Path Reconstruction based on Gyroscope Bias Estimation using GPS

István Engedy, Gábor Horváth
Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
engedy@mit.bme.hu, horvath@mit.bme.hu

***Abstract*—This paper presents a method for path reconstruction from logged MEMS gyroscope and odometer measurements, through the estimation of the gyroscope bias from GPS data. The bias estimation is done by minimizing the distance between the reconstructed path and the external reference positions (GPS) by tuning the bias with common minimization methods. Both gradient based and quasi-Newton minimization methods are investigated. An efficient quaternion based mathematical model of the problem is presented. The path produced by the proposed method has higher spatial and temporal resolution than the GPS path, it is more accurate than a path from a purely inertial measurement unit and it is at least as accurate as the GPS path, according to real-life experiment.**

Keywords-gyroscope bias; path reconstruction; GPS;

I. Introduction

In the recent years the price of inertial measurement units (IMU) based on MEMS technology dropped rapidly, allowing a wide range of professional and hobbyist applications of these sensors: control system of unmanned aerial vehicles e.g. in [1, 2], assisting the GPS in car navigation e.g. in [3, 4, 5], indoor navigation of both robots e.g. in [6] or humans in [7], motion tracking or motion capture in [8], and many others.

IMUs usually consist of a gyroscope and an accelerometer that measure the angular velocities and the accelerations. These two inertial sensors are usually combined with magnetometers that measure the Earth's magnetic field, with barometers that measure the air pressure that can be used to determine the altitude. When it is possible GPS is also often used. Occasionally these are also combined with application specific sensors (e.g. odometer). Sensor fusion is required to combine these into a virtual sensor that is able to measure both the precise position and orientation.

Most applications where IMU is used, focus on either tracking the motion of the object in order to control it in real-time or storing the sensor measurements to later reconstruct its trajectory for analysis. The most often used methods for this challenge are extended Kalman filters (EKF) [1, 4, 5, 7, 8], particle filters (PF) [3, 6, 11] and model predicting controllers (MPC) [2].

Both EKF and PF track the probability distribution of the state of the system. This is a Gaussian in the case of Kalman filters, and it is an arbitrary distribution in PFs, modeled with a set of particles. Although these methods are considered as state of the art methods, they have some limitations. EKF must be initialized with correct values, otherwise it will fail to converge. PF does not suffer from this limitation, but it is computationally intensive.

Since both methods are online, they can estimate the state of the modeled system only from the past observations. This means that when some of the sensors cannot be sampled with the same frequency as others, there will be iterations with some sensory inputs missing, just like in [1].

Consider, for example, the following scenario. The orientation and position of a device are to be determined. Using an IMU by itself is not sufficient, since only dead reckoning could be achieved with it, meaning that the uncertainty of the state of the system would grow indefinitely over time. It must be combined with some absolute position sensor, like a GPS. IMUs can be sampled with 100 Hz frequency, but GPS usually can be sampled only with 1 Hz. This means that only each 100th iteration would contain an update with a GPS measurement. The uncertainty of the state in the intermediate iterations grows till the next GPS measurement. This way the 99th iteration has the largest uncertainty, but it is only 1 iteration away from a GPS measurement in the 100th iteration. If this GPS measurement would have been known beforehand in the 99th iteration, the uncertainty could have been much smaller.

In the case of online applications, where immediate results are required, like control tasks, this is the best option we have. However, there are other applications, like [8], where delayed or even offline results are sufficient. In such applications, it might be worth it to use other methods for the sensor fusion, which can take all data into consideration.

The field of mobile robot navigation has a similar problem too. As most mobile robots are designed to operate indoors, the use of GPS for the solution of the localization problem is not possible. Not just because of the blocked satellite signal, but because of its insufficient accuracy, as the robot might need to navigate in narrow spaces. One of the most basic localization methods is dead reckoning using odometers or IMUs. In these cases the measurements are integrated over time, causing the uncertainty to increase continuously. To overcome this

This work was partially supported by the ARTEMIS JU and the Hungarian National Research, Development and Innovation Fund in the frame of the R5-COP (Reconfigurable ROS-based Resilient Reasoning Robotic Cooperating Systems) project, and was partly sponsored by the project ref. no. GOP-2011-1.1.1 (Pattern recognition and data analysis methodology applied to data collected using different diagnostic systems).

problem, these dead-reckoning methods are used together with other localization methods, which sense the environment of the robot.

