Keywords:
Augmented reality
Markerless model-based camera tracking
Inertial sensors
Sensor fusion
Extended Kalman filter

A B S T R A C T

This article presents a new visual–inertial tracking device for augmented and virtual reality applications and addresses two fundamental issues of such systems. The first one concerns the definition and modelling of the sensor fusion problem. Much work has been conducted in this area and several models for exploiting gyroscopes and linear accelerometers have been proposed. However, the respective advantages of each model and in particular the benefits of the integration of the accelerometer data in the filter are still unclear. A comparison of different models with special investigation of the effects of using accelerometers on the tracking performance is therefore provided. The second contribution is the development of an image processing approach that does not require special landmarks but uses natural features. The solution relies on a 3D model of the scene that is used to predict the appearances of the features by rendering the model based on data from the sensor fusion algorithm. The feature localisation is robust and accurate mainly because local lighting is also estimated. The final system is evaluated with help of ground-truth and real data. High stability and accuracy are demonstrated also for large environments.

1. Introduction

User tracking is an enabling technology for augmented reality and is also crucial for immersive virtual reality. If a camera is rigidly attached to the user, e.g. to a head-mounted display (HMD), vision-based tracking methods can be applied. As pure vision-based systems usually handle only relatively slow motions, the fusion of vision-based and inertial tracking technology has been proposed several times in the past. Inertial sensors (gyroscopes and accelerometers) are suited for capturing fast movements, while the slower vision sensor provides absolute references to compensate for errors accumulating rapidly when integrating noisy and biased inertial data (dead reckoning). General information about the error characteristics of inertial sensors are given in [1].

While it has been shown often enough in the past that inertial measurements improve pure vision-based tracking in several ways [2–6], it is still unclear how to best exploit the information in the inertial data. In particular, to incorporate accelerometer data into the estimate of the translational states is often argued to introduce instabilities rather than support. This is due to the fact that accelerometers measure not only free acceleration but also acceleration due to gravity. The gravity component has to be subtracted using the estimated orientation before double-integrating to position. Hence, small errors in the orientation result in unbounded position errors. Accelerometers are therefore often only used to stabilise the camera attitude with respect to gravity (roll and tilt angle)—rather than estimating the attitude and the translational states—by modelling the free acceleration as noise [7]. This is also the common operating mode of the commercially available orientation trackers, which internally fuse measurements from accelerometers, gyroscopes and magnetometers and deliver absolute orientation (XSens MTi/MTx, InterSense InertiaCube). In [5] such a device is used to reinitialise the natural feature registration, if the track has been lost by the vision sensor.

Because of the special nature of the accelerometer measurements, many researchers use only gyroscopes to support vision-based tracking. In [8] a particle filter is used to fuse gyroscope measurements and point-based vision tracking. During the particle generation step the rotation angles are sampled according to the measured angular velocities, while a random walk is applied to the translation. In most cases the fusion is done in an extended Kalman filter (EKF). In [4] an EKF is used to combine gyroscopes with a line-based vision tracking system. The linear velocities are filtered from previous vision-based position estimates. The final camera pose is computed solely from the vision

☆ Revised and extended version of the conference paper with the same title presented at the IEEE Virtual Reality 2008 (Reno, Nevada, March 2008).

* Corresponding author. Tel.: +49 631 205 75 356.
E-mail addresses: Gabriele.Bleser@dfki.de (G. Bleser), Didier.Stricker@dfki.de (D. Stricker).

0097-8493/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2008.11.004

measurements and is then used as an EKF measurement to estimate the gyroscope biases. In [9] gyroscope data and vision measurements from fiducial tracking are fused in an EKF. The 2D/3D correspondences obtained from the vision sensor are directly passed as measurements to the EKF without an intermediate vision-based pose computation step as in [4,7]. In [10] this approach is extended to outdoor augmented reality by replacing the fiducial tracking with a system based on natural line features.

Some systems have also been proposed, which exploit gyroscopes and accelerometers for the estimation of both rotational and translational states. Most of them use an EKF to fuse all measurements uniformly to a pose estimate. The most popular example is the commercially available InterSense VIS-Tracker [11], which is based on artificial markers. A theoretical framework for combining such a system with markerless line- and point-based tracking technology is provided in [12,13], respectively. However, both works base their experiments on simulated data. In [14] the model given in [13] is applied to realistic inertial data and simulated vision measurements, whereas the biases of the inertial sensors are estimated online as parts of the state vector. Some recent visual–inertial SLAM (simultaneous localisation and mapping) systems provide experimental results on realistic data, but within simple test environments [6,15].

The system described here uses an EKF to combine model-based tracking of natural point features with inertial sensors. The sensor fusion core is able to estimate the camera pose, velocities, accelerations and sensor biases by processing data obtained from gyroscopes (angular velocities), accelerometers (linear accelerations) and the vision system (2D/3D correspondences). In this contribution, an in-depth evaluation of different sensor fusion models with a special investigation of the effects of using accelerometers on the tracking performance is provided. The experiments are performed under controlled conditions and in real world environments and prove the benefits of fully exploiting the accelerometers for estimating both the rotational and the translational states.

Natural feature detection and tracking are essential research areas for augmented reality. Many works focus on the development of invariant and robust feature detectors, descriptors and alignment methods, suitable for wide baseline matching [16–19]. In the system described here the problem of feature association is simplified, as a prediction of the camera pose is always available. A technique, which exploits the pose prediction in a very elegant manner, is to predict the appearances of natural features by rendering a 3D model of the environment. This is sometimes referred to as analysis-by-synthesis. In [4] a 3D wireframe model of the target scene is rendered from the predicted camera pose and a 1D search for local gradient maxima is carried out orthogonally to the rendered edges at equally distributed sample points. In [20] this approach is enhanced by learning a visual appearance state for each sample point during the tracking, thus achieving increased discrimination. Instead of learning the appearances of the edge sample points from the images, a textured CAD model is used in [7].

The vision system proposed here also uses a simplified textured CAD model of the environment, but in contrast tracks feature points instead of edges. The contribution in this area is to develop a technique for registering small patches of a rendered image in a live camera frame also under severe changes of illumination. This is mainly achieved by estimating not only 2D displacements but also local lighting changes using an advanced Kanade–Lucas feature tracker [21]. The analysis-by-synthesis technique, that is the technique to use a CAD model for the registration has many advantages: (1) the graphics hardware can be used to efficiently generate a rendering from the predicted pose, (2) the correct level of detail is guaranteed and (3) the feature tracking is drift-free.

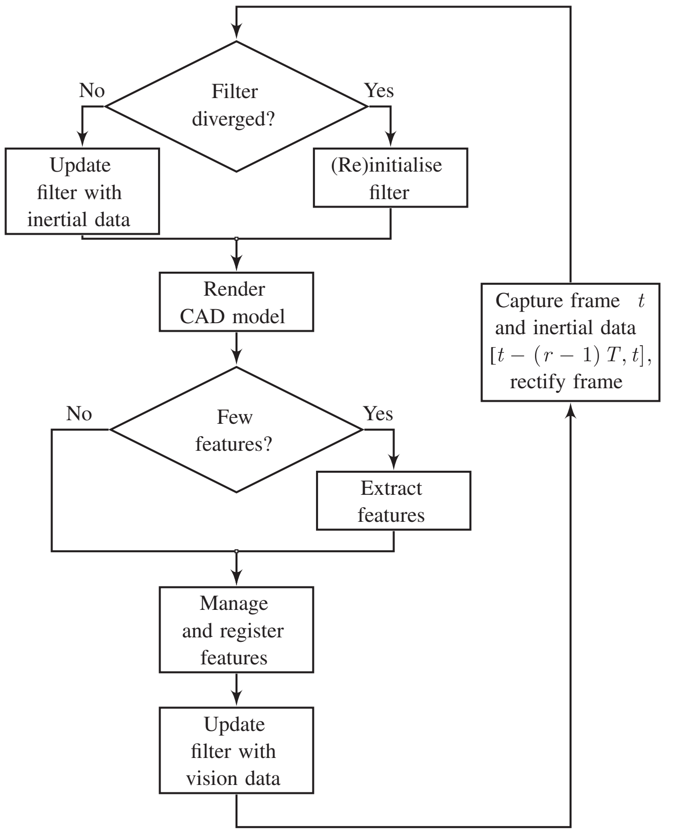
Filter diverged?
No
Yes
Update filter with inertial data
(Re)initialise filter
Render CAD model
Capture frame t and inertial data $[t-(r-1)T, t]$, rectify frame
Few features?
No
Yes
Extract features
Manage and register features
Update filter with vision data

Fig. 1. System workflow.

2. Approach

The sensors providing physical inputs to the system, i.e. the camera and the Inertial Measurement Unit (IMU), are assumed to be synchronised in hardware and to run at different rates.[1] Fig. 1 outlines the system workflow. A frame is captured at timestep t and the IMU readings from the previous time interval $[t-(r-1)T, t]$ are buffered. T is the sample rate of the IMU and rT is the sample rate of the camera. The image is resampled to compensate for distortion effects based on the model of [22]. A divergence test is then performed on the Kalman filter state, based on which the system either (re)initialises or processes the measured angular velocities and accelerations up to timestep t.

The (re)initialisation is semi-automatic. A rough orientation is obtained from the IMU and it serves as starting point for the orientation states in the EKF. The XSens MT9-C used for the experiments delivers besides the measured angular velocities and accelerations also an absolute orientation (cf. Section 5). The initial position is entered manually and the user has to move the camera close enough to this position so that the vision-based tracking snaps on.

Both (re)initialisation and filter update yield a pose prediction for timestep t, which is then used to generate a rendering of the textured CAD model on the graphics card. On demand new features are extracted in free areas of the rendered image using FAST (features from accelerated segment test) [23]. They are

[1] In the experiments, the IMU runs at 100 Hz and provides a trigger signal to the camera at 25 Hz.

stored with their 3D positions in a feature map, which is managed by a set of simple rules. Details are given in [24]. At each timestep, a fixed-size subset of features predicted to be visible in the given frame is registered. The 2D/3D correspondences are then used as vision measurements to update the EKF.

This article is organised as follows. Section 3 describes the natural feature tracking. Section 4 details the fusion of visual and inertial measurements presenting four fusion models: model 1 (gyro) only uses gyroscopes to support the camera orientation estimate, model 2 (gravity) uses additional accelerometers to stabilise the camera attitude and models 3 (acc) and 4 (acc input) exploit the accelerometers also for estimating the translational states. Models 3 and 4 differ by treating the inertial data either as measurements or as control inputs. Both approaches have been proposed in the past [25,26]. Moreover, outlier rejection and divergence monitoring are addressed. Section 5 investigates the tracking performance and the computational efficiency of the different state-space models demonstrating that model 4 performs best with respect to both criteria. The high stability and accuracy achieved by fully exploiting the accelerometer data for both the estimation of the rotational and the translational states, is shown in different real environments. Section 6 draws final conclusions.

