Fusion of unsynchronized optical tracker and inertial sensor in EKF framework for in-car Augmented Reality delay reduction

Jason Rambach*[1], Alain Pagani[1], Sebastian Lampe†[2], Ruben Reiser[1], Manthan Pancholi[1] and Didier Stricker[1,3]

[1]German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany
[2]Volkswagen AG Group Research , Wolfsburg, Germany
[3]TU Kaiserslautern, Kaiserslautern, Germany

Abstract

In this paper we present a novel sensor fusion framework between unsychronized optical tracking systems and inertial measurement units based on an Extended Kalman Filter (EKF). The main benefit from the fusion is taking advantage of the faster speed of measurement availability of the inertial sensor in order to decrease the delay of the system while maintaining the accuracy of the optical tracking system. The tracking framework is applicable in a car cockpit Augmented Reality system displaying augmentations using the tracked 6 Degree of Freedom pose of a head mounted display.

Index Terms: H.5.1 [Multimedia Information Systems]: Artificial, augmented, and virtual realities—; I.4.8 [Scene Analysis]: Sensor Fusion—MotionTracking; I.2.9 [Robotics]: Sensors—

1 Introduction

Fusion between visual and inertial sensors is commonly used for pose tracking applications and odometry because of the complementary nature of the sensors. Visual tracking is in general more accurate than the high noise level inertial sensors, but is provided at a lower frequency and is susceptible to errors when the input image quality is compromised.

A number of different approaches for visual-inertial fusion have been proposed over the years, including the use of statistical filtering (Extended Kalman Filter, Particle Filter) [4, 3, 7], optimization [6], or more recently learning approaches [12]. However, most approaches consider sensors that are synchronized on hardware level, e.g. through a triggering signal.

Precise 6 Degree of Freedom (DoF) pose tracking is a crucial prerequisite for any Augmented Reality (AR) application since it allows for precise placement of 3D augmentations in the real world without jittering or floating effects [8]. See-through head mounted displays (HMD) provide the most immersive AR experience for the user but have high requirements regarding the latency between the user's movement and the update of the pose of the rendered objects [2].

For a constrained area AR application like in the inside of a vehicle, it is reasonable to rely on the so called inside-out tracking for the determination of the pose. To this extent, an optical tracking system that consists of several cameras tracking reflective markers can be used. This type of tracking systems can compute a pose very accurately and robustly, however they induce some delay for the computation of the pose from the marker positions and the transmission of the pose to the HMD.

*Jason_Raphael.Rambach@dfki.de
†sebastian.lampe@volkswagen.de

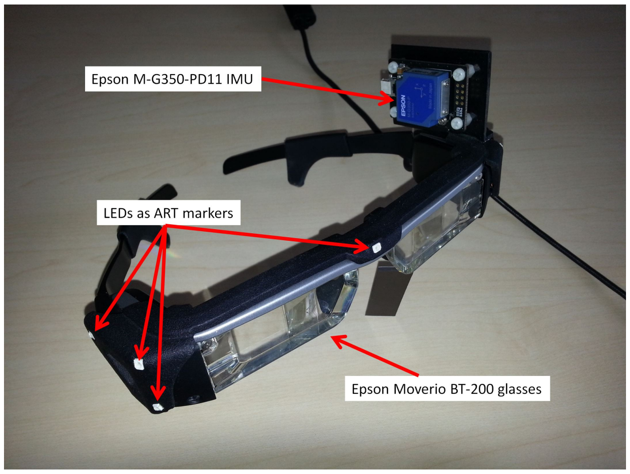

Figure 1: **The equipped AR headset** *The headset consists of a pair of AR glasses (Epson Moverio BT-200), active LEDs used as markers for the optical tracking system, and a rigidly fixed IMU (Epson M-G350-PD11).*

In [5] the requirement for exact synchronization of different sensors for Mixed Reality applications is stressed and a calibration and time delay estimation strategy is presented. In [10], an optical tracker is used inside a vehicle to track the head pose of the driver from reflective markers without using additional sensor input. Finally, in [9] it is attempted to solve the problem of tracking inside a vehicle by deploying inside-out tracking, using cameras on the HMD that localize a fiducial on the car console. An inertial sensor is used for stabilization of the tracking and resilience to visual target occlusion.

In this work we provide for the first time the Extended Kalman Filter formulation to perform tightly coupled fusion of an optical target with an attached inertial sensor. Furthermore, we present an entire EKF-based formulation that deals with non-synchronized optical and inertial measurements. Within this framework, the faster availability of inertial measurements together with a motion model is applied in order to predict the correct pose to be used for displaying 3D augmentations on the HMD of the user.