In these cases some other sensors like cameras or range finders detect so-called landmarks and measure the distance between them and the robot. Sometimes the landmarks themselves have some active components, like radio beacons. When a landmark is detected, the position of the robot is determined using the relative distance between the robot and the currently visible landmarks and the absolute positions of these landmarks. For the latter, a map is needed, which might also be created by the robot in the case of an unknown environment, leading to the simultaneous localization and mapping (SLAM) problem [9].

During the mapping phase, the robot detects the landmarks for the first time, but since its own position is yet uncertain due to the lack of absolute position updates, the position of the landmarks cannot be determined with the required accuracy at the time of the detection. However, as the robot moves, and detects a previously detected landmark by closing a loop, all the positions of the landmarks detected between the two observations can be improved retrospectively. This is something that the online methods EKF and PF cannot achieve: taking new measurements to update old estimates.

In this paper, we are presenting an easily implementable offline method to reconstruct the path by the fusion of MEMS gyroscope, odometer and GPS sensors. The method considers all measurements at all points of the path. It is based on the estimation of the gyroscope bias that is estimated by minimizing the distance between the reconstructed path from the gyroscope and odometer measurements, and the GPS path, using simple optimization methods.

In the next section, we will present our method. We will show how the method produces a path with much higher spatial and temporal resolution and also with higher accuracy than the GPS. We will also show test results on real-world measurement data, using it on a bicycle-mounted measurement system.

II. THE METHOD IN DETAIL

A. Path reconstruction

The greatest limitations of MEMS gyroscopes are their bias and drift [10]. This prohibits the use of these sensors for scenarios, where the sensor signal must be integrated over a long time, since one cannot fully compensate the changing bias, and the resulting error accumulates over time.

However, if one could estimate the bias over the course of the measurement, this limitation could be canceled [11]. One possible approach is to reconstruct the trajectory only from the IMU data, and compare it to a simultaneously measured trajectory using some external reference (e.g. GPS). The bias then can be determined by changing it with optimization algorithms to minimize the difference between the two.

To reconstruct the physical trajectory of the sensor at the time of the measurement, we must calculate two variables from the available sensory data. These variables are the position and orientation of the sensor, at each sample. To do this, we process the 3D gyroscope measurements along with a filtered version of the odometer measurements, which are basically the traveled distance.

The position of the sensor $\boldsymbol{p}_t$ is a three-dimensional vector at each sample, taken in a Cartesian coordinate system, aligned to the east, north and vertical up geographical axes. The orientation of the sensor $\boldsymbol{q}_t$ is represented with a unit quaternion, where the imaginary units coincide with the axes of the frame of reference of the position vector. Quaternions are extensions of complex numbers and are ideal for representing rotations in 3D space.

Reconstruction of the path from gyroscope samples and traveled distance samples is almost straightforward. The initial position $\boldsymbol{p}_0$ is always placed to the origin of the frame of reference, but the initial orientation $\boldsymbol{q}_0$ is a parameter of the method, which is determined using an optimization step, which will be covered later in the II./C. section. After the initial setup, at each sample the position and orientation are updated in chronological order, according to the following equations:

$$\boldsymbol{p}_t = \boldsymbol{p}_{t-1} + \boldsymbol{dir}_t d_t; \quad \boldsymbol{dir}_t = \boldsymbol{q}_t^* \boldsymbol{h} \boldsymbol{q}_t \qquad (1)$$