3. Natural feature tracking

A natural feature point is tracked by cropping a small window of its surrounding texture from a rendering of the textured CAD model and registering it in the live camera image. The position and perspective distortion of the feature in the rendering can be assumed very similar to those in the live camera image, a suited condition for estimating a 2D displacement using an iterative registration method like the Kanade–Lucas tracker [21]. However, due to changing lighting conditions and the photometric response function of the camera used, the entire appearance of the feature in the rendering can differ significantly from that in the live frame. An example is given in Fig. 2(a,b). These effects can be compensated for by using the relatively simple photometric model proposed in [27].

Let $\mathbf{m}_p = [x, y]^{\mathrm{T}}$ be a feature position in an image. Let $I(\mathbf{m}_p)$ be the intensity value of this position in the rendering and $J(\mathbf{m}_p)$ the intensity value in the live camera image. We describe the appearance change of a feature from the rendering to the live camera image by

$$I(\mathbf{m}_p) = \lambda J(\mathbf{m}_p + \mathbf{d}) + \delta \quad \forall \mathbf{m}_p \in W. \tag{1}$$

$\mathbf{d} = [d_1, d_2]^{\mathrm{T}}$ is a 2D displacement vector, λ can be regarded a parameter adjusting the contrast and δ as a parameter adjusting the brightness of a region W around the feature point. The registration consists in finding the parameter vector $\alpha = [d_1, d_2, \lambda, \delta]^{\mathrm{T}}$ that minimises

$$\varepsilon = \sum_W (I(\mathbf{m}_p) - \lambda J(\mathbf{m}_p + \mathbf{d}) + \delta)^2. \tag{2}$$

Substituting $\lambda J((\mathbf{m}_p + \mathbf{d}) + \delta$ with its first-order Taylor expansion and setting the derivatives of (2) to 0 with respect to α results in a linear system $G\alpha = \mathbf{b}$, from which α can be computed. Newton–Raphson-style iterations of this procedure are performed until either a maximum number of iterations is exceeded or the change in α becomes neglectable. To speed up the convergence, the illumination parameters λ and δ are initialised with the mean and standard deviation of the texture around the predicted feature position in the rendering and the live image, respectively. A registration is assumed valid, if the sum of squared differences (SSD) over the registered feature windows is below a threshold.

In order to increase the convergence radius of the registration, a coarse-to-fine strategy is applied over an image pyramid. This enables the system to (re)initialise from a very rough initial pose, and to continue the tracking, if the pose prediction has drifted after some frames without features visible.

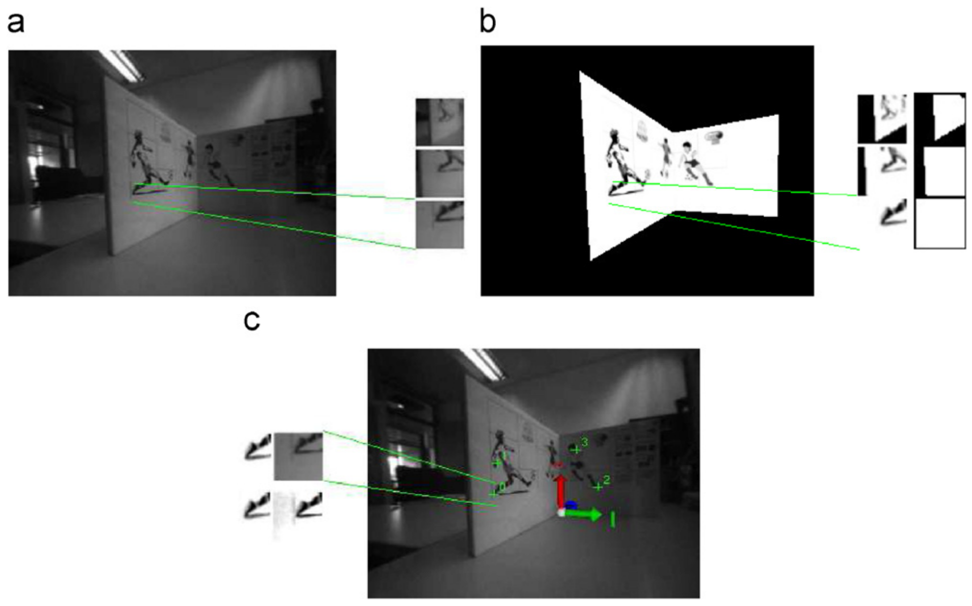

Fig. 2. The feature registration process over an image pyramid: (a) shows a live camera image of a simple target object and texture patches cropped around a feature position from three levels of the image pyramid. (b) shows the rendering, the patch pyramid and the binary masks. The upper two levels of the patch pyramid contain parts of the black background, which would disturb the registration. The masks on the right side mark exactly these parts as invalid. The registration therefore succeeds. (c) shows the texture patches from the rendered and live image (bottom level of image pyramid) in comparison. The top row demonstrates the differing appearances of the patches before the registration. After the registration, both patches look quite similar. In the image on the right side the four registered feature positions are marked, which have been used to compute the camera pose for the augmentation.

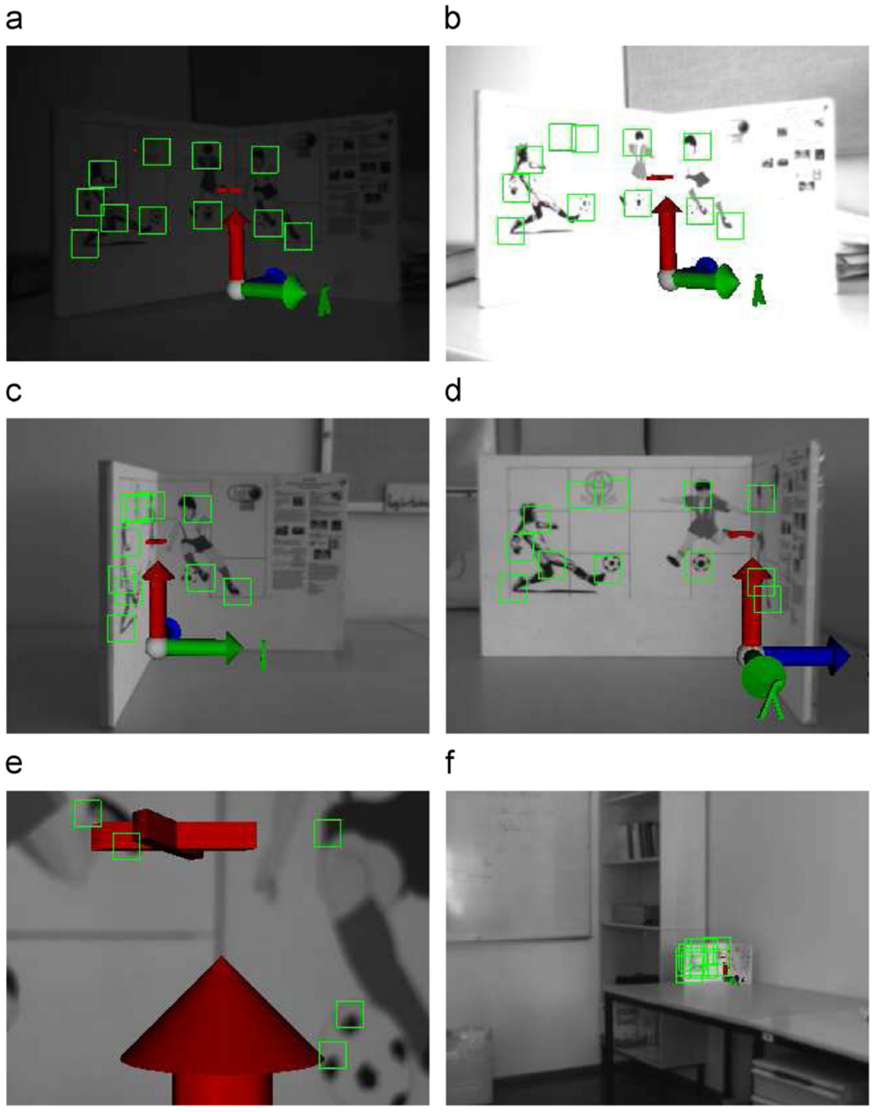

Fig. 3. Augmented camera images showing the successful feature registration (indicated by the green rectangles) and pose estimation (indicated by the pose of the augmented coordinate system) under different illuminations and under varying viewing angles and distances with respect to the target object: (a,b) show under- and overexposed camera images, (c,d) present nearly orthogonal views onto the sides of the target object and (e,f) show the target object from significantly different viewing distances of 10 and 350 cm, respectively.

To make the feature registration as robust as possible, two other issues, that can arise from the rendering process, are handled: If the CAD model used for generating the rendering has holes or does not fill the entire image, the OpenGL background colour appears and disturbs the registration. As the information of pixel validity is easily available from the depth buffer, a binary mask is generated for the feature window in the rendered image and is used during the registration for masking out invalid pixels (Fig. 2(b)). A second issue concerns the aliasing effects introduced by the rendering process. In order to reduce those and to stabilise the registration, the synthetic image is blurred. The feature registration process is outlined in Fig. 2.

Fig. 3 demonstrates the potential of the natural feature tracking as presented here. The affine photometric model given in (1) allows for tracking the features under significant changes of illumination. Moreover, the technique of rendering a CAD model for predicting the feature appearances enables their successful registration from a variety of different view points, whereas the tracking remains drift-free.

