Sensor Fusion for Motion Estimation of Mobile Robots with Compensation for Out-of-Sequence Measurements

Berntorp, Karl; Årzén, Karl-Erik; Robertsson, Anders

2011

Citation for published version (APA):
Berntorp, K., Årzén, K.-E., & Robertsson, A. (2011). *Sensor Fusion for Motion Estimation of Mobile Robots with Compensation for Out-of-Sequence Measurements*. 211-216. Paper presented at 2011 11th International Conference on Control, Automation and Systems.

Total number of authors:
3

2011 11th International Conference on Control, Automation and Systems
Oct. 26-29, 2011 in KINTEX, Gyeonggi-do, Korea

Sensor Fusion for Motion Estimation of Mobile Robots with Compensation for Out-of-Sequence Measurements

Karl Berntorp[1], Karl-Erik Årzén[1], Anders Robertsson[1]

[1]Department of Automatic Control, Lund University, Box 118, Lund, Sweden
(E-mail: {berntorp, karlerik, andersro}@control.lth.se)

Abstract: The position and orientation estimation problem for mobile robots is approached by fusing measurements from inertial sensors, wheel encoders, and a camera. The sensor fusion approach is based on the standard extended Kalman filter, which is modified to handle measurements from the camera with unknown prior delay. A real-time implementation is done on a four-wheeled omni-directional mobile robot, using a dynamic model with 11 states. The algorithm is analyzed and validated with simulations and experiments.

Keywords: Extended Kalman filter, out-of-sequence, localization, estimation, mobile robotics, sensor fusion

1. INTRODUCTION

How to combine internal and external sensors for position and velocity estimation in real-time is crucial for autonomous mobile robots. Methods based on dead-reckoning using wheel encoders, inertial measurement units (IMU), and global positioning systems (GPS) are common to localize the robot, see [1]. The major drawback with using GPS is that it may not be an option when navigating indoors. The dead-reckoning approaches based on wheel encoders and inertial measurement units all suffer from integration and accumulation of erroneous signals, for example caused by noise or bias in the sensors.

To get rid of the drift in dead-reckoning, vision can be used to give absolute position measurements which are then combined with the other sensors. Vision for mobile robot navigation is by no means a new concept, see the survey [2] which covers the developments from the 1980's to the late 1990's. Vision in combination with inertial sensors has been investigated before, see [3] for an introduction, and [4] for multi-rate fusion of inertial and visual sensors.

In multisensor systems there are usually different time delays associated with the various sensors. The time delays can arise from, for example, communication delays. When vision algorithms are involved, it is the computation time concerned with finding features on the robot and transforming them to world coordinates that causes the main part of the time delay. This will result in that position estimates that arrive at the same time as accelerometer measurements will be delayed in time, causing out-of-sequence measurements (OOSM). If the delay is constant, standard state augmented Kalman filters can be used to produce an estimate. However, in practical situations the time delay varies over time, which implies that the constant delay model may not be appropriate. Several articles have been produced on the subject of how to handle delayed measurements. In [5], the optimal solution to the OOSM estimation problem where the delay was assumed to be between the last two samples was presented. One assumption was that the state transition matrix was nonsingular. Some comparisons with two suboptimal algorithms were also given, and it was found that the suboptimal approaches performed nearly as good. Suboptimal should here be interpreted in the mean-square sense. A best linear unbiased estimator, also in the mean-square sense, was presented in [6], where there were no restrictions on the state transition matrix. Furthermore, the only restriction on the time delay was that its maximum was known. The drawback with this approach is that it is computationally demanding. The work of [5] was extended in [7], where the suboptimal algorithms were able to handle an arbitrary delay, with only the maximum delay known in advance. In [8], the algorithms were also able to handle biases in the delayed measurements.

This paper builds on [7] to estimate the position and orientation of a mobile robot in real-time. It has its main focus on fusing inertial sensors (accelerometers and rate gyroscopes), wheel encoders, and vision measurements using an extended Kalman filter approach with OOSM compensation, where only an arbitrary maximum time delay is known a priori. A time stamp is attached to every camera measurement. The proposed extension to the standard Kalman filter algorithm only contributes a small portion to the computational effort of the estimation procedure. The extension from [7] and [5] to the work done here is that the algorithm is implemented in a real-time environment, with more sensors and a more complex process model. The justifications for fusing these sensors and not using a camera alone are the following:

- A camera gives position and angle measurements with a rather low frequency. Also, for several reasons any vision algorithm experiences problems from time to time with low quality or no vision data. This could, e.g., be the case during fast motion.
- The IMU and wheel encoders give high rate acceleration, angular rate, and velocity measurements. However, these sensors alone will cause drift in the long term, which is corrected for by the position measurements of the camera.