2 Problem Description

The main objective of this work is to track the 6 DoF pose of an Augmented Reality headset inside a vehicle in order to be able to display 3D information in the user's field of view. An optical tracking system (OT) and an inertial measurement unit (IMU) consisting of an accelerometer and a gyroscope are available for the localiza-

(a)

(b)

Figure 2: **The ART optical tracking system** *(a) One of the two ART cameras and (b) calibration markers*

tion. The data from the two systems are fused in order to combine the accuracy of the optical tracker with the higher frequency and speed of transmission of the inertial sensor.

2.1 Trackable AR headset

The AR headset used in this work consists of various components and sensors rigidly attached (see Figure 1). The main device used for displaying graphical information is an optical see-through HMD of type Epson Moverio BT-200. The aim of an AR system is to make the virtual objects rendered in the field of view of the user and the real world coincide in a seamless way. In order to track the HMD it is equipped with a plastic mount containing a number of infrared LEDs that serve as markers for an external optical tracking system (OT). The optical tracking system used was an ART TRACKPACK [1]. The relative positions between the LEDs are calibrated once for all such that the whole headset can be tracked as one rigid object with multiple markers by the OT system (see Figure 2). An inertial sensor Epson M-G350-PD11 IMU is rigidly attached on the HMD, on the left side of the user.

2.2 Notations

The notation used throughout the paper is introduced here. The position of a point m in 3D space, expressed in coordinate system A is defined as $\boldsymbol{m}_a = (X,Y,Z)^\mathsf{T}$ and its velocity as $\dot{\boldsymbol{m}}_a = (\dot{X},\dot{Y},\dot{Z})^\mathsf{T}$.

$\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ is a multidimensional Gaussian distribution of mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$.

Throughout this work three coordinate systems are used and defined here, namely the world frame W defined by the optical tracking system during the area calibration, the optical frame O fixed to the optical target and the inertial frame S is fixed to the inertial system.

The 6 DoF conversion from one coordinate system A to a coordinate B is described through a rotation matrix and a translation vector. For the transformation from A to B, we use the following notation: the rotation matrix is noted $\boldsymbol{R}_{ba}$ and the translation vector is noted $\boldsymbol{a}_b$, so that the transformation of a point $\boldsymbol{m}$ can be written

$$\boldsymbol{m}_b = \boldsymbol{R}_{ba}\boldsymbol{m}_a + \boldsymbol{a}_b \tag{1}$$

Note that the translation vector $\boldsymbol{a}_b$ corresponds to the coordinates of the center of the coordinate system A expressed in the coordinate system B. Throughout the paper, the equivalent quaternion representation of a rotation matrix $\boldsymbol{R}_{ba}$, $\boldsymbol{q}_{ba} = [q_w,q_x,q_y,q_z]$ is often used.

2.3 Delay definition

In practice, many steps are required between the time an image of markers (LEDs) is acquired by the OT and the moment the corresponding virtual view appears on the display in front of the user's eye, including detection of markers in the images of the infrared tracking cameras, transmission of the positions of the markers from the cameras to the OT controller, computation of a pose, transmission of the pose, computing of a virtual scene for this pose, transmission of the virtual scene to the display and final rendering of the scene on the display. This leads to a non-negligible delay between user's movement and update of the virtual object's position. This delay is inherent to any optical see-through system, and produces a visual effect of virtual objects floating instead of being rigidly positioned in the world.

The hardware setup used in the project has the constraint that the diverse sensors (IMU, OT) are not synchronized on the hardware level (e.g. by triggering), but in software, using the available APIs in parallel after a calibration procedure. This software-based synchronization induces a number of important consequences in terms of parametrization and delay reduction strategies. Internally, the synchronization module has interfaces to the two independent APIs, implemented to run in parallel using multiple threads. The IMU thread runs a loop that constantly seeks for new data from the IMU sensor and stores the received values along with a timestamp t_g^s in a first FIFO-queue $Queue_{\mathrm{imu}}$. Similarly, the OT thread runs a loop that constantly seeks for new data from the OT system and stores the received values along with a timestamp t_g^o in a second FIFO-queue $Queue_{\mathrm{ot}}$. The subscripts g for the timestamps in the queues refer to the fact that these are timestamps from the global system clock of the application PC.

As can be expected, the software API of the sensors do not deliver data instantaneously, but naturally induce some delay that we call signal delivery delay. This delay can be defined as the time required by the system between the moment a physical event is measured (activation event) and the time the measured data is available in the application through the sensor's API (time at which the event can be timestamped in the application by the global clock).