4. Sensor fusion

The EKF [28–30] is used for fusing the 2D/3D correspondences from the image analysis and the inertial measurements from the IMU to a pose estimate. For the sake of completeness, the general EKF equations are given here. Let

$$\mathbf{x}_t = f(\mathbf{x}_{t-T}, \mathbf{u}_t, \mathbf{v}_t), \tag{3a}$$
$$\mathbf{y}_t = h(\mathbf{x}_t, \mathbf{e}_t) \tag{3b}$$

be a nonlinear state-space model, where $\mathbf{x}_t$ denotes the state vector, $\mathbf{u}_t$ denotes a known control input, $\mathbf{v}_t$ denotes the process noise, with $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, Q_t)$, $\mathbf{y}_t$ denotes a measurement and e_t denotes the measurement noise, with $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, R_t)$. Let $\hat{\mathbf{x}}_t$ be the estimate of $\mathbf{x}_t$ at time t with $\mathbf{x}_t \sim \mathcal{N}(\hat{\mathbf{x}}_t, P_t)$. The equations for the time update are

$$\hat{\mathbf{x}}_{t|t-T} = f(\hat{\mathbf{x}}_{t-T|t-T}, \mathbf{u}_t, \mathbf{0}), \tag{4a}$$
$$P_{t|t-T} = F_t P_{t-T|t-T} F_t^{\mathrm{T}} + V_t Q_t V_t^{\mathrm{T}}, \tag{4b}$$

with

$$F_t = \frac{\partial f}{\partial \mathbf{x}}(\hat{\mathbf{x}}_{t-T|t-T}, \mathbf{u}_t, \mathbf{0}), \tag{4c}$$

$$V_t = \frac{\partial f}{\partial \mathbf{v}}(\hat{\mathbf{x}}_{t-T|t-T}, \mathbf{u}_t, \mathbf{0}). \tag{4d}$$

The equations for the measurement update are

$$S_t = H_t P_{t|t-T} H_t^{\mathrm{T}} + E_t R_t E_t^{\mathrm{T}}, \tag{5a}$$

$$\mathbf{z}_t = \mathbf{y}_t - h(\hat{\mathbf{x}}_{t|t-T}, \mathbf{0}), \tag{5b}$$

$$K_t = P_{t|t-T} H_t^{\mathrm{T}} S_t^{-1}, \tag{5c}$$

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-T} + K_t \mathbf{z}_t, \tag{5d}$$

$$P_{t|t} = P_{t|t-T} - K_t H_t P_{t|t-T}, \tag{5e}$$

with

$$H_t = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\hat{\mathbf{x}}_{t|t-T}, \mathbf{0}), \tag{5f}$$

$$E_t = \frac{\partial \mathbf{h}}{\partial \mathbf{e}}(\hat{\mathbf{x}}_{t|t-T}, \mathbf{0}). \tag{5g}$$

S_t is the innovation covariance, $\mathbf{z}_t$ the innovation and K_t the Kalman filter gain.

After introducing the different coordinate systems involved and the notation used subsequently, the four different state-space models announced in Section 2 are presented.

4.1. Notation

The following coordinate systems are used: the *world frame*, **w** (fixed to the target scene model), the *camera frame*, **c** (fixed to the moving camera), the *global frame*, **g** (fixed to earth with the x-axis pointing to local magnetic north and the z-axis pointing opposite gravity), the *sensor frame*, **s** (fixed to the moving IMU), the *pixel coordinate system*, **p** (assumed ideal, as distortion effects are compensated for before the feature registration) and the *normalised image frame*, **n**, which is obtained from **p** with

$$\mathbf{m}_n = \begin{bmatrix} \frac{1}{f_x} & \frac{-s}{f_x f_y} & \frac{-c_x f_y + c_y s}{f_x f_y} \\ 0 & \frac{1}{f_y} & \frac{-c_y}{f_y} \end{bmatrix} \cdot \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}. \tag{6}$$

f_x, f_y, c_x, c_y and s are the internal camera parameters. The global frame is the inertial reference frame of the IMU in the same way as the world frame is the reference frame of the camera. The IMU provides an estimate of its absolute orientation $\mathbf{q}_{gs}$ with respect to this frame. The reference frame, in which a quantity is resolved, is indicated by subscribing the abbreviation introduced above. The abbreviation is also used for indicating the origin of a reference frame, e.g. $\mathbf{s}_c$ is the origin of the IMU, **s**, given in the camera frame c. Transformation subscripts contain two letters denoting the mapping. Unit quaternions are used to parametrise rotations, for instance $\mathbf{q}_{sc}$ describes the rotation from the camera frame c to the IMU frame s. The corresponding rotation matrix is termed Q_{sc}. The quaternion product is denoted $\odot$. See [31] for more information on quaternions and the conversion formulae. Fig. 4 illustrates the 3D coordinate systems and transformations used throughout this article.

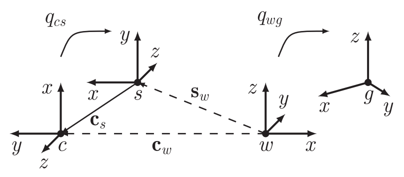

Fig. 4. Illustration of the different 3D coordinate systems and how they are related. Rigid transformations are indicated by solid lines, non-rigid by dashed lines.

4.2. Model 1 (gyro)

Gyroscopes are used to support the vision-based tracking. The state vector **x** comprises

$$\mathbf{x}^{\mathrm{T}} = [\mathbf{s}_w^{\mathrm{T}} \; \dot{\mathbf{s}}_w^{\mathrm{T}} \; \mathbf{q}_{sw}^{\mathrm{T}} \; \boldsymbol{\omega}_s^{\mathrm{T}} \; \mathbf{b}_s^{\omega\mathrm{T}}], \tag{7}$$

where $\mathbf{s}_w$ denotes the position, $\dot{\mathbf{s}}_w$ the velocity, $\mathbf{q}_{sw}$ the orientation, $\boldsymbol{\omega}_s$ the angular velocities and $\mathbf{b}_s^{\omega}$ the gyroscope biases of the IMU.

Note that to simplify the equations the translation $\mathbf{s}_w$ and orientation $\mathbf{q}_{sw}$ of the IMU with respect to the world frame are estimated. However, the camera pose is easily obtained from the state vector using

$$\mathbf{c}_w = (\mathbf{s}_w + Q_{ws}\mathbf{c}_s), \tag{8a}$$

$$\mathbf{q}_{cw} = \mathbf{q}_{cs} \odot \mathbf{q}_{sw}, \tag{8b}$$

where $\mathbf{q}_{cs}$ and $\mathbf{c}_s$ are the hand-eye rotation and translation between the rigidly coupled camera and IMU (cf. Fig. 4). The hand-eye transformation is calibrated once offline (cf. Section 5). The motion model assumes constant velocity and constant angular velocity. The expression for the time update is

$$\begin{bmatrix} \mathbf{s}_{w,t} \\ \dot{\mathbf{s}}_{w,t} \\ \mathbf{q}_{sw,t} \\ \boldsymbol{\omega}_{s,t} \\ \mathbf{b}_{s,t}^{\omega} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t-T} + T\dot{\mathbf{s}}_{w,t-T} + \frac{T^2}{2}\mathbf{v}_{w,t}^{\ddot{\mathbf{s}}} \\ \dot{\mathbf{s}}_{w,t-T} + T\mathbf{v}_{w,t}^{\ddot{\mathbf{s}}} \\ \exp\left(-\frac{T}{2}(\boldsymbol{\omega}_{s,t-T} + \mathbf{v}_{s,t}^{\boldsymbol{\omega}})\right) \odot \mathbf{q}_{sw,t-T} \\ \boldsymbol{\omega}_{s,t-T} + \mathbf{v}_{s,t}^{\boldsymbol{\omega}} \\ \mathbf{b}_{s,t-T}^{\omega} + \mathbf{v}_{s,t}^{\mathbf{b}^{\omega}} \end{bmatrix}, \tag{9}$$

where $\mathbf{v}_{w,t}^{\ddot{\mathbf{s}}}$, $\mathbf{v}_{s,t}^{\boldsymbol{\omega}}$ and $\mathbf{v}_{s,t}^{\mathbf{b}^{\omega}}$ denote the time independent process noise and

$$\exp(\mathbf{v})^{\mathrm{T}} = \left[\cos\|\mathbf{v}\| \quad \frac{\mathbf{v}^{\mathrm{T}}}{\|\mathbf{v}\|}\sin\|\mathbf{v}\|\right] \tag{10}$$

is the quaternion exponential. The noise is assumed uncorrelated in all components. The gyroscope measurement model is

$$\mathbf{y}_{s,t}^{\omega} = \boldsymbol{\omega}_{s,t} + \mathbf{b}_{s,t}^{\omega} + \mathbf{e}_{s,t}^{\omega}, \tag{11}$$

where $\mathbf{y}_{s,t}^{\omega}$ are the measured angular velocities and $\mathbf{e}_{s,t}^{\omega}$ denotes the measurement noise, which is assumed time independent.

The image analysis initially provides a set of 2D/3D correspondences $(\mathbf{m}_{p,t}, \mathbf{m}_{w,t})$ with measurement noises $\mathbf{e}_{p,t}^{c} \sim \mathcal{N}(\mathbf{0}, R_{pp,t})$ and $\mathbf{e}_{w,t}^{c} \sim \mathcal{N}(\mathbf{0}, R_{ww,t})$, where $R_{pp,t}$ and $R_{ww,t}$ are diagonal matrices. The measurement $\mathbf{m}_{p,t}$ and the covariance $R_{pp,t}$ are first transformed to the normalised image coordinate system using (6), giving $\mathbf{c}_t = (\mathbf{m}_{n,t}, \mathbf{m}_{w,t})$ with covariances $R_{nn,t}$ and $R_{ww,t}$. The implicit correspondence measurement model is then given by

$$\begin{aligned} \mathbf{0} &= h(\mathbf{x}_t, \mathbf{m}_{n,t}, \mathbf{m}_{w,t}, \mathbf{e}_{n,t}^{c}, \mathbf{e}_{w,t}^{c}) \\ &= [I_2 - (\mathbf{m}_{n,t} + \mathbf{e}_{n,t}^{c})]Q_{cs}(Q_{sw,t}(\mathbf{m}_{w,t} + \mathbf{e}_{w,t}^{c} - \mathbf{s}_{w,t}) - \mathbf{c}_s). \end{aligned} \tag{12a}$$

This can be reformulated with additive measurement noise $\mathbf{e}_t^{c}$ using (5a) and (5g):

$$\begin{aligned} \mathbf{0} &= h(\mathbf{x}_t, \mathbf{m}_{n,t}, \mathbf{m}_{w,t}) + \mathbf{e}_t^{c} \\ &= [I_2 \quad -\mathbf{m}_{n,t}]Q_{cs}(Q_{sw,t}(\mathbf{m}_{w,t} - \mathbf{s}_{w,t}) - \mathbf{c}_s) + \mathbf{e}_t^{c}, \end{aligned} \tag{12b}$$

where $\mathbf{e}_t^{c} \sim \mathcal{N}(\mathbf{0}, R_t)$ and with $\mathbf{h}_t = h(\mathbf{x}_t, \mathbf{m}_{n,t}, \mathbf{m}_{w,t}, \mathbf{e}_{n,t}^{c}, \mathbf{e}_{w,t}^{c})$

$$R_t \approx \begin{bmatrix} \frac{\partial \mathbf{h}_t}{\partial \mathbf{m}_{n,t}} & \frac{\partial \mathbf{h}_t}{\partial \mathbf{m}_{w,t}} \end{bmatrix} \begin{bmatrix} R_{nn,t} & 0_{2\times 3} \\ 0_{3\times 2} & R_{ww,t} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathbf{h}_t}{\partial \mathbf{m}_{n,t}}^{\mathrm{T}} \\ \frac{\partial \mathbf{h}_t}{\partial \mathbf{m}_{w,t}}^{\mathrm{T}} \end{bmatrix}. \tag{12c}$$

$\mathbf{q}_{cs}$ and $\mathbf{c}_s$ are as in (8). The correspondences are all processed sequentially.