This work has partially been funded by SSF ENGROSS and by the LCCC center.

1.1 Related Work

Fusion of IMU and vision for state estimation of autonomous vehicles has been studied before. In [9], IMU data is fused with vision in a SLAM framework, employing an extended Kalman Filter. In [10], OOSM state estimation considering wheel encoders and a GPS with time delay is performed and verified by simulations. Another work is [11], where the OOSM problem considering that the delay is within a fraction of a sample, as explained in [5], is solved. An autonomous vehicle's target is estimated using a constant velocity model with the help of vehicle mounted cameras, and inertial sensors are used to estimate the stereo cameras' motion.

2. EXPERIMENTAL SETUP

The robot, which is built at IPA Fraunhofer in Stuttgart, see Fig. 1, has previously been used in the DESIRE project [12]. It is a four-wheeled omni-directional mobile robot equipped with eight motors, two for each wheel. For the experiments in this work it has been equipped with a six degrees of freedom IMU from Xsens, see [13] for more information, which is aligned with the coordinate frame of the robot. The IMU provides measurements with a rate of 50 Hz. A calibration procedure has been used to calibrate for imperfections in the physical alignment of each component, gains, offsets and temperature relations. Using the calibration the accelerometer and gyro vectors, expressed in the IMU's local coordinate frame, are computed using an onboard processor.

The wheel encoder measurements are the foundation of the velocity measurements; based on the kinematics of the robot it is possible to extract the translational velocity from the robot with a rate of 20 Hz.

A roof mounted camera is situated above the robot's workspace. From the camera the absolute position of the robot, as well as the orientation can be calculated. The algorithms for object tracking and feature detection fall outside the scope of this article, but are surveyed in [14]. In this work the camera vision algorithm gives position measurements with a frame rate between 1.5-2 Hz, but that differs somewhat due to jitter and computational demands. Furthermore, the vision algorithm provides a timestamp.

3. MODELING

The system is modeled by a state space model. At time t the vehicle state is denoted x_t, and the input is denoted u_t. This implies the state space model

$$x_{t+1} = g(x_t, u_t) + v_t, \tag{1}$$

where v_t is Gaussian process noise. At time t the vehicle measurements are given by

$$y_t = h(x_t) + e_t, \tag{2}$$

where e_t is Gaussian measurement noise.

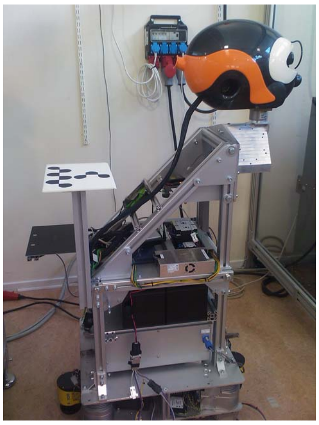

Fig. 1 The mobile robot used for the experiments. The IMU is situated below the black tablet to the left in the picture, and the feature detection pattern used for the vision algorithm is the black and white paper, also to the left in the picture.

3.1 Coordinate Frames

When working with moving objects different coordinate frames have to be used. For the combined vehicle and camera system three coordinate systems are of particular interest:

1. World (W): This frame is considered an inertial frame and is fixed to the surroundings of the vehicle. The robot pose is estimated with respect to this frame.
2. Body (B): This frame is fixed to the robot, with its origin placed at the position of the IMU. Both the accelerations and velocities coming from the IMU and the wheel encoders are resolved in this frame.
3. Camera (C): The camera measurements are given in this frame, which is fixed relative to W.

Assume that the position of B expressed in W is p_w, and that the rotation is given by a roll, ϕ, a pitch, θ, and a yaw, ψ. Then the transformation between the frames is done through an addition of p_w followed by a rotation given by the rotation matrix

$$R_B^W = \begin{pmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix},$$

where c_θ and s_θ are short for $\cos\theta$ and $\sin\theta$, respectively.