$$\boldsymbol{q}_t = \boldsymbol{q}_{t-1} \boldsymbol{g}_t \qquad (2)$$

, where $\boldsymbol{h}$ is the heading vector in the frame of reference of the sensor, d_t is a scalar value denoting the traveled distance in this time step. So (1) is basically rotating the traveled distance vector into the world frame of reference from the sensor frame of reference. In (2), $\boldsymbol{g}_t$ is the quaternion created from the gyroscope sensor readings:

$$\boldsymbol{g}_t = 1 + \frac{\varphi\theta\omega}{8} + i\left(\frac{\varphi}{2} - \frac{\theta\omega}{4}\right) + j\left(\frac{\theta}{2} + \frac{\varphi\omega}{4}\right) + k\left(\frac{\omega}{2} - \frac{\theta\varphi}{4}\right) \qquad (3)$$

, where φ, θ, ω are the rotation angles around the x, y and z axes of the frame of reference, calculated from the bias compensated gyroscope readings. The imaginary units for the axes are i, j, and k, respectively. Note that this equation is an approximate one. In the accurate version trigonometric functions are used on these angles. But since the value of these angles are so small even at full-scale output that we can approximate $\sin(x) \approx x$ and $\cos(x) \approx 1$ with very low error. This approximation, on the other hand, gives us considerable performance gain.

B. Noise model

The equations in II./A. would be sufficient to estimate the path from the gyroscope and odometer measurements, if they were accurate without any error or noise. Unfortunately, that is not the case.

Both the gyroscope and the odometer are sources of various errors. Since both are digital sensors, they have quantization error. When considering the odometer, the traveled distance (which is always positive) is integrated during the calculations, causing the relative quantization error of the integrated traveled distance converging to zero.

The quantization error of the gyroscope is also ignored because the uncorrelated Gaussian thermal noise on the output is far greater than one LSB, effectively dithering the output.

This noise is actually very useful to measure less than one LSB angular velocities with the gyroscope.

The odometer has a time-invariant gain error, D_{gain}, so the measured d_t value is the real odometer value $\widehat{d_t}$, multiplied by the gain error (4). To determine this error, we use an optimization step, which will be detailed in the II./C. section.

$$d_t = \widehat{d_t} D_{gain} \tag{4}$$

We assume that the gyroscope is calibrated, so there is no considerable gain error on its output. On the other hand, MEMS gyroscopes in general suffer from bias (offset error) $\boldsymbol{B}_t$, which is also time-variant, and also a Gaussian noise component (thermal noise) N, which has already been mentioned (5). This Gaussian noise can be mitigated by averaging, which is implicitly done by the integrations during the calculation in (1) and (2).

$$[\varphi_t, \theta_t, \omega_t] = [\widehat{\varphi_t}, \widehat{\theta_t}, \widehat{\omega_t}] + \boldsymbol{B}_t + N \tag{5}$$

$$\boldsymbol{B}_t = [b_{\varphi,t}, b_{\theta,t}, b_{\omega,t}] \tag{6}$$

The biggest source of errors is the gyroscope bias. This can be mitigated by measuring the bias, and subtracting it from every sample. However, the drift of the bias makes this solution imperfect.

C. Gyroscope bias estimation

The aim of the method is to estimate the gyroscope bias for each measurement sample by minimizing an objective function:

$$C = \sum_{t \in T_{GPS}} (\boldsymbol{p}_t - \boldsymbol{GPS}_t)^2 \tag{7}$$

, which is basically the deviation of the GPS path and the path deducted from the gyroscope samples. The minimization is done in two steps, with different algorithms.

The first step is needed to set the gyroscope bias to a reasonable value, and it can be fine-tuned in the second step. The bias $\boldsymbol{B}_t$ is considered constant, which makes it possible to solve the problem quickly with fast Quasi-Newton solvers, as the number of tunable parameters is low. Here not only the gyroscope bias, but the initial orientation $\boldsymbol{q}_0$ and the odometer gain D_{gain} are also considered as tunable parameters. Optimizing these parameters with respect to the objective function, we get estimates for these values, including the mean gyroscope bias values for all three axes. These will be tuned further in the second step.