4.3. Model 2 (gravity)

In the gravity model, accelerometers are used for stabilising the camera attitude. The state vector comprises additional parameters for the accelerometer biases $\mathbf{b}_s^a$:

$$\mathbf{x}^{\mathrm{T}} = [\mathbf{s}_w^{\mathrm{T}}\ \dot{\mathbf{s}}_w^{\mathrm{T}}\ \mathbf{q}_{sw}^{\mathrm{T}}\ \omega_s^{\mathrm{T}}\ \mathbf{b}_s^{a\mathrm{T}}\ \mathbf{b}_s^{\omega\mathrm{T}}]. \tag{13}$$

The motion model assumes constant velocity and constant angular velocity including the additional equation:

$$\mathbf{b}_{s,t}^a = \mathbf{b}_{s,t-T}^a + \mathbf{v}_{s,t}^{\mathbf{b}^a}, \tag{14}$$

where $\mathbf{v}_{s,t}^{\mathbf{b}^a}$ denotes the time independent process noise of the accelerometer biases. The accelerometer measurement model involves only the estimated orientation $\mathbf{q}_{sw,t}$:

$$\mathbf{y}_{s,t}^a = -(Q_{sw,t}\mathbf{g}_w) + \mathbf{b}_{s,t}^a + \mathbf{e}_{s,t}^a, \tag{15}$$

where $\mathbf{g}_w$ denotes the gravity direction in the world coordinate system and $\mathbf{e}_{s,t}^a$ denotes the time independent accelerometer measurement noise. In this formulation $\mathbf{e}_{s,t}^a$ includes also free accelerations. The gyroscope (11) and the correspondence measurement model (12) are the same.

4.4. Model 3 (acc)

The acc model uses all information given in the accelerometer measurements, i.e. information about the camera attitude and free acceleration as second derivative of the position. The state vector comprises additional parameters for the free accelerations $\ddot{\mathbf{s}}_w$:

$$\mathbf{x}^{\mathrm{T}} = [\mathbf{s}_w^{\mathrm{T}}\ \dot{\mathbf{s}}_w^{\mathrm{T}}\ \ddot{\mathbf{s}}_w^{\mathrm{T}}\ \mathbf{q}_{sw}^{\mathrm{T}}\ \omega_s^{\mathrm{T}}\ \mathbf{b}_s^{a\mathrm{T}}\ \mathbf{b}_s^{\omega\mathrm{T}}]. \tag{16}$$

The motion model assumes constant acceleration and constant angular velocity changing the upper three equations of the time update to

$$\begin{bmatrix} \mathbf{s}_{w,t} \\ \dot{\mathbf{s}}_{w,t} \\ \ddot{\mathbf{s}}_{w,t} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t-T} + T\dot{\mathbf{s}}_{w,t-T} + \frac{T^2}{2}(\ddot{\mathbf{s}}_{w,t-T} + \mathbf{v}_{w,t}^{\ddot{\mathbf{s}}}) \\ \dot{\mathbf{s}}_{w,t-T} + T(\ddot{\mathbf{s}}_{w,t-T} + \mathbf{v}_{w,t}^{\ddot{\mathbf{s}}}) \\ \ddot{\mathbf{s}}_{w,t-T} + \mathbf{v}_{w,t}^{\ddot{\mathbf{s}}} \end{bmatrix}. \tag{17}$$

The accelerometer measurement model also involves free accelerations:

$$\mathbf{y}_{s,t}^a = Q_{sw,t}(\ddot{\mathbf{s}}_{w,t} - \mathbf{g}_w) + \mathbf{b}_{s,t}^a + \mathbf{e}_{s,t}^a. \tag{18}$$

This changes during the EKF measurement update both the rotational and the translational states. Clearly, this measurement function requires a certain prior estimate of the orientation, as small errors in the orientation result in unbounded errors in the position. Note that this model includes the gravity model. The gyroscope (11) and the correspondence measurement model (12) are the same.

4.5. Model 4 (acc input)

Instead of considering the inertial readings to be measurements, as in the acc model, they can be treated as control inputs $\mathbf{u}^{\mathrm{T}} = [\mathbf{y}_s^{\omega}, \mathbf{y}_s^a]$ to the time update (cf. (4)). Since the angular velocities ω_s and free accelerations $\ddot{\mathbf{s}}_w$ are then given by the control inputs, they can be removed from the state vector $\mathbf{x}$, leading to a reduced dimensionality of the space that has to be estimated. The reduced state vector comprises

$$\mathbf{x}^{\mathrm{T}} = [\mathbf{s}_w^{\mathrm{T}}\ \dot{\mathbf{s}}_w^{\mathrm{T}}\ \mathbf{q}_{sw}^{\mathrm{T}}\ \mathbf{b}_s^{a\mathrm{T}}\ \mathbf{b}_s^{\omega\mathrm{T}}]. \tag{19}$$

The motion model is obtained by solving the gyroscope (11) and the accelerometer measurement equation (18) for $\omega_{s,t}$ and $\ddot{\mathbf{s}}_{w,t}$, respectively, and replacing these variables in the respective time update equations in (9) and (17). The resulting expression is

$$\begin{bmatrix} \mathbf{s}_{w,t} \\ \dot{\mathbf{s}}_{w,t} \\ \mathbf{q}_{sw,t} \\ \mathbf{b}_{s,t}^{\omega} \\ \mathbf{b}_{s,t}^{a} \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{w,t-T} + T\dot{\mathbf{s}}_{w,t-T} + \frac{T^2}{2}Q_{ws,t-T}(\mathbf{y}_{s,t}^a - \mathbf{b}_{s,t-T}^a - \mathbf{v}_{s,t}^a) + \frac{T^2}{2}\mathbf{g}_w \\ \dot{\mathbf{s}}_{w,t-T} + TQ_{ws,t-T}(\mathbf{y}_s^a - \mathbf{b}_{s,t-T}^a - \mathbf{v}_{s,t}^a) + T\mathbf{g}_w \\ \exp\left(-\frac{T}{2}(\mathbf{y}_{s,t}^{\omega} - \mathbf{b}_{s,t}^{\omega} - \mathbf{v}_{s,t}^{\omega})\right) \odot \mathbf{q}_{sw,t-T} \\ \mathbf{b}_{s,t-T}^{\omega} + \mathbf{v}_{s,t}^{b^{\omega}} \\ \mathbf{b}_{s,t-T}^{a} + \mathbf{v}_{s,t}^{b^{a}} \end{bmatrix}. \tag{20}$$

Note that the measurement noises $\mathbf{e}_{s,t}^a$ and $\mathbf{e}_{s,t}^{\omega}$ are here included in the process noises $\mathbf{v}_{s,t}^a$ and $\mathbf{v}_{s,t}^{\omega}$, respectively. The correspondence measurement model (12) is the same.

By treating the angular velocities and accelerations as input signals, six states and two measurement update steps can be saved. The tuning becomes easier, as fewer noises have to be adjusted. Moreover, these noises can be determined experimentally from the IMU hardware by looking at the standard deviations of several seconds of sensor data captured while holding the device stationary. There are two further ramifications to mention. As the state space (19) comprises no angular velocities, the orientation can only be predicted, when inertial readings are available. Furthermore, the state space comprises no body accelerations, implying that no correlations are kept between the orientation and the body acceleration.

4.6. Outlier rejection

The Kalman filter is not robust, a single erroneous measurement can cause total filter divergence. Inertial sensors generally do not produce outliers, but the image analysis does. As mentioned in Section 3, Newton–Raphson-style iterations are used over multiple image resolutions to manage large feature displacements. This gives the whole system essential robustness, but also allows outliers, which pass the simple SSD based validity test after the registration. Since the sensor fusion algorithm produces a very good prediction of where the features should occur in the images, a simple χ^2 test is used for detecting outliers. Let $\mathbf{z}_i$ be the innovation computed for the ith 2D/3D correspondence and let S_i be its innovation covariance with $\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, S_i)$. The outlier test is

$$\mathbf{z}_i^{\mathrm{T}} S_i^{-1} \mathbf{z}_i > \chi^2_{\alpha,2}, \tag{21}$$

where $s_i = \mathbf{z}_i^{\mathrm{T}} S_i^{-1} \mathbf{z}_i$ is referred to as normalised squared residual and α is the significance level. A common value for α is 0.05 yielding the threshold 5.991. During the experiments, this threshold was found to be too tight. Real outliers produced considerably larger residuals. A much looser threshold 15, corresponding to the significant level $\alpha = 10^{-4}$, is therefore applied.

4.7. Divergence monitoring

In order to be applicable a tracking system needs a reliable failure detection. A failure of the vision sensor is easily recognised by looking at the number of registered features. However, due to the inertial sensors the proposed system is able to survive some frames without any features visible. A reinitialisation is therefore only desired if the dead reckoning has introduced a drift inevitably leading to a filter divergence. Thus, the divergence check is based on a combination of several simple tests:

- If the filter diverges, it shows up in the normalised residuals of measurements providing absolute reference such as the vision measurements. The quantity computed in (21) for recognising outliers can therefore be used for the divergence detection [32], too. The normalised residuals s_i are low-pass filtered using $g = \lambda g + (1 - \lambda)s_i$ with $0 \ll \lambda < 1$ and g is compared to a threshold. Note that, in combination with the outlier rejection, g is bounded by the outlier threshold.
- Obviously, the above test does not detect a divergence, when no vision measurements are available. However, a lack of vision measurements results in an increasing state uncertainty. The Frobenius norm of the state covariance is therefore computed and compared to a threshold. Values between 3 and 5 were chosen during the experiments.
- Finally, the norm of the orientation quaternion in the state vector is monitored. The quaternion is normalised before each time update to keep it at unit length, as each measurement update causes a slight change in the norm. A significant change from unit length indicates an estimation error. The threshold 0.95 was chosen during the experiments. By normalising the state quaternion, the filter ceases to propagate the conditional mean. In order to adapt the state covariance appropriately, a method proposed in [33] is used. Let $\mathbf{q}^*_{sw,t}$ and $P^*_{qq,t}$ be the quaternion estimate with minimum squared error at time t. The normalisation step performed after each measurement update is then given by

$$\mathbf{q}_{sw,t} = \frac{\mathbf{q}^*_{sw,t}}{\|\mathbf{q}^*_{sw,t}\|}, \tag{22a}$$

$$P_{qq,t} = P^*_{qq,t} + (\mathbf{q}_{sw,t} - \mathbf{q}^*_{sw,t})(\mathbf{q}_{sw,t} - \mathbf{q}^*_{sw,t})^{\mathrm{T}}, \tag{22b}$$

  where $\mathbf{q}_{sw,t}$ and $P_{qq,t}$ constitute the new constrained estimate used in the subsequent time update.