Naturally, this delay is non-zero and differs from sensor to sensor. We call Δ_ϕ^o the signal delivery delay of the OT system, where the subscript ϕ stands for the difference between the time of the physical event and the global clock. Similarly, we name Δ_ϕ^s the signal delivery delay of the IMU sensor. As a consequence, if an OT measurement has a timestamp t_g^o, then the measured pose was actually observed at time $t_\phi^o = t_g^o - \Delta_\phi^o$. Similarly, if the IMU measurement has a timestamp t_g^s, then the corresponding accelerations and angular velocities have been observed at time $t_\phi^s = t_g^s - \Delta_\phi^s$. In other words, when using the global system clock in software synchronization of two sensors, we are actually using two different clocks that differ by a constant difference of $\Delta_{os} = (\Delta_\phi^o\text{-}\ \Delta_\phi^s)$.

Figure 3 shows the different delays we are considering in the project. The two top timelines show the OT signal delay between the physical events (called *OT events*) and the moment at which the measured signal is available in the synchronization module (*OT delivery*). Similarly the third and fourth timelines show the signal delivery delay for the IMU sensor. If we consider a given time t_0 (dashed line marked C in the middle), then the available OT data dates back to time $t_0 - \Delta_\phi^o$ (A-mark), and the latest available IMU data dates back to time $t_0 - \Delta_\phi^s$ (B-mark). The delay Δ_{os} between instants A and B is calibrated once during the IMU-OT calibration (Section 2.4). In order to synchronize OT measurements with IMU measurements, one has to shift the IMU measurements a number of samples corresponding to the delay Δ_{os} (called *IMU shifts* in the figure). We call this synchronization of sensors *time shift*.

Figure 3 also shows the *rendering* delay Δ_r, which is the time required to generate an image and to transmitted and get it physically rendered on the screen of the glasses. This delay is unknown, but we can try to find an estimate by eye inspection and adjusting parameters in our implementation. Note that we will rather estimate the whole *processing* delay $\Delta_p = \Delta_r + \Delta_\phi^s$, which is the sum of two

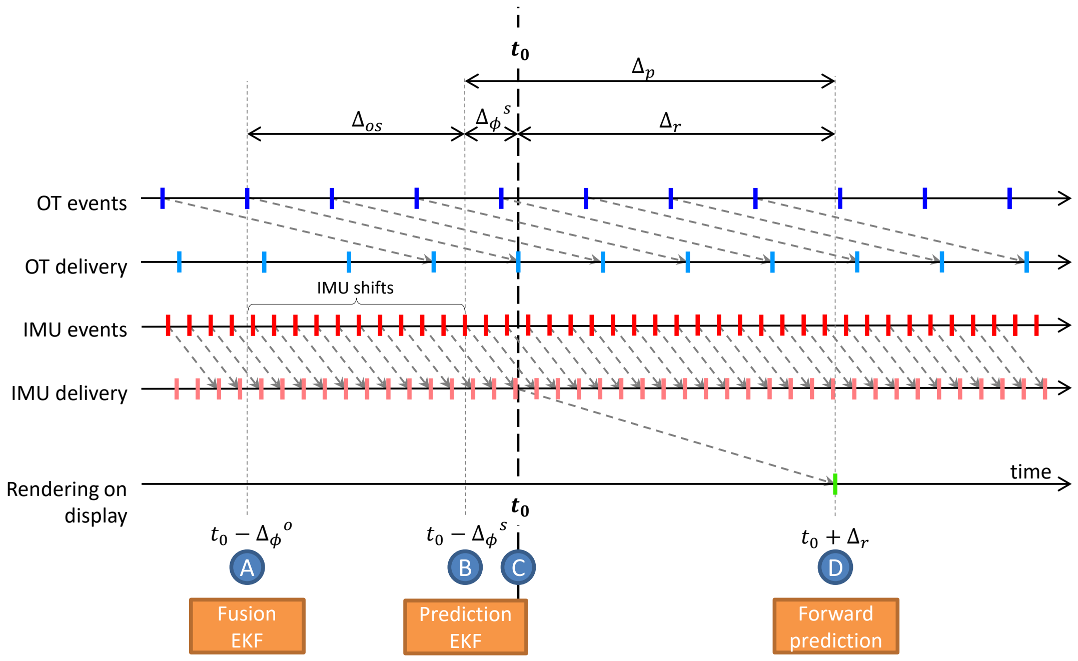

Figure 3: **Time shifts and delays between sensors** *Both OT and IMU sensors have a non-zero delivery time, which needs to be taken into account in delay reduction strategies (see Section 3 for details)*

unknown delays.