In the second optimization step, the bias is no longer considered constant throughout the whole measurement data series, but it is divided into equally long time segments of 5 seconds, where the bias is considered constant. The reason for this partition is twofold, (*) this way we have only fifth of tunable parameters than GPS positions (provided that we have 1 sample/sec GPS data). and (**) this also reflects the short-time stability of the gyroscope bias. We don't want to model the uncorrelated Gaussian noise (N in (5)) on the gyroscope output as offset error.

One can consider different optimization algorithms for the second step. We have used gradient methods, where the parameter vector is modified with the gradient of the objective function:

$$\boldsymbol{B}_{t,i+1} = \boldsymbol{B}_{t,i} + \boldsymbol{\delta} \tag{8}$$

$$\boldsymbol{\delta} = \gamma \frac{\delta C}{\delta \boldsymbol{B}} \tag{9}$$

We have also tested quasi-Newton methods, namely the Levenberg-Marquardt (LM) method, which modifies the parameters by solving a linear equation system for δ created from the Jacobian matrix of the objective function:

$$(\boldsymbol{J}^T \boldsymbol{J} + \lambda \boldsymbol{I})\boldsymbol{\delta} = \boldsymbol{J}^T \boldsymbol{e} \tag{10}$$

$$\boldsymbol{e}_t = \boldsymbol{p}_t - \boldsymbol{GPS}_t;\ \boldsymbol{J}_t = \frac{\delta \boldsymbol{p}_t}{\delta \boldsymbol{B}} \tag{11}$$

D. Efficient analytic derivative calculation

Both methods need the partial derivatives of the objective function with respect to the tunable parameters (gyroscope bias). These derivatives are calculated as follows:

$$\frac{\delta C}{\delta \boldsymbol{B}_\tau} = \sum_{t \in T_{GPS}} 2\boldsymbol{e}_t \frac{\delta \boldsymbol{p}_t}{\delta \boldsymbol{B}_\tau} \tag{12}$$

$$\frac{\delta \boldsymbol{p}_t}{\delta \boldsymbol{B}_\tau} = \frac{\delta \boldsymbol{p}_{t-1}}{\delta \boldsymbol{B}_\tau} + \frac{\delta \boldsymbol{dir}_t}{\delta \boldsymbol{q}_t} \frac{\delta \boldsymbol{q}_t}{\delta \boldsymbol{B}_\tau} d_t \tag{13}$$

$$\frac{\delta \boldsymbol{q}_t}{\delta \boldsymbol{B}_\tau} = \begin{cases} \frac{\delta \boldsymbol{q}_t}{\delta \boldsymbol{q}_{t-1}} \frac{\delta \boldsymbol{q}_{t-1}}{\delta \boldsymbol{B}_\tau} + \frac{\delta \boldsymbol{q}_t}{\delta \boldsymbol{g}_t} \frac{\delta \boldsymbol{g}_t}{\delta \boldsymbol{B}_\tau}, & \text{if } t \geq \tau \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The equation (12) needs to be solved for all τ indices, which cover all gyroscope samples, but at least each different 5 seconds long gyroscope bias time segment. Since the number of GPS samples (which determines the length of the summation) and the number of gyroscope bias time segments are proportional to the length of the measurement, the naive implementation of (12) has at least quadratic complexity.

However, the above equations can be rewritten in closed form, using some intermediary variables $\boldsymbol{Q}$, $\boldsymbol{A}$ and $\boldsymbol{D}$ as follows:

$$\boldsymbol{Q}_t = \prod_{i=t}^{0} \frac{\delta \boldsymbol{q}_i}{\delta \boldsymbol{q}_{i-1}} \tag{15}$$

$$\boldsymbol{A}_t = \frac{\delta \boldsymbol{q}_t}{\delta \boldsymbol{g}_t} \frac{\delta \boldsymbol{g}_t}{\delta \boldsymbol{B}_\tau} \tag{16}$$

$$\boldsymbol{D}_t = \sum_{i=0}^{t} \frac{\delta \boldsymbol{dir}_i}{\delta \boldsymbol{q}_i} \boldsymbol{Q}_i\, d_i \tag{17}$$