If at least one of the tests fails, the system is reinitialised.

5. Experimental setup and results

The camera-IMU system used for the experiments is shown in Fig. 5. It combines a monochrome PGR camera with an XSens MT9-C IMU in one housing and is subsequently referred to as CamIMU.

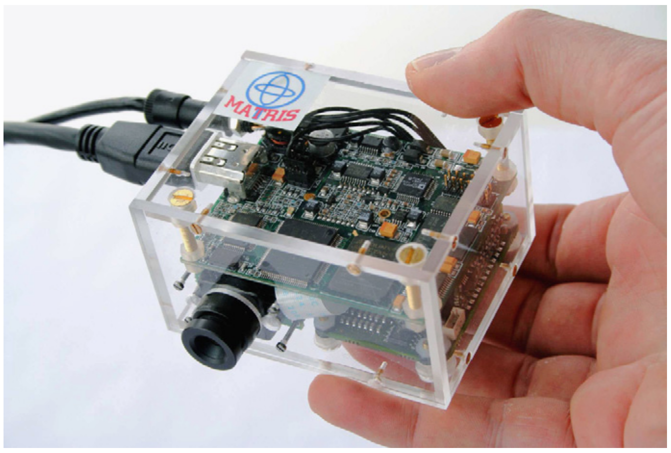

Fig. 5. CamIMU: integrated camera and inertial sensor package. Both devices are synchronised in hardware.

In order to relate the measurements from both sensors, the hand-eye rotation $\mathbf{q}_{cs}$ and translation $\mathbf{c}_s$ have to be calibrated. $\mathbf{q}_{cs}$ has been calculated using Horn's method [34] with a set of gravity measurements from both sensors. These measurements are easily obtained as a byproduct of the camera calibration [22] by placing the calibration pattern horizontally or vertically and by recording several seconds of accelerometer data for each calibration image while keeping the device stationary. The hand-eye translation $\mathbf{c}_s$ is derived from the specifications of the manufacturer concerning the locations of the different sensors (accelerometers, CCD Chip) with respect to the outer casing.

Another rigid rotation $\mathbf{q}_{gw}$ is given between the global reference frame and the world coordinate system (cf. Fig. 4). This quantity is calibrated once online by averaging over several samples $\mathbf{q}_{gw,t} = \mathbf{q}_{gs,t} \odot \mathbf{q}_{sw,t}$, where $\mathbf{q}_{gs,t}$ is the orientation estimate obtained from the IMU and $\mathbf{q}_{sw,t}$ is the estimated sensor orientation at time t. The average $\mathbf{q}_{gw}$ is then used during the system (re)initialisation to predict the initial quaternion state $\mathbf{q}_{sw,t} = \mathbf{q}_{sg,t} \odot \mathbf{q}_{gw}$ (cf. Section 2). The accuracy of this quantity is not critical, as it is only needed for (re)initialisation.

Three different test cases were considered in the experiments, a small-scale test scenario with movements controlled by an industrial robot (*Desktop*), a realistic mid-scale (*Room*) and a large-scale scenario (*Foyer*), both with uncontrolled movements, where the CamIMU was held in the hand. The textured CAD models required by the image processing method were created either manually or semi-automatically from some photos using a commercial modelling tool. Textured CAD models can also be generated automatically from an image sequence using structure from motion algorithms [35,36].

In all test scenarios, the proposed system was able to track the trajectory of the CamIMU online and stably at 25 Hz on 320×240 resolution images. The image analysis revealed no problems with textures from synthetic images or cameras other than the one used for the tracking. For an in-depth evaluation representative data sequences (synchronised images and IMU data) were captured from each scenario. The results are presented in the sequel. The system parameters and noise settings used during the experiments are given in the Appendix.

5.1. *Test case: Desktop*

In order to evaluate the tracking performance of the four sensor fusion models introduced in Section 4, the CamIMU was mounted on a robotic arm observing the small two-sided target object shown previously in Figs. 2 and 3. The robotic arm performed a continuous movement in the shape of an eight at various speeds. Fig. 6 shows camera images from the outer points of this movement. Two data sets were considered for the tests, a *slow sequence* and a *fast sequence*. The estimated accelerations and angular velocities are shown in Fig. 7.

As the fast sequence contains significant body accelerations, which violate the assumptions of the gravity model severely, a simple detection criterion has been applied. Basically, the accelerometer measurement update is omitted at time t, if

$$\left| \|\mathbf{y}^a_{s,t} - \mathbf{b}^a_{s,t}\| - \|\mathbf{g}_w\| \right| > D^a, \tag{23}$$

where $\mathbf{y}^a_{s,t}$ are the measured accelerations, $\mathbf{b}^a_{s,t}$ is the latest bias estimate, $\mathbf{g}_w$ is the gravity vector in the world frame and D^a is a user-defined threshold. The values are given in the Appendix. Additional body acceleration detection criteria are given for instance in [37,38].

The prediction errors of the features in the image plane, that is the Euclidean distances between the predicted feature positions and the registered positions, are used as indicators for the tracking

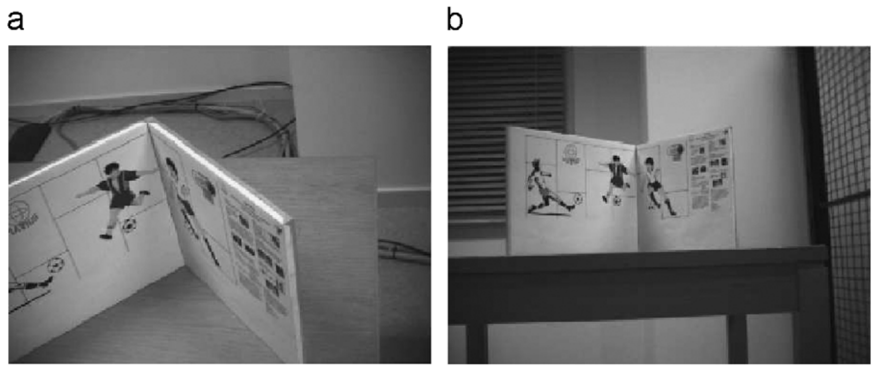

Fig. 6. Desktop: two frames of the image sequences, where the CamIMU was nearest to (a) and farthest from (b) the target object.

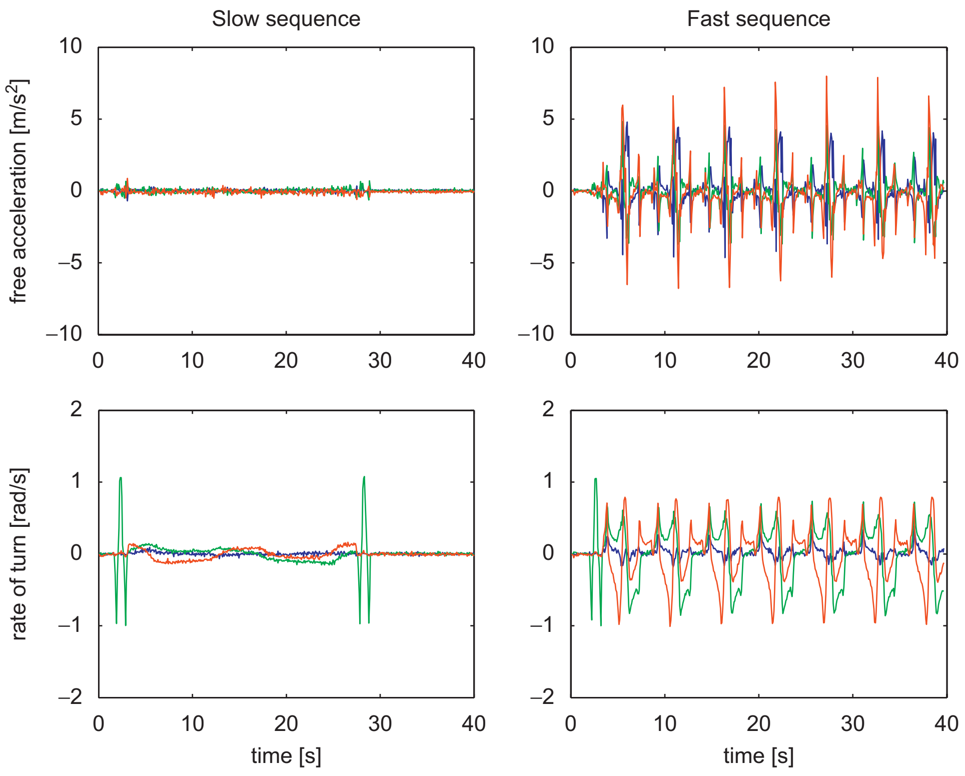

Fig. 7. Desktop: free linear accelerations and angular velocities of the slow and the fast sequence as estimated by the proposed system running in the acc mode.

Table 1
Desktop: root mean square prediction errors in pixels for the different sensor fusion models given as mean [std].

	Gyro	Gravity	acc	acc input
Fast sequence	3.82 [4.04]	3.83 [4.05]	0.91 [0.59]	0.77 [0.45]
Slow sequence	0.51 [0.47]	0.50 [0.47]	0.59 [0.37]	0.69 [0.40]

performance. Table 1 shows clearly how the acc and the acc input model outperform the other models on the fast sequence producing a significantly smaller root mean square prediction error. This also becomes apparent in Fig. 8, where the prediction errors produced by these models remain constant, while the gyro and the gravity model produced errors that are highly correlated with the acceleration peaks in Fig. 7. On the slow sequence, all models perform comparably well, as expected.

Table 1 also shows that the gravity model, which uses accelerometers to stabilise two angles of the orientation estimate, gives no improvement to the gyro model on any of the two sequences. This is explained by the fact that a relatively high feature density has been used for the experiments, providing sufficient information about the orientation. Fig. 9 presents results on the slow sequence, where only two features were used for the vision-based tracking, thus revealing the benefits of using the accelerometers in the gravity as well as in the acc model.

Finally, Table 1 shows that the acc and the acc input model yield comparable results, indicating that the missing correlations between the orientation and the body acceleration in the acc input model do not impact the tracking performance. The acc input model yielded equally stable and precise estimation results as the acc model, but with significantly reduced computational costs. Fig. 10 shows the processing time needed by both models for the EKF time and measurement updates. Using the acc input model reduced the computational costs by a factor of 2.7.