2.4 Calibration and time synchronization

For the calibration of the IMU and OT coordinate systems the translation and rotation between them has to be determined. We refer to the calibration of the rigid transorfmation between the IMU coordinate frame S and the OT coordinate frame O as *hand-eye calibration*. The output of the hand-eye calibration is the transformation between S and O, described by a rotation matrix $\boldsymbol{R}_{os}$ and a translation vector $\boldsymbol{s}_o$. For determining this calibration, we use the genetic algorithm approach introduced in [11], modified so that the time shift Δ_{os} between IMU and OT can be estimated as well. To achieve this, a range of possible time shifts are tested and the time shift giving the best result based on the overlap of the simulated to the real IMU data as defined in [11] is output together with the corresponding translation and rotation calibration parameters. In our experiments, this difference turned out to be positive, which means that at any given point in time, the latest available OT data is older than the latest available IMU data.

3 FUSION APPROACH

For robust fusion of IMU readings and OT pose measurements we implemented a statistical filtering framework using an Extended Kalman Filter (EKF) framework [4]. In such approaches for visual-inertial fusion the higher frequency IMU readings are used to perform multiple **prediction steps** of the filter while the lower frequency and lower noise level 2D-3D correspondences from matching of visual features from a camera are used for the **correction step** of the filter. Using an EKF is proven to be a well performing and computationally efficient solution for the problem.

In this work there are two main differences concerning the sensor fusion implementation. Primarily instead of the 2D-3D correspondences from a camera the correction input is given as a 6 DoF pose from the OT system, and secondly the two devices (OT and IMU) are not synchronized on hardware level, e.g. through triggering. Since the optical system provides very accurate poses, the main gain from using the IMU measurements is dealing with the delay of the OT system instead of an increase in accuracy.

3.1 Fusion Strategy

We will now describe the different fusion strategies depending on the available data at different moments in time. Let us again consider Figure 3 and consider a current time t_0 (dashed line marked C in the middle):

- **For time instant *A***, which lies Δ^o_ϕ time units in the past, we have access to all the previous OT poses and all the previous IMU signals until ***A***. We can therefore apply a complete Extended Kalman Filter where the OT poses serve as filter measurements and the IMU signals as control input for the prediction step. This EKF will also learn the possible biases of the IMU. We name it **fusion EKF**.
- **For time instant *B***, which lies Δ^s_ϕ time units in the past, we have access to all the previous OT poses and IMU signals until A, plus a number of IMU signals between ***A*** and ***B***. Starting from the filtered pose computed for ***A***, we can continue to predict the pose using IMU signals only, by using the prediction part of an EKF. We name it **prediction EKF**.
- **After time instant *B***, no further data is available. Moreover the exact duration of the delay Δ^s_ϕ between ***B*** and ***C*** and the delay Δ_r between ***C*** and ***D*** is not known. The only possibility is to assume a certain motion model (*such as constant velocity, constant acceleration*) and predict the pose after a

time $\Delta_p = \Delta_r + \Delta_\phi^s$ under this assumption. In order to estimate the duration Δ_p, we process the data in offline mode, where we have access at each point in time to "future" OT measurements. Thus, we can infer the pose of the glasses at the expected instant $\boldsymbol{D}$ (instant where the rendered image appears on the screen), by using a motion-model-based **forward prediction**.

3.2 Extended Kalman Filter

Extended Kalman Filters are commonly used for fusion of inertial and visual sensors for tracking [4]. The general EKF equations are given here. We denote $\boldsymbol{x}_t$ the filter state, $\boldsymbol{u}_t$ the control input, $\boldsymbol{v}_t$ the process noise with $\boldsymbol{v}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}_t)$, $\boldsymbol{y}_t$ a measurement and $\boldsymbol{e}_t$ the measurement noise with $\boldsymbol{e}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{R}_t)$. $\hat{\boldsymbol{x}}_t$ is the estimate of $\boldsymbol{x}_t$ at time t with $\boldsymbol{x}_t \sim \mathcal{N}(\hat{\boldsymbol{x}}_t, \boldsymbol{P}_t)$. The equations for the **time update (prediction step)** of the state and state covariance of the EKF are

$$\hat{\boldsymbol{x}}_{t|t-T} = f(\hat{\boldsymbol{x}}_{t|t-T}, \boldsymbol{u}_t, \mathbf{0}), \tag{2a}$$

$$\boldsymbol{P}_{t|t-T} = \boldsymbol{F}_t \boldsymbol{P}_{t-T|t-T} \boldsymbol{F}_t^\mathsf{T} + \boldsymbol{V}_t \boldsymbol{Q}_t \boldsymbol{V}_t^\mathsf{T}, \tag{2b}$$

with f being the non-linear state update function and $\boldsymbol{F}_t$ and $\boldsymbol{V}_t$ being the Jacobians of f with respect to the state and the process noise respectively

$$\boldsymbol{F}_t = \frac{\partial f}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}_{t|t-T}, \boldsymbol{u}_t, \mathbf{0}), \tag{2c}$$

$$\boldsymbol{V}_t = \frac{\partial f}{\partial \boldsymbol{v}}(\hat{\boldsymbol{x}}_{t|t-T}, \boldsymbol{u}_t, \mathbf{0}). \tag{2d}$$