3.2 Process Model

In order to simplify the modeling it is assumed that the robot moves in a plane, which implies that only one parameter is needed to estimate the orientation, whereas all position, velocity, and acceleration vectors are two-dimensional. With the states as described in Table 1, the state vector at time t is given by the 11-dimensional vector

$$x_t = \begin{pmatrix} p_t & v_t & a_t & b_{a,t} & \psi_t & \omega_t & b_{\omega,t} \end{pmatrix}^T.$$

By modeling the process as a constant acceleration process the model, Eq. (1), is

$$x_{t+1} = Fx_t + v_t, \tag{3}$$

Table 1 Vehicle States with description and dimension.

State	Description	Dim.
p	Vehicle position in world coordinates.	2
v	Vehicle velocity in world coordinates.	2
a	Vehicle acceleration in world coordinates.	2
b_a	Bias state for acceleration measurement.	2
ψ	Yaw angle relative to the world frame.	1
ω	Yaw velocity in world coordinates.	1
b_ω	Bias state for yaw velocity measurement.	1

where

$$F = \begin{pmatrix} I & TI & \frac{T^2}{2}I & 0 & 0 & 0 & 0 \\ 0 & I & TI & 0 & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & TI & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I \end{pmatrix}.$$

The submatrices I and 0 are of appropriate dimensions. The process noise v_t is assumed independent and Gaussian, according to

$$\begin{aligned} &v_t \sim \mathcal{N}(0, Q), \\ &Q = GQ^vG^T, \end{aligned} \tag{4}$$

where

$$G = \begin{pmatrix} \frac{T^3}{6}I & 0 & 0 & 0 \\ \frac{T^2}{2}I & 0 & 0 & 0 \\ TI & 0 & 0 & 0 \\ 0 & TI & 0 & 0 \\ 0 & 0 & \frac{T^2}{2} & 0 \\ 0 & 0 & T & 0 \\ 0 & 0 & 0 & T \end{pmatrix},$$

and

$$Q^v = \operatorname{diag}(q_a \quad q_{ba} \quad q_\omega \quad q_{b\omega}).$$

All q-variables are process noise variance parameters.

3.3 Measurement Model

The measurement model as described in Eq. (2) is given by the eight-dimensional vector

$$h(x_t) = \begin{pmatrix} p_t \\ R_W^B v_t \\ R_W^B a_t + b_{a,t} \\ \psi \\ \omega_t + b_{\omega,t} \end{pmatrix}. \tag{5}$$

Also the measurement noise e_t is assumed to be independent and Gaussian, leading to

$$e_t \sim \mathcal{N}(0, Q^e),\; Q^e = \operatorname{diag}\big(q_p \quad q_v \quad q_{ma} \quad q_\psi \quad q_{m\omega}\big).$$

The matrix Q^e is determined experimentally.

The velocity measurements are resolved in the robot's geometric center. Therefore it has to be translated to the position of the IMU in order to be used as a measurement. Assume that the IMU is positioned at $p_{IMU} = \big(x_{IMU} \quad y_{IMU}\big)^T$ metres away from the geometric center, and that the velocity of the geometric center is $v = \big(v_x \quad v_y\big)^T$. Then the transformed velocity at the IMU is

$$v_{IMU} = v + \omega \times p_{IMU},$$

where ω is the angular velocity of the robot.

4. EXTENDED KALMAN FILTER DESIGN

In order to solve the state estimation problem an extended Kalman filter (EKF) is employed, see [15] for a thorough investigation. The algorithm consists of two steps; the time update step and the measurement update step. In the time update step Eq. (3) is used to update the states and covariances as

$$\hat{x}_{t+1|t} = F\hat{x}_{t|t}, \tag{6}$$

$$P_{t+1|t} = FP_{t|t}F^T + Q, \tag{7}$$

where $\hat{x}_{t|k}$ should be interpreted as the estimate of x at time t given measurements up to time k. When a measurement arrives Eq. (8) is used to update the mean of the state estimate. To update the covariance estimate, the measurement model is linearized at $\hat{x}_{t|t-1}$, i.e., the mean from the previous time update step. The Kalman filter equations for the measurement update are now

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t(y_t - h(\hat{x}_{t|t-1})), \tag{8}$$

$$P_{t|t} = (I - K_tH_t)P_{t|t-1}, \tag{9}$$

where H_t is the linearization of Eq. (5) around $\hat{x}_{t|t-1}$, and $K_t = P_{t|t-1}H_t^T(H_tP_{t|t-1}H_t^T + Q_e)^{-1}$. When no measurement arrives, the measurement update equations are simply ignored.