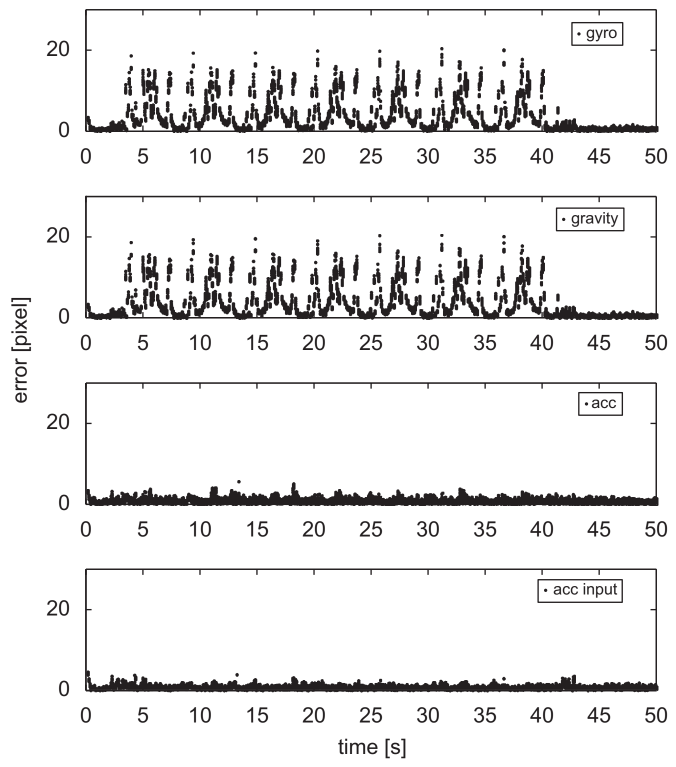

Fig. 8. Desktop, fast sequence: the prediction errors of the registered features show clearly the superiority of the acc and the acc input model. The outliers are already removed.

The experimental results presented so far allow the following conclusions. The gravity model is superior to the gyro model by reducing the number of features needed for correcting the drift in the gyroscopes, but both models show—as expected—a poor tracking quality in the presence of changing linear velocities. The acc and the acc input model also reduce the number of features needed and provide significantly higher prediction quality. Under slow movements, where the constant acceleration model is actually overparameterised, they still perform just as well without introducing instabilities. Moreover, treating the inertial data as control inputs significantly reduces the computational demand, while keeping the tracking quality.

In order to give an idea of the absolute accuracy of the overall system under quick motions, the reference trajectory of the robot has been transformed into the spatial and time reference of the

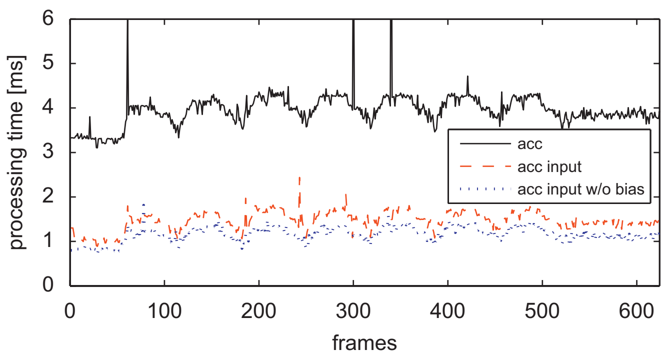

Fig. 10. Desktop, fast sequence: per frame processing time needed for the EKF time and measurement updates by the acc, the acc input and the acc input model without accelerometer bias estimation. This has been measured on a 2.2 GHz laptop. Note that the processing time varies due to a varying number of vision measurements per frame. The three peaks are explained by frame misses, where the IMU buffer contained twice as many IMU readings (cf. Section 2).

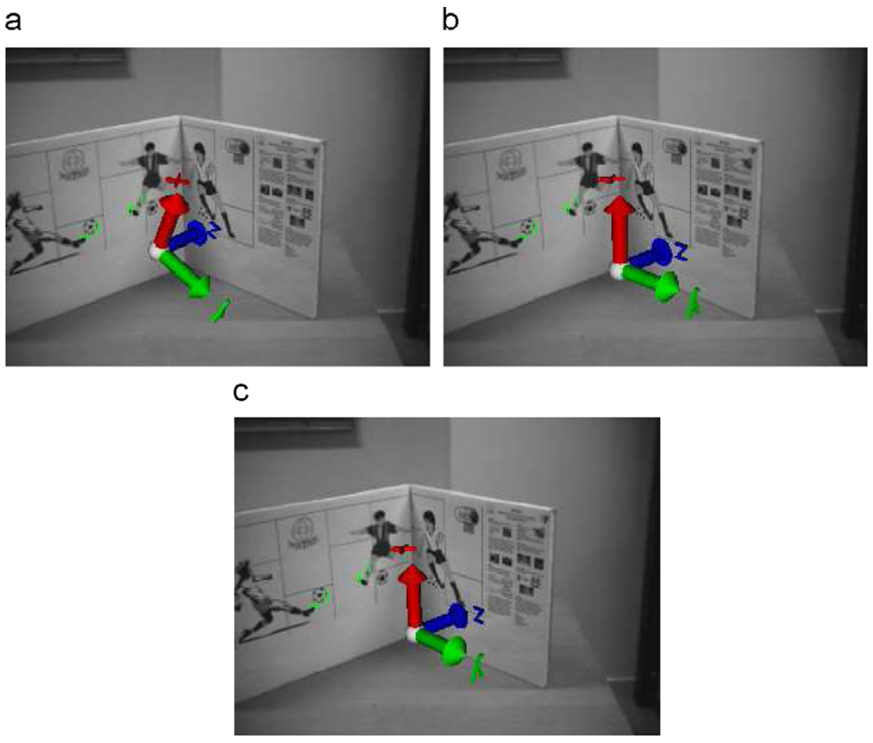

Fig. 9. Desktop, slow sequence: after 22.32 s of tracking with only two features, marked by green crosses, the gyro model (a) has drifted significantly, while the gravity and acc model (b,c) show comparably stable results. This can be seen from the pose of the augmented coordinate system.

fast sequence and compared to the camera trajectory produced by the acc and the acc input model. The average Euclidean position error obtained from this comparison was 1 cm and the average error in the Euler angles 0.76°. Note that the transformations used to synchronise the trajectories contribute to the measured estimation errors, so that the actual accuracy can be assumed higher.

Finally, the online bias estimation has been investigated on the fast sequence using the acc model. The result is shown in Fig. 11. The gyroscope bias estimates—initialised to zero—converge quickly and stay constant over the short duration of the experiment, while the accelerometer bias estimates change with the orientation. This indicates that the gyroscope biases are estimated reliably, while other errors such as calibration or model errors tend to show up incorrectly in the accelerometer bias parameters. As a matter of fact, the online accelerometer bias estimation caused repeated instabilities in the tracking without significantly improving the overall tracking performance as long as corrective vision measurements were available. Table 2

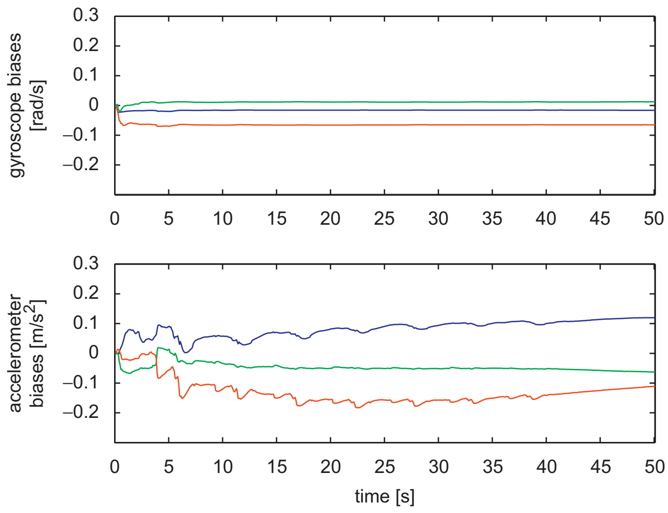

Fig. 11. Desktop, fast sequence: gyroscope and accelerometer biases as estimated by the acc model.

Table 2
Desktop: root mean square prediction errors in pixels for the acc model with and without accelerometer bias estimation given as mean [std].

	w/ bias estimation	w/o bias estimation
Fast sequence	0.908 [0.586]	0.913 [0.589]
Slow sequence	0.586 [0.373]	0.725 [0.412]

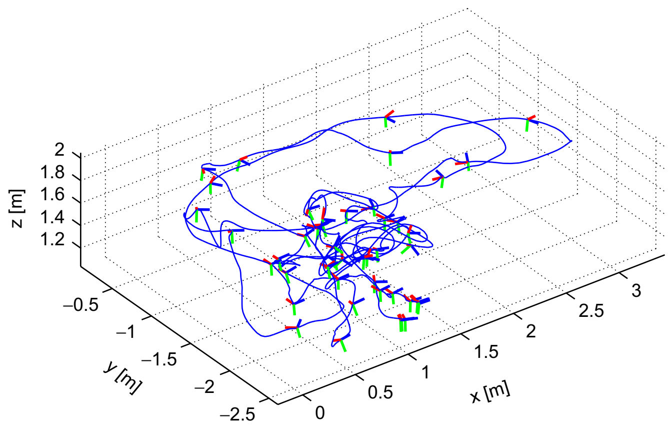

Fig. 13. Room: camera trajectory in world coordinates as estimated by the proposed system. The coordinate systems denote the camera orientation for every 50th frame.