For the **measurement update (correction step)** the equations are:

$$\boldsymbol{S}_t = \boldsymbol{H}_t \boldsymbol{P}_{t|t-T} \boldsymbol{H}_t^\mathsf{T} + \boldsymbol{E}_t \boldsymbol{R}_t \boldsymbol{E}_t^\mathsf{T}, \tag{3a}$$

$$\boldsymbol{z}_t = \boldsymbol{y}_t - h(\hat{\boldsymbol{x}}_t | t-T, \mathbf{0}), \tag{3b}$$

$$\boldsymbol{K}_t = \boldsymbol{P}_{t|t-T} \boldsymbol{H}_t^\mathsf{T} \boldsymbol{S}_t^{-1}, \tag{3c}$$

$$\hat{\boldsymbol{x}}_{t|t} = \hat{\boldsymbol{x}}_{t|t-T} + \boldsymbol{K}_t \boldsymbol{z}_t, \tag{3d}$$

$$\boldsymbol{P}_{t|t} = \boldsymbol{P}_{t|t-T} - \boldsymbol{K}_t \boldsymbol{H}_t \boldsymbol{P}_{t-T} \tag{3e}$$

with h being the correction function, $\boldsymbol{S}_t$ the innovation covariance, $\boldsymbol{z}_t$ the innovation, $\boldsymbol{K}_t$ the Kalman Gain and $\boldsymbol{H}_t$, $\boldsymbol{E}_t$ the Jacobians of h with respect to the state and the measurement noise respectively,

$$\boldsymbol{H}_t = \frac{\partial h}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}_{t|t-T}, \mathbf{0}), \tag{3f}$$

$$\boldsymbol{E}_t = \frac{\partial h}{\partial \boldsymbol{e}}(\hat{\boldsymbol{x}}_{t|t-T}, \mathbf{0}). \tag{3g}$$

For the integration of inertial measurements we selected the control input model showing the best combination of good performance during fast motion while still being reasonably computationally expensive [4]. In the control input model, the inertial measurements are treated as control inputs $\boldsymbol{u}_t^\mathsf{T} = [\boldsymbol{\alpha}_t^\mathsf{T}, \boldsymbol{\omega}_t^\mathsf{T}]$ for the time update of the filter, where $\boldsymbol{\alpha}_t$ is the 3-dimensional acceleration and $\boldsymbol{\omega}_t$ is the 3-dimensional angular velocity input from the IMU.

The state of the filter $\boldsymbol{x}$ is defined as

$$\boldsymbol{x}^\mathsf{T} = [\boldsymbol{s}_w^\mathsf{T}, \dot{\boldsymbol{s}}_w^\mathsf{T}, \boldsymbol{q}_{sw}^\mathsf{T}, \boldsymbol{b}_s^{\alpha\mathsf{T}}, \boldsymbol{b}_s^{\omega\mathsf{T}}], \tag{4}$$

where $\boldsymbol{s}_w$ is the IMU sensor position and $\dot{\boldsymbol{s}}_w$ the velocity, $\boldsymbol{q}_{sw}^\mathsf{T} = [q_w, q_x, q_y, q_z]$ is the sensor orientation in quaternion represenation, $\boldsymbol{b}_s^\alpha$ the accelerometer bias and $\boldsymbol{b}_s^\omega$ the gyroscope bias.

With this particular definition for the state, the equations for the **time update (prediction step)** of the state are

$$\begin{bmatrix} \boldsymbol{s}_{w,t} \\ \dot{\boldsymbol{s}}_{w,t} \\ \boldsymbol{q}_{sw,t} \\ \boldsymbol{b}_{s,t}^\alpha \\ \boldsymbol{b}_{s,t}^\omega \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_{w,t-T} + T\dot{\boldsymbol{s}}_{w,t-T} + \frac{T^2}{2}\boldsymbol{R}_{ws,t-T}(\boldsymbol{\alpha}_t - \boldsymbol{b}_{s,t-T}^\alpha - \boldsymbol{v}_{s,t}^\alpha) + \frac{T^2}{2}\boldsymbol{g}_w \\ \dot{\boldsymbol{s}}_{w,t-T} + T\boldsymbol{R}_{ws,t-T}(\boldsymbol{\alpha}_t - \boldsymbol{b}_{s,t-T}^\alpha - \boldsymbol{v}_{s,t}^\alpha) + T\boldsymbol{g}_w \\ exp\left(-\frac{T}{2}(\boldsymbol{\omega}_t - \boldsymbol{b}_{s,t-T}^\omega - \boldsymbol{v}_{s,t}^\omega)\right) \odot \boldsymbol{q}_{sw,t-T} \\ \boldsymbol{b}_{s,t-T}^\alpha + \boldsymbol{v}_{s,t}^{b\alpha} \\ \boldsymbol{b}_{s,t-T}^\omega + \boldsymbol{v}_{s,t}^{b\omega} \end{bmatrix} \tag{5}$$