4.1 Handling Out-of-Sequence Measurements

Since the vision algorithm only produces about two position and angle measurements per second on average, a time varying delay has to be considered. As already mentioned there are several approaches to account for measurements that arrive delayed in time. The following approach, which has been derived in [7], assumes the retrodicted noise to be zero. This assumption implies that the algorithm is only suboptimal, but as pointed out in [7] it performs very similar to the optimal algorithm while reducing the memory consumption. The algorithm is explained below: Denote the time delay of the measurement with l. It is assumed that l is a multiple of the sampling time T of the estimation algorithm, that is $(t - l)/T = \tau$, $\tau = 0, 1, \cdots$, where t is the present time. Also the maximum delay, l_{max}, is known. The retrodiction of the state to τ from t is

$$\hat{x}_{\tau|t} = F_{\tau,t}\hat{x}_{t|t} = F^{-\tau}\hat{x}_{t|t}. \tag{10}$$

The covariance for the state retrodiction is

$$P_{\tau|t} = F_{\tau,t}\left[P_{t|t} + P^{vv}_{t,\tau|t} - P^{xv}_{t,\tau|t} - (P^{xv}_{t,\tau|t})^T\right]F^T_{\tau,t},$$

where

$$P^{vv}_{t,\tau|t} = Q_{t,\tau},$$
$$P^{xv}_{t,\tau,t} = P_{t|t} P_{t|t-l} P^{vv}_{t,\tau|t}. \tag{11}$$

$P^{vv}_{t,\tau|t}$ is the covariance of the process noise for the retrodiction interval, and $P^{xv}_{t,\tau,t}$ is the cross-covariance between the cumulated process noise and the current state. The covariance between the state at time t and the delayed measurement is calculated as

$$P^{xy}_{t,\tau|t} = \left[P_{t|t} - P^{xv}_{t,\tau|t}\right] F^T_{\tau,t} H^T_\tau.$$

Now, the gain used for the EKF update is

$$K_{t|\tau} = P^{xy}_{t,\tau|t} S^{-1}_\tau,$$

where S_τ is

$$S_\tau = H_\tau P_{\tau|t} H^T_\tau + Q^e.$$

The new estimate $\hat{x}_{t|\tau}$ due to the OOSM measurement is

$$\hat{x}_{t|\tau} = \hat{x}_{t|t} + K_{t|\tau}\left[y_\tau - \hat{y}_{\tau|t}\right], \tag{12}$$

where the predicted OOSM is

$$\hat{y}_{\tau|t} = H_\tau \hat{x}_{\tau|t}, \tag{13}$$

and the retrodicted state is given by Eq. (10). Finally, the covariance for the updated state estimate is

$$P_{t|\tau} = P_{t|t} - P^{xy}_{t,\tau|t} S^{-1}_\tau (P^{xy}_{t,\tau|t})^T. \tag{14}$$

Note that no storage of the old state estimates is needed. The only stored covariance matrix that is needed for the algorithm is $P_{t|t-l}$, which is used in Eq. (11). The storage requirements for the algorithm in addition to the standard EKF is :

- Storage of $P_{t-l_{max}|t-l_{max}}, \ldots, P_{t-1|t-1}$ requires, due to symmetry of P, $\frac{l_{max}n(n+1)}{2}$ scalars, where n is the number of states.

4.2 Summarizing the Algorithm

The algorithm can be summarized as follows: Start with updating the state estimate and covariance with Eqs. (6) and (7). When velocity and/or IMU measurements arrive, update the state estimate and covariance with Eqs. (8) and (9). As soon as a delayed camera measurement arrives, also use Eqs. (12)$\sim$(14).

5. SIMULATION RESULTS

Results from two different simulations will be shown next. The sample time is $T = 0.02$ s, which equals the rate that the IMU delivers data with. The noise for the IMU is set to white Gaussian noise, where the variance has been calculated through experiments. Likewise, the noise for the velocity measurements stemming from the wheel encoders is modeled as white Gaussian, and are also estimated experimentally. Some small biases are added to both the accelerometer and gyro sensors. The time delay is modeled as uniformly distributed random numbers between 0.5 and 0.7 seconds. The filter starts up at time $t_0 = 0$ with initial state estimate based on the first camera measurement. The covariance is initialized to equal the process noise matrix Q, which is defined in Eq. (4).