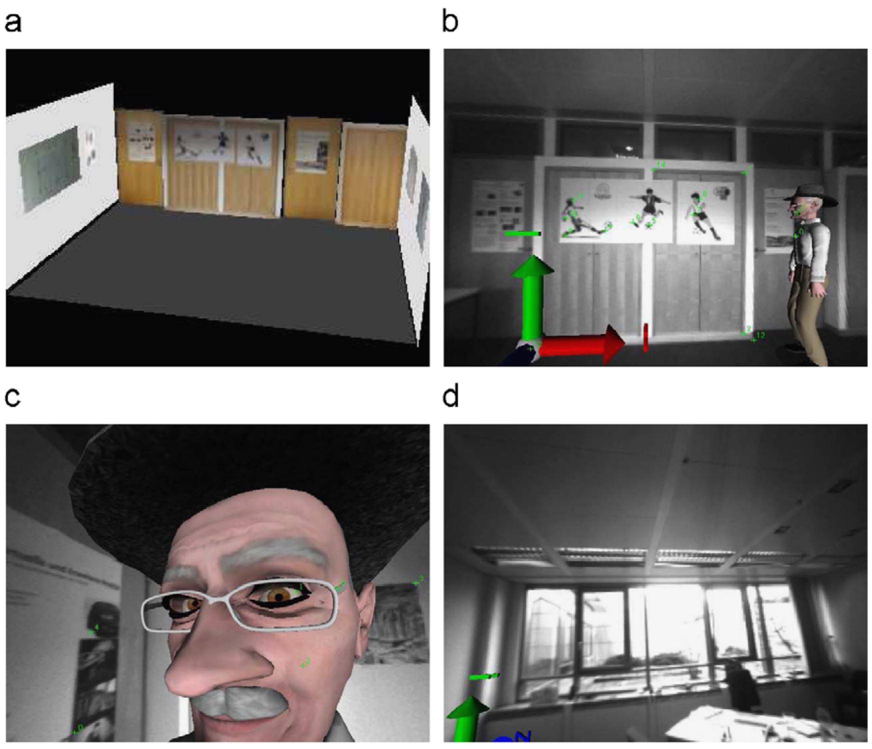

Fig. 12. Room: the CAD model used for the feature registration (a) and some overlaid frames demonstrating the range of possible movements (b–d). Note that the wall observed in (d) is not part of the model. However, the augmentation in the lower left corner stays in its place.

presents the root mean square prediction errors for the features with and without online accelerometer bias estimation and shows that the performance gain is minor. Moreover, the proposed system was found to be very sensitive to the accelerometer bias process noise. These observations and the fact that the gyroscope biases are the dominant sources of error, not only for the rotational, but—if the acc or the acc input model is used—also for the translational states, are the reasons, why only the

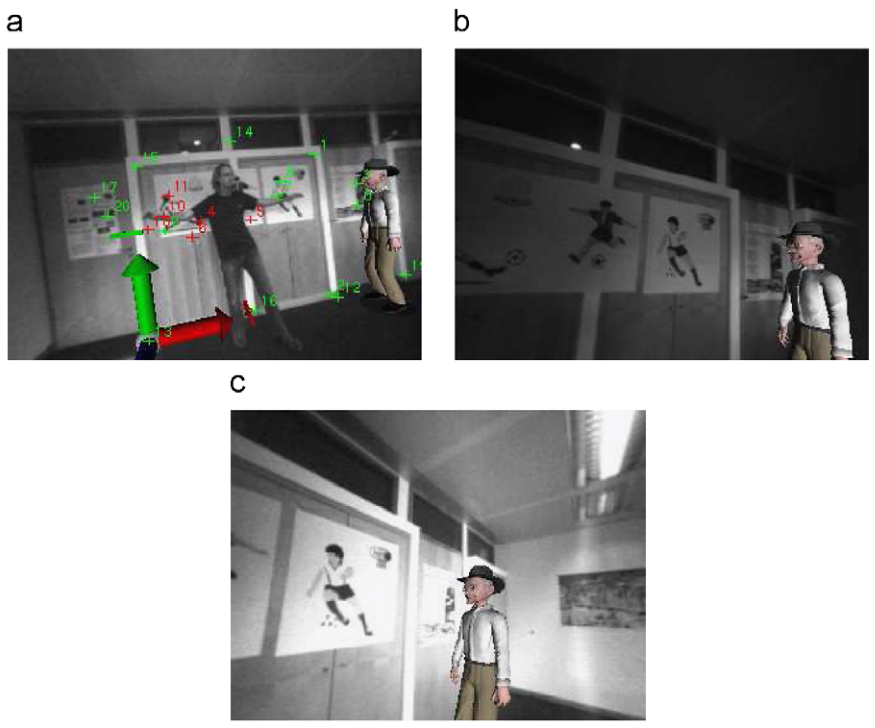

Fig. 14. Room: some overlaid frames demonstrating the stable operation of the system under occlusion (a) and changing illumination (b,c). In (a) the green crosses indicate registered features and the red ones mark features where the registration failed due to the occlusion.

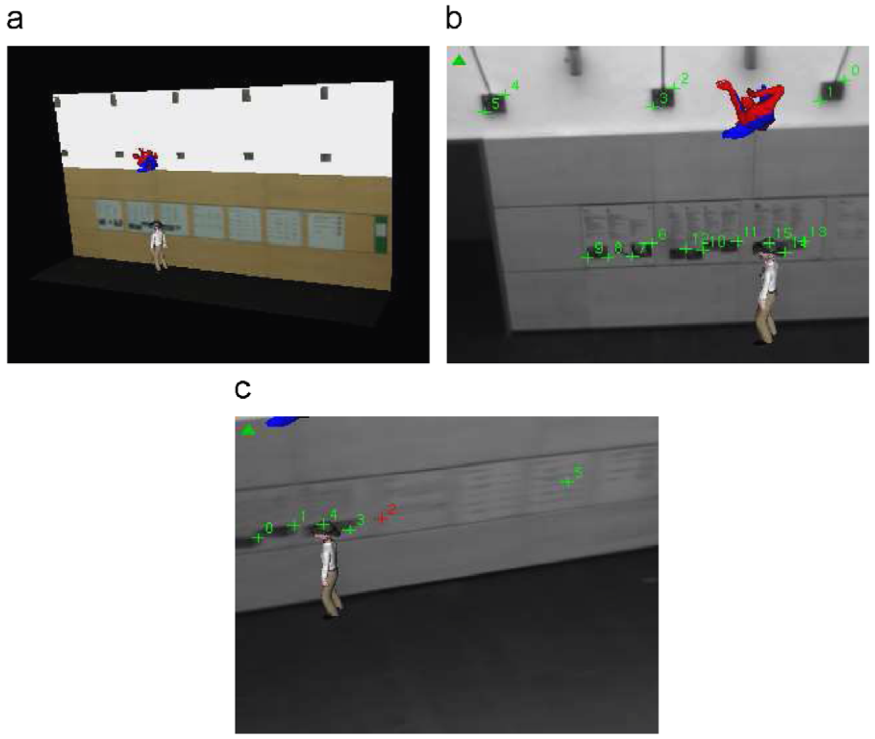

Fig. 15. Foyer: (a) CAD model used for the feature registration and virtual objects, (b) rectified live image with feature positions and augmentations, and (c) another live image with significant motion blur where features were nevertheless registered.

gyroscope biases were estimated online during the subsequent experiments. This implies a reduction of the state vector by three states and hence further computational savings, which are visualised in Fig. 10 as well.

5.2. *Test case: Room*

In this scenario a person walks in a room of base area 5×4 m and carries the CamIMU. Three walls of the room are modelled as shown in Fig. 12(a). The captured data sequences contain a significant range of rotations and translations demonstrated in Fig. 13 and some other challenging parts, which are all handled robustly by the proposed system running in the acc or the acc input mode. Two 360° rotations were performed whereas one wall was not included in the CAD model. The tracking survived these periods of pure dead reckoning as indicated in Fig. 12(d). When the user moved the camera very close to the target scene seeing only few features, the augmentation still remained in its place (Fig. 12(c)). Finally, some fast movements with free accelerations up to $12.9\,\mathrm{m/s^2}$ and rotations up to 9.6 rad/s were tracked successfully. Fig. 14 demonstrates the stable operation of the system under occlusion and severe lighting changes.

5.3. *Test case: Foyer*

In this scenario a person carrying the CamIMU is standing on the first floor of a building behind a handrail looking down into a large open foyer. A part of the opposite wall (4.5×8.5 m) at a distance of approximately 7 m is used for the natural feature registration. Fig. 15 shows the textured CAD model and some augmented live camera images. Note that the features on the posters look quite similar due to the big viewing distance. A very precise prediction is therefore crucial to maintain the tracking.

This sequence could not be processed using the gravity or the gyro model, although the camera movements were indeed mainly rotational. However, the rotations were off-centre and hence produced also high linear accelerations. Fig. 16 shows the estimated free linear accelerations and angular velocities based on the acc model, which tracked robustly throughout the whole sequence.

Fig. 17 presents the prediction errors for the features with the detected outliers marked by stars. The root mean square prediction error over the whole image sequence was 1.42 pixels with 1.47 pixels standard deviation. Some increased displacements between time 60 and 70 are explained by vision data repeatedly missing for up to half a second. Without vision data the pose was based solely on dead reckoning.

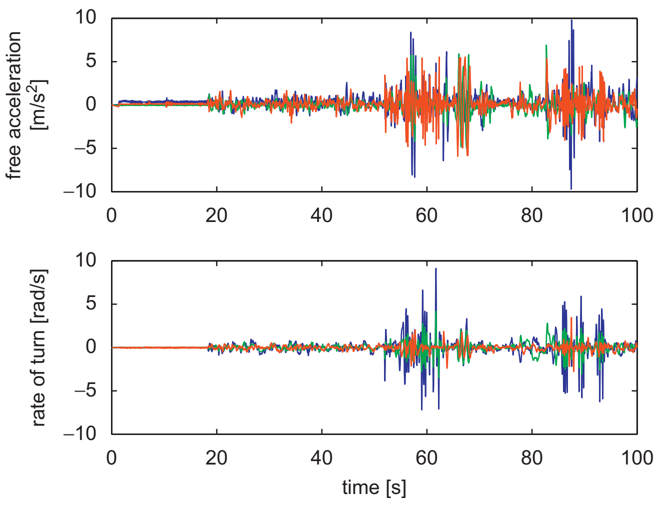

Fig. 16. Foyer: free linear accelerations and angular velocities as estimated by the proposed system running in the acc mode.

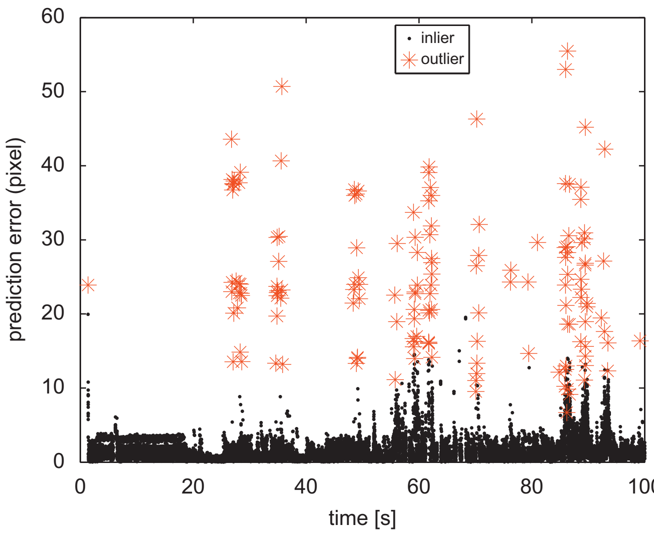

Fig. 17. Foyer: prediction errors for the features produced by the acc model.