For the **correction step** we receive a 6 DoF pose directly from the OT system to use as a measurement. Thus, this input can be transformed to a measurement vector $\boldsymbol{y}_t^\mathsf{T} = [\overline{s}_{w,t}^\mathsf{T}, \overline{q}_{sw,t}^\mathsf{T}]$. Thus, the **measurement update (correction step)** function becomes

$$[\overline{s}_{w,t}^\mathsf{T}, \overline{q}_{sw,t}^\mathsf{T}]^\mathsf{T} = [s_{w,t}^\mathsf{T}, q_{sw,t}^\mathsf{T}]^\mathsf{T} + \boldsymbol{e}_{s,t}^\mathsf{T}. \tag{6}$$

3.3 Fusion Algorithm

In this Section we describe the entire fusion algorithm used in our approach. As previously explained and referring to Figure 3 the purpose of the algorithm is to estimate as accurately as possible the 6 DoF pose at time point $\boldsymbol{D}$ when the available OT measurements only reach time point $\boldsymbol{A}$. At any time point corresponding to $\boldsymbol{A}$ we have available a queue of IMU data $Queue_{\text{imu}}$ and a queue of OT data $Queue_{\text{ot}}$. The timestamps of $Queue_{\text{imu}}$ are shifted by adding the calibration delay Δ_{os} to their timestamps ($t_g^s = t_g^s + \Delta_{os}$). Further we denote an OT pose as $\boldsymbol{y}_t$ and an IMU measurement as $\boldsymbol{u}_t$. The procedure for processing the data is described in Algorithm 1.

Algorithm 1 Fusion process

```
if fusionEKF is not initialized then
    $\boldsymbol{y}_t = pop(Queue_{\text{ot}})$
    Initialize fusionEKF with $\boldsymbol{y}_t$
else
    while $Queue_{\text{ot}}$ is not empty do
        if $head(Queue_{\text{imu}}).t_g^s < head(Queue_{\text{ot}}).t_g^o$ then
            $\boldsymbol{u}_t = pop(Queue_{\text{imu}})$
            fusionEKF time update with $\boldsymbol{u}_t$
        else
            $\boldsymbol{y}_t = pop(Queue_{\text{ot}})$
            fusionEKF correction with $\boldsymbol{y}_t$
        end if
    end while
    if fusionEKF is diverged then
        Set fusionEKF not initialized
    end if
    Set predictionEKF = fusionEKF
    Set $Queue_{\text{imu}} = Queue_{\text{imu}}$
    while $Queue_{\text{imu}}$ is not empty do
        $\boldsymbol{u}_t = pop(Queue_{\text{imu}})$
        predictionEKF time update with $\boldsymbol{u}_t$
    end while
    Apply forward prediction with MotionModel for $\Delta_p$
    Output Pose estimate for time point $D$
end if
```

3.4 Forward prediction with Motion Model

As mentioned already, for dealing with the delay Δ_p there is no available data to use. For this reason the only possibility is to assume that some of the parameters of the motion remain constant and predict the future position based on that. In our implementation we used a constant velocity and constant angular velocity model. Clearly, the constant velocity assumption becomes less accurate as Δ_p increases. Furthermore it is impossible to predict an abrupt change of direction in the tracking. We obtain the velocity value used from the state of the **prediction EKF** filter ($\dot{\boldsymbol{s}}_{w,t}$) and the angular velocity by calculating an average over a number of past IMU angular velocity measurements($\boldsymbol{\omega}_t$) to obtain a less noisy value.

4 EVALUATION

In this section we present experimental results verifying that the implemented fusion framework performs with the expected functionality in a practical setup.

Table 1: EKF experiments noise levels

#	variance σ^2
acceleration noise $\boldsymbol{v}_{s,t}^{\alpha}$	4×10^{-2}
angular velocity noise $\boldsymbol{v}_{s,t}^{\omega}$	2×10^{-3}
acceleration bias noise $\boldsymbol{v}_{s,t}^{b\alpha}$	1×10^{-10}
angular velocity bias noise $\boldsymbol{v}_{s,t}^{b\omega}$	1×10^{-10}
OT position noise $\boldsymbol{e}_{s,t}^{\bar{s}}$	1×10^{-8}
OT rotation noise $\boldsymbol{e}_{s,t}^{\bar{q}}$	1×10^{-9}