The EKF/OOSM algorithm was compared with the EKF without any compensation, denoted EKFwoC, with an EKF with constant time delay compensation equaling the mean time delay, EKFwC, and with an EKF with a worst case time delay compensation, called EKFwWCC. In the first simulation the path was circular. To see the discrepancy between the different approaches, the mean squared error between the real positions and the estimated positions are shown in Table 2. From the table it is obvious that the EKF/OOSM combination is superior to the other approaches despite not demanding much more computational effort. The filter with constant time compensation, EKFwC, performed second best but still almost a factor two worse.

Table 2 Mean-squared error of a circular path for various estimation techniques.

Algorithm	**MSE**
EKF/OOSM	0.7
EKFwoC	14.85
EKFwC	1.21
EKFwWCC	3.95

To emphasize the difference in precision of the estimates over time, a plot of the estimated angle and the measured angle as a function of time are shown in Fig 2. It is clear that while the EKF without compensation lags behind significantly, the EKF with worst case scenario compensation overcompensates for the delay most of the time. However, the EKF/OOSM algorithm is consistent. It is important to note that while the differences between the approaches might seem small at first glance, they could be crucial when, for example, trying to perform high precision tasks with a robot arm mounted on the base of the mobile platform. As a final validation, zoomed in positions in the forward direction are shown in Fig. 3 when the robot is simulated to drive along a straight line. Once again the superior performance of the OOSM-compensation algorithm compared to the others is seen. While the EKF with worst case compensation tends to overcompensate for the delay, and the standard EKF always is a bit behind, the EKF with OOSM compensation still performs consistently. Noticeable is that all approaches except the OOSM-compensation-algorithm experience jumps in the position estimation, due to that they compensate for the wrong delay. The results from the average delay compensation are not shown in order not to clutter the figures excessively. However, this strategy performed well in all tests, with a degradation in performance of the same order as in Table 2.

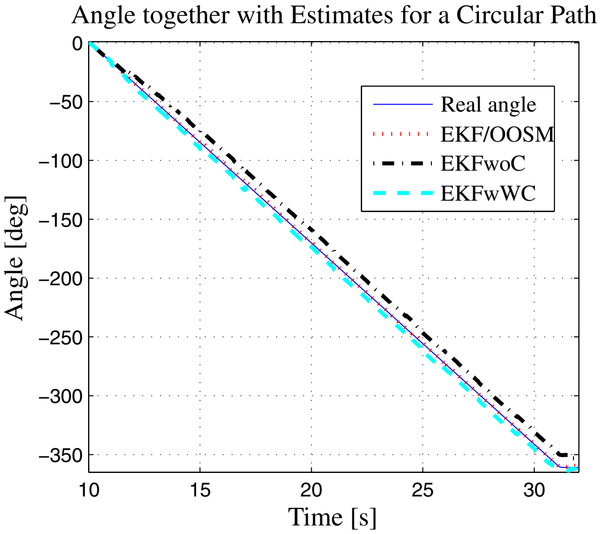

Fig. 2 Angle estimates for various EKF algorithms together with the real angle for a circular path. Note that the real angle is practically the same as for the EKF/OOSM.

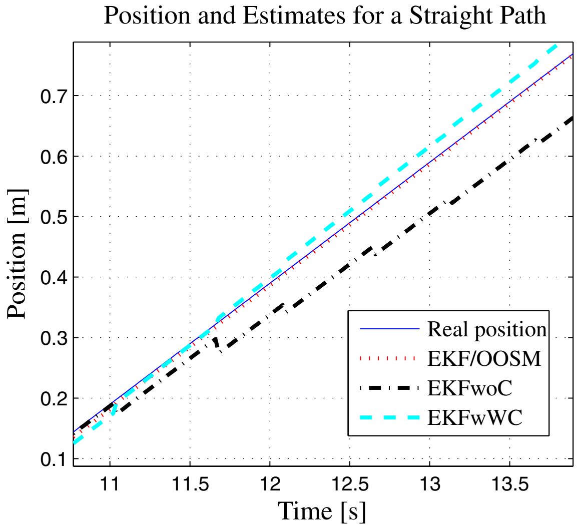

Fig. 3 Position estimates for various EKF algorithms together with the real values for a straight path.