6. Conclusion

This article presents a markerless visual–inertial tracking system that works robustly in small- and large-scale environments, under varying lighting conditions, fast camera movements and even short periods without vision data.

A markerless image processing method based on the analysis-by-synthesis technique has been developed, which in the most efficient way exploits a running pose prediction by first rendering a simplified textured 3D model of the environment for predicting the feature appearances and second registering the features in the live camera images by estimating their 2D displacements and local illumination.

Moreover, different sensor fusion models have been evaluated under controlled movements, showing clearly the benefits of fully exploiting all information given by the accelerometer measurements. The benefits are better tracking quality and reduced demand for vision measurements. These results were also verified under uncontrolled movements in realistic mid- and large-scale environments.

By combining the developed image processing algorithm with the fusion strategy of the acceleration input model, a very efficient and robust final system is obtained.

It is important to note that, due to the image processing approach used, the proposed tracking system is suitable for environments containing richly textured planar regions and walls. However, the analysis-by-synthesis technique has also been applied successfully to poorly textured environments by exploiting prominent contours, see for instance [39–41]. The developed system allows for a smooth exchange of the image processing algorithm, thus making it easy to extend its applicability to such scenarios.

Model-based approaches provide stable tracking and are suitable for a wide range of applications. However, the model generation and maintenance can be time consuming and the assumption of a static environment is often violated. Future work will therefore consist in expanding the system with real-time SLAM capabilities.

Acknowledgements

This work has been performed within the EU research program MATRIS (IST-002013). The authors would like to thank the project partners within the consortium and the EU for the financial support. The data sequences controlled by the industrial robot were provided by Linköping University. For more information, visit the website http://www.ist-matris.org.

Appendix A

This appendix provides the settings for the system parameters and the noises—given as standard deviations and assuming equal noise in all dimensions—used during the different experiments. The noise affecting the feature registration has been determined experimentally by looking at the change in the 2D pixel locations while the camera was stationary. The 3D model coordinates, which are obtained from the target scene model, were assumed certain.

The settings for the test case *Desktop* are given below. Note that the process and measurement noise settings related to the accelerations differ significantly between the models and data sequences. In order to assure meaningful evaluation results, appropriate settings were derived for each model from the model assumptions and the reference trajectory of the robot.

	Gyro	Gravity	acc	acc input
Fast sequence				
$\mathbf{v}^{\ddot{s}}_{w,t}$	4	4	0.1	–
$\mathbf{e}^{a}_{s,t}$	–	0.84	0.14	–
D^{a}	–	0.4	–	–
Slow sequence				
$\mathbf{v}^{\ddot{s}}_{w,t}$	0.19	0.19	0.1	–
$\mathbf{e}^{a}_{s,t}$	–	0.28	0.14	–
D^{a}	–	–	–	–
Both				
$\mathbf{v}^{a}_{s,t}$	–	–	–	0.14
$\mathbf{v}^{\omega}_{s,t}$	0.1	0.1	0.1	0.01
$\mathbf{v}^{b^{\omega}}_{s,t}$	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
$\mathbf{v}^{b^{a}}_{s,t}$	1×10^{-5}	1×10^{-5}	1×10^{-5}	1×10^{-5}
$\mathbf{e}^{\omega}_{s,t}$	0.01	0.01	0.01	–
$\mathbf{e}^{n}_{t}$	7×10^{-3}	7×10^{-3}	7×10^{-3}	7×10^{-3}

The subsequent table provides the settings for the test cases *Room* and *Foyer*, where the acc model was used.

	Room	Foyer
$\mathbf{v}^{\ddot{s}}_{w,t}$	0.1	1
$\mathbf{v}^{\omega}_{s,t}$	0.1	0.1
$\mathbf{v}^{b^{\omega}}_{s,t}$	5.5×10^{-4}	5.5×10^{-4}
$\mathbf{e}^{a}_{s,t}$	0.04	0.18
$\mathbf{e}^{\omega}_{s,t}$	0.023	0.06
$\mathbf{e}^{n}_{t}$	7×10^{-3}	7×10^{-3}

References

[1] Titterton D, Weston J. Strapdown inertial navigation technology. 2nd ed. NJ: American Institute of Aeronautics and Astronautics; 2004.
[2] Yokokohji Y, Sugawara Y, Yoshikawa T. Accurate image overlay on video see-through HMDs using vision and accelerometers. In: IEEE virtual reality conference, New Brunswick, NJ, USA, March 2000. p. 247–54.
[3] Qian G, Chellappa R, Zheng Q. Robust structure from motion estimation using inertial data. Optical Society of America 2001;18:2982–97.
[4] Klein G, Drummond T. Tightly integrated sensor fusion for robust visual tracking. In: British machine vision conference, vol. 2, Cardiff, September 2002. p. 787–96.
[5] Aron M, Simon G, Berger M-O. Handling uncertain sensor data in vision-based camera tracking. In: International symposium on mixed and augmented reality, Arlington, VA, November 2004. p. 58–67.
[6] Pinies P, Lupton T, Sukkarieh S, Tardos JD. Inertial aiding of inverse depth SLAM using a monocular camera. In: IEEE international conference on robotics and automation, Rome, Italy, April 2007.
[7] Reitmayr G, Drummond T. Going out: robust model-based tracking for outdoor augmented reality. In: International symposium on mixed and augmented reality, Santa Barabara, CA, October 2006. p. 109–18.
[8] Qian G, Chellappa R, Zheng Q. Bayesian structure from motion using inertial information. In: International conference on image processing, Rochester, NY, USA, September 2002.
[9] You S, Neumann U. Fusion of vision and gyro tracking for robust augmented reality registration. In: Virtual reality, Yokohama, Japan, March 2001.
[10] Jiang B, Neumann U, You S. A robust hybrid tracking system for outdoor augmented reality. In: IEEE virtual reality conference, Chicago, Illinois, USA, March 2004.
[11] Foxlin E, Naimark L. VIS-tracker: a wearable vision-inertial self-tracker. In: IEEE virtual reality, Los Angeles, CA, March 2003.
[12] Rehbinder H, Gosh B. Pose estimation using line-based dynamic vision and inertial sensors. IEEE Transactions on Automatic Control 2003;48: 186–99.
[13] Schön T, Gustafsson F. Integrated navigation of cameras for augmented reality. In: International federation of automatic control (IFAC) World congress, Prague, Czech Republic. International Federation of Automatic Control; July 2005.
[14] Hol J, Schön T, Gustafsson F, Slycke P. Sensor fusion for augmented reality. In: International conference on information fusion, Florence, Italy, August 2006.
[15] Schön T, Karlsson R, Törnqvist D, Gustafsson F. A framework for simultaneous localization and mapping utilizing model structure. In: International conference on information fusion, Quebec, Canada, July 2007.
[16] Mikolajczyk K, Schmid C. A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005;27:1615–30.
[17] Lepetit V, Fua P. Keypoint recognition using randomized trees. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(9): 1465–79.
[18] Boffy A, Genc Y. Real-time feature matching using adaptive and spatially distributed classification. In: British machine vision conference, Edinburgh, September 2006.
[19] Grabner M, Grabner H, Horst B. Tracking via discriminative online learning of local features. In: Conference on computer vision and pattern recognition, Minneapolis, MN, June 2007.
[20] Wuest H, Vial F, Stricker D. Adaptive line tracking with multiple hypotheses for augmented reality. In: International symposium on mixed and augmented reality, Vienna, Austria, October 2005. p. 62–9.
[21] Shi J, Tomasi C. Good features to track. In: Conference on computer vision and pattern recognition, Seattle, June 1994. p. 593–600.
[22] Heikkilae J, Silven O. A four-step camera calibration procedure with implicit image correction. In: Conference on computer vision and pattern recognition, San Juan, Puerto Rico, June 1997.
[23] Rosten E, Drummond T. Fusing points and lines for high performance tracking. In: IEEE international conference on computer vision, vol. 2, Beijing, China, October 2005. p. 1508–11.
[24] Bleser G, Wuest H, Stricker D. Online camera pose estimation in partially known and dynamic scenes. In: International symposium on mixed and augmented reality, Santa Barabara, CA, October 2006. p. 56–65.
[25] Bleser G, Stricker D. Advanced tracking through efficient image processing and visual-inertial sensor fusion. In: IEEE virtual reality conference, Reno, Nevada, March 2008.
[26] Kim J, Sukkarieh S. Real-time implementation of airborne inertial-SLAM. Robotics and Autonomous Systems 2007;55:62–71.
[27] Zinßer T, Gräßl C, Niemann H. Efficient feature tracking for long video sequences. In: Deutsche Arbeitsgemeinschaft Mustererkennung, Täbingen, Germany, August 2004. p. 326–33.
[28] Kalman RE. A new approach to linear filtering and prediction problems. Transactions of the ASME—Journal of Basic Engineering 1960;82(Series D): 35–45.
[29] Welch G, Bishop G. An introduction to the Kalman filter. Technical Report, Department of Computer Science, University of North Carolina at Chapel Hill; 2001.
[30] Thrun S, Burgard W, Fox D. Probabilistic robotics. Cambridge: MIT Press; 2005.
[31] Zhang Z, Faugeras O. 3D dynamic scene analysis. Berlin: Springer; 1992.
[32] Gustafsson F. Adaptive filtering and change detection. New York: Wiley; 2000.
[33] Julier S, LaViola J. On Kalman filtering with nonlinear equality constraints. IEEE Transactions on Signal Processing 2007;55:2774–84.
[34] Horn BKP. Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America 1987;4:629–42.
[35] Pollefeys M, Koch R, Gool LJV. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In:

International conference on computer vision, Bombay, India, January 1998. p. 90–5.
[36] Pollefeys M, Gool LV, Vergauwen M, Verbiest F, Cornelis K, Tops J, et al. Visual modeling with a hand-held camera. International Journal of Computer Vision 2004;207–32.
[37] Rehbinder H, Hu X. Drift-free attitude estimation for accelerated rigid bodies. In: IEEE international conference on robotics and automation, Seoul, Korea, May 2001.
[38] Harada T, Mori T, Sato T. Development of a tiny orientation estimation device to operate under motion and magnetic disturbance. The International Journal of Robotics Research 2007;26(6):547–59.
[39] Thomas G. Real-time camera pose estimation for augmenting sports scenes. In: Conference on Visual Media Production, London, UK, November 2006.
[40] Wuest H, Wientapper F, Stricker D. Adaptable model-based tracking using analysis-by-synthesis techniques. In: International conference on computer analysis of images and patterns, Vienna, Austria, August 2007.
[41] Koch O, Teller S. Wide-area egomotion estimation from known 3D structure. In: Conference on computer vision and pattern recognition, Minneapolis, Minnesota, June 2007.