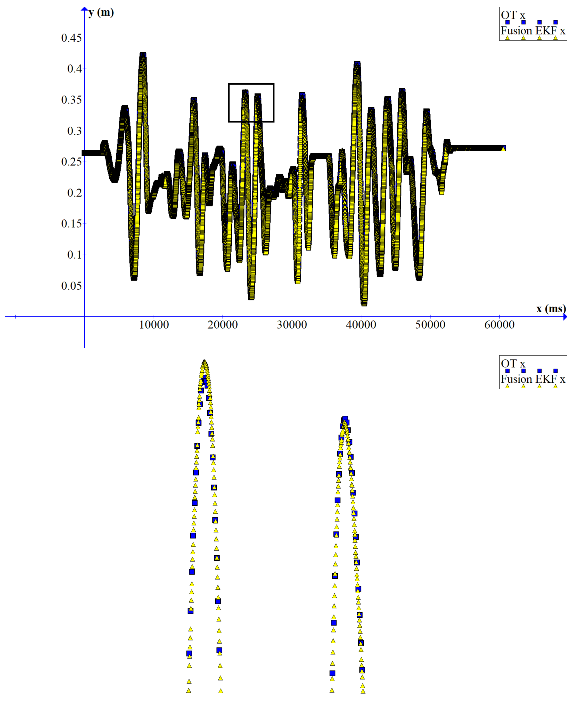

Figure 4: OT measurements (blue squares) and fusion EKF measurements (yellow triangles), entire graph and zoomed in for detail

4.1 Experimental Setup and calibration

We conducted our experiments for IMU frequencies of 250 Hz and 500 Hz of the IMU. Lower IMU frequencies are not of interest in terms of fusion since only marginal benefit would be gained. Higher frequencies produce more measurements and allow the EKFs to learn the sensor biases and covariances faster and are be less affected by noisy IMU data. The OT system frequency was 62,5 Hz. From the calibration procedure a sample shift of 12 and 21 was estimated between IMU and OT at IMU frequencies of 250 and 500 Hz respectively corresponding to a time shift Δ_{os} of 42 and 48 ms between the IMU and OT. The time delay Δ_p was empirically estimated to be around 70 ms.

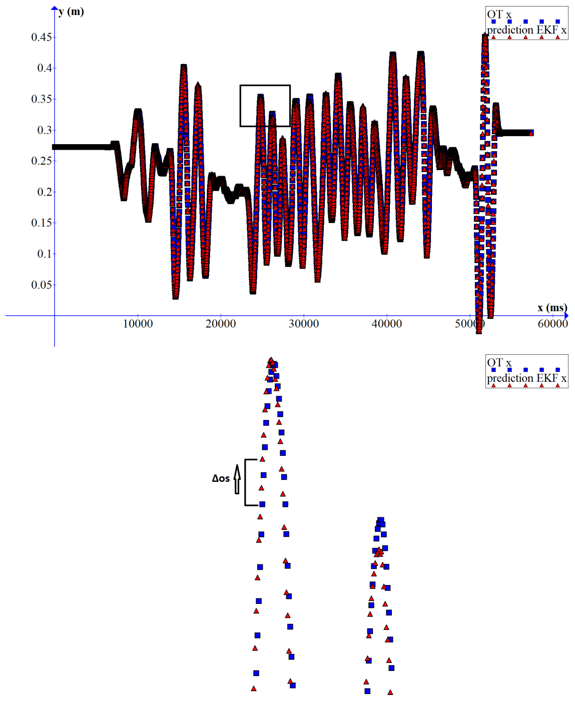

Figure 5: OT measurements (blue squares) and prediction EKF measurements (red triangles), entire graph and zoomed in for detail

For the EKF to function properly the noise levels have to be tuned accordingly. The determination of the noise levels is not a straightforward task. For the IMU it has been previously proposed to measure the noise from the variance of the measurements while the device is stationary, however we observed that there is a large increase of the noise levels, especially in acceleration measurements when the device is moving. Thus we estimated the IMU noise levels by inspection of sequences of the device in motion. For the OT, the output of the system room calibration gives an indication on the accuracy of the tracking. In Table 1 we present the noise variances that we used for the EKF in our experiments.

4.2 Experimental Results

In Figure 4 we show the x-axis position track from the fusion EKF (yellow triangles) compared to the x-axis position given by the OT system (blue squares). We show the graph in 2 different zoom levels to demonstrate both the robustness over time and the accuracy of single EKF outputs. The EKF is able to follow the correct track (given by the OT) at all times during the experiment. The EKF filter is well tuned and practically follows the correct path at a a higher frequency than the OT (frequency of the IMU, here 250 Hz).

In Figure 5 we show the x-position of the OT output (blue squares) and the output given by the prediction EKF using only the extra available IMU measurements (red triangles). We can see that the prediction EKF output is accurately about 2 OT samples ahead in time compared to the OT output, which corresponds approximately to the imu shift Δ_{os}. We can thus show that we can succesfully compensate for the delay Δ_{os} using the additional available imu data in the prediction EKF.