6. EXPERIMENTAL RESULTS

In order to illustrate the real-time performance of the EKF/OOSM algorithm, it was implemented in Simulink with embedded Matlab functions. The vision algorithm was written in Matlab. The velocity and IMU data was sent through a Java implemented socket connection to the estimation algorithm. As previously mentioned the position measurement has its own timestamp. At this stage, the IMU data and the velocity measurements are considered to arrive at the sampling instants. Therefore the estimates and the measured states could differ up to approximately a sample period in synchronization.

The robot was manually driven around using a joystick, with an excerpt of the path displayed in Fig. 4. The estimates seem to cohere well to the measured positions, although some discrepancies exist. Angle estimates are shown in Fig. 5, together with the measured values. From this plot, the estimation can be verified to work well. The estimated angle is within one degree from the measured angle most of the time, even during fast turning. However, there are some disrepancies in this figure as well. The most probable reasons for this are errors in the timestamps and the fact that the camera is the only sensor that comes with a timestamp, violation of white noise assumptions, and dynamic effects due to the height of the robot. The dynamics of the robot introduce nonzero acceleration and gyro measurements. Furthermore, the vision algorithm assumes planar motion, so it should not always be assumed to deliver ground truth values.

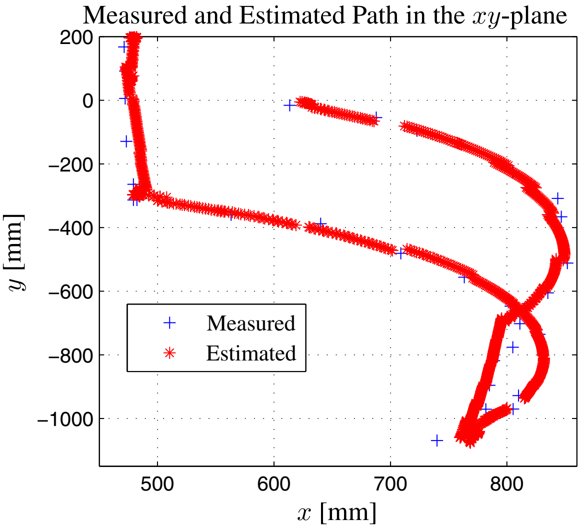

Fig. 4 Trajectory in the xy-plane of a path for the Kalman filter approach with OOSM-compensation, shown as stars. The camera measurements are shown as crosses.

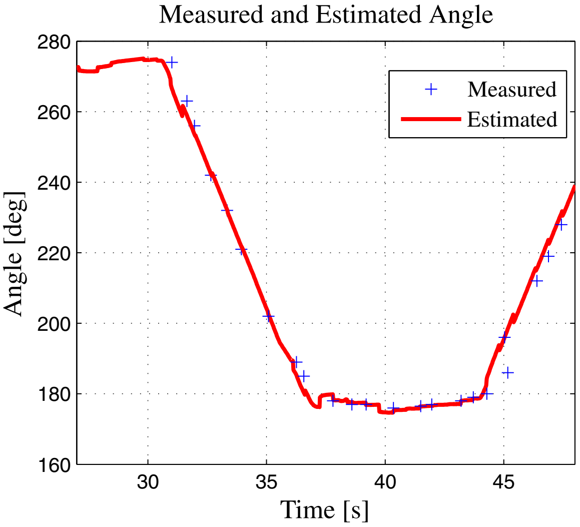

Fig. 5 Angle estimates for the path in Fig. 4 with the Kalman filter approach using OOSM-compensation, together with the measured angle. The delays in the camera measurements are accounted for.

In Fig. 6 the forward position is shown for further validation of the algorithm. The path taken here is that of a straight line, so the angle and lateral position estimates are not worthwhile to exhibit. The estimates are very close to the measured positions at all times, often within 2-3 millimeters. The largest differences can be seen when the robot suddenly changes speed, differences which probably are due to the same reasons as before. Note that although only results for the EKF with OOSM-compensation have been shown for two different scenarios many different trajectories have been experimented with, and the algorithm performs similarly in all tests. Also note that the timestamps have been used to move the corresponding measurement to its correct place in time in all figures.

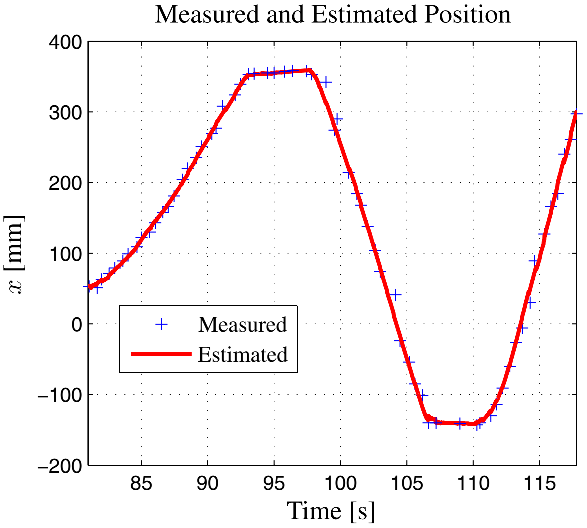

Fig. 6 Position estimates in the forward direction for a straight path using the Kalman filter approach with OOSM-compensation, together with the measured positions. The delays in the camera measurements are accounted for.

7. CONCLUSIONS AND FUTURE WORK

A sensor fusion approach to motion estimation has been implemented. An 11-state constant acceleration model, including three bias states, has been used. Because of fusion from different sensors, the established tracking system applies the l-step lag OOSM suboptimal state estimation solution from [7]. The results in a perfect environment have been verified in simulation, and very satisfactory results have been shown in experiments. For future work, the OOSM solution may be applied to the other sensors as well. Furthermore, the algorithm is going to be adopted in a SLAM setting where onboard cameras will be used for map building, and the tracking system will be used for improved algorithm speed.

REFERENCES

[1] J. Yi, J. Zhang, D. Song, and S. Jayasuriya, “Imu-based localization and slip estimation for skid-steered mobile robots,” *Proc. of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[2] G. N. DeSouza and A. C. Kak, “Vision for mobile robot navigation: A survey,” *IEEE, TRANS. PAMI*, vol. 24, no. 2, pp. 237–267, 2002.

[3] P. Corke, J. Lobo, and J. Dias, “An introduction to inertial and visual sensing,” *The International Journal of Robotics*, vol. 26, pp. 519–535, 2007.

[4] L. Armesto, S. Chroust, M. Vincze, and J. Tornero, “Multi-rate fusion with vision and inertial sensors,” *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 193–199, 2004.

[5] Y. B. Shalom, “Update with out-of-sequence measurements in tracking: exact solution,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 3, 2002.

[6] K. Zhang, X. Li, and Y. Zhu, “Optimal update with out-of-sequence measurements,” *Trans. Sig. Proc.*, vol. 53, pp. 1992–2004, June 2005.

[7] Y. Bar-Shalom, H. Chen, and M. Mallick, “One-step solution for the multistep out-of-sequence-measurement problem in tracking,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, pp. 27–37, Jan. 2004.

[8] S. Zhang, Y. Bar-Shalom, and G. Watson, “Tracking with multisensor out-of-sequence measurements with residual biases,” in *Information Fusion (FUSION), 2010 13th Conference on*, pp. 1–8, July 2010.

[9] G. Nuetzi, S. Weiss, D. Scaramuzza, and R. Siegwart, “Fusion of imu and vision for absolute scale estimation in monocular slam,” *Journal of Intelligent and Robotic Systems*, vol. 61, pp. 287–299, 2011.

[10] M. Choi, W. K. Chung, S. Yi, W. Choi, and K.-S. Chang, “State estimation with delayed observations considering uncertainty in time,” *International Conference on Ubiquitous Robots and Ambient Intelligence*, 2007.

[11] Z. Jia, A. Balasuriya, and S. Challa, “Sensor fusion-based visual target tracking for autonomous vehicles with the out-of-sequence measurements solution,” *Robotics and Autonomous Systems*, vol. 56, pp. 157–176, Feb. 2008.

[12] U. Reiser, R. Klauser, C. Parlitz, and E. Verl, “Desire web 2.0- integration management and distributed software development for service robots,” *Proc. of the 14th International Conference on Advanced Robotics (ICAR)*, 2009.

[13] N. Xsens Technologies B.V., “Mti and mtx user manual and technical documentation,” 2010.

[14] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *ACM Comput. Surv.*, vol. 38, pp. 13+, Dec. 2006.

[15] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Prentice Hall, 1979.