In Figure 6 we show the OT output (blue squares) and the output from the motion model forward prediction (green triangles). Similarly to Figure 5 with the use of the motion model we are able to predict more OT measurements ahead to additionally cover the delay Δ_p so that the entire delay $\Delta_{os} + \Delta_p$ is covered. However, since the motion model prediction is done based on the filter state and

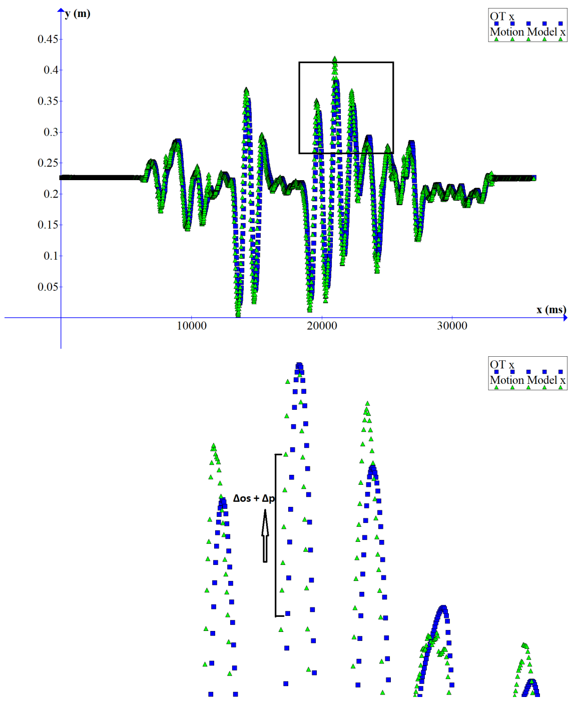

Figure 6: OT measurements (blue squares) and forward predictions using a motion model (green triangles), entire graph and zoomed in for detail

without having actual data available we are not able to correctly follow abrupt direction changes. When the motion is linear (constant velocity model is used) the forward predictions are very accurate.

Note that in the presented experimental results we only show the estimated position instead of all 3 axes and the estimated orientation. However, in a tightly coupled visual-inertial fusion scheme as the one applied here, showing the x-axis is sufficient since if one of the pose parameters has errors, the entire filter will not function at all.

5 Conclusion

We presented an EKF-based framework for fusion of unsynchronized optical trackers and inertial sensors, mainly aiming to reduce the delay of correctly positioning 3D augmentations on a HMD based on the computed 6 DoF pose. The experimental evaluation shows that the approach is able to estimate the correct pose ahead in time compared to the optical tracker with a slight reduction in accuracy. This is inevitable since the estimation is based first on noisy IMU data and then on a motion model without any available data. The delay reduction achieved however, greatly improves the user perception of the stability of the rendered scene. Future work includes experimentation and comparison with other hardware setups, as well as investigating methods to determine and measure the delays in the system more accurately.

References

[1] http://www.ar-tracking.com/products/.

[2] W. Barfield. *Fundamentals of wearable computers and augmented reality*. CRC Press, 2015.

[3] G. Bleser and D. Stricker. Using the marginalised particle filter for real-time visual-inertial sensor fusion. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pages 3–12. IEEE, 2008.

[4] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual–inertial sensor fusion. *Computers & Graphics*, 33(1):59–72, 2009.

[5] M. Huber, M. Schlegel, and G. Klinker. Application of time-delay estimation to mixed reality multisensor tracking. *Journal of Virtual Reality and Broadcasting*, 11(3), 2014.

[6] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[7] M. Li and A. I. Mourikis. High-precision, consistent ekf-based visual–inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[8] E. Marchand, H. Uchiyama, and F. Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 2015.

[9] M. Miezal, G. Bleser, D. Stricker, and J. Tümler. Towards practical inside-out head tracking for mobile seating bucks. In *Proceedings of International Symposium on Mixed and Augmented Reality (ISMAR). Atlanta, USA*, 2012.

[10] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness. *IEEE Transactions on intelligent transportation systems*, 11(2):300–311, 2010.

[11] M. Pancholi, S. Dimitrov, N. Schmitz, S. Lampe, and D. Stricker. Relative translation and rotation calibration between optical target and inertial measurement unit. In *S-Cube International Conference on Sensor Systems and Software*, 2016.

[12] J. Rambach, A. Tewari, A. Pagani, and D. Stricker. Learning to fuse: A deep learning approach to visual-inertial camera pose estimation. In *Mixed and Augmented Reality (ISMAR), 2016 IEEE International Symposium on*, 2016.