$$\frac{\delta \boldsymbol{q}_t}{\delta \boldsymbol{B}_\tau} = \boldsymbol{Q}_t \boldsymbol{Q}_\tau^{-1} \boldsymbol{A}_\tau \tag{18}$$

$$\frac{\delta p_t}{\delta B_\tau} = (\boldsymbol{D}_t - \boldsymbol{D}_\tau)\boldsymbol{Q}_\tau^{-1}\boldsymbol{A}_\tau \qquad (19)$$

$$\frac{\delta C}{\delta B_\tau} = \left[\left(\sum_{t\in T_{GPS}} 2\boldsymbol{e}_t\boldsymbol{D}_t\right) - \left(\sum_{t\in T_{GPS}} 2\boldsymbol{e}_t\right)\boldsymbol{D}_\tau\right]\boldsymbol{Q}_\tau^{-1}\boldsymbol{A}_\tau \qquad (20)$$

The computation complexity of the above calculation method is linear in the length of the measurement data, which makes the whole method very efficient, as these equations must be calculated at each optimization iteration step.

The main innovation in this computation method is that (13) and (14) is decomposed to a sum and a product of the same function $\boldsymbol{D}$ and $\boldsymbol{Q}$ at t and τ, in (19) and (18), respectively. It should be noted, that the matrix inverse $\boldsymbol{Q}_\tau^{-1}$ in (18) always exists, since $\boldsymbol{Q}_t$ is a product of non-singular Jacobian matrices of unit quaternion multiplication.

III. The Data Collection System

In order to get real life data, we have investigated several options. We have chosen a bicycle as our main testing platform, for several reasons. One of the main reasons (other than availability) was that it is relatively easy to get odometer sensor data on a bicycle, compared to cars or motorbikes. Bicycle computers, to get odometry data, use a magnetic switch mounted on the fork, and a magnet mounted on the spoke of the front wheel. This measures each full revolution of the wheel. Sampling this sensor would give 1 if the magnetic switch is on, 0 otherwise. With 100 Hz sample rate, this would result in a very sparse non-zero signal. Multiplying it by the perimeter of the wheel and using moving average on it will produce the travelled distance.

Another main reason for choosing a bicycle was that when riding it, the bicycle rotates around all three axes and not just the axis perpendicular to the ground surface, as it bends to the side during cornering. This way we could test our method on all three axes simultaneously, as the rotations around the axes affect each other.

We have mounted a custom built sensor logging device on the bicycle as shown in Fig. 1. It is built around an ATMEGA328p microcontroller. It has the following sensors: L3G4200D 3-axis MEMS gyroscope, BMP085 pressure sensor, HMC5883L 3-axis compass, ADXL345 3-axis accelerometer, MPU6050 3-axis gyroscope and accelerometer, and Quectel L80 GPS module. All sensors including the odometer are sampled at 100 Hz, even those that has lower maximum sampling rate, except the GPS, which is sampled at 1 Hz.

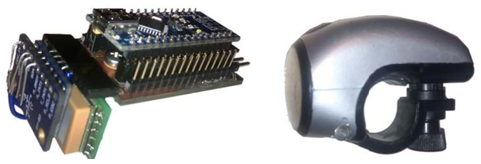

Fig. 1. The data collection system, built in the case of a bicycle lamp

For our tests, we have used only the two gyroscopes and the GPS module. Since the altitude reading of the GPS was very noisy, we have used the pressure sensor instead, to determine the altitude of the device. The output of the sensors was logged to a micro SD memory card.

IV. Test Results

We have used the sensor data collection system on a path around Budapest. The path includes some sharp turns, long straight and mildly turning segments, a bridge with some elevation change, and some places with blocked GPS signal. The path took about half an hour.

After the sensor data was collected, it was processed on a desktop computer. The results are shown on the following figures.

Fig. 2. shows the path after the first step of our method, which is optimizing the objective function for constant bias values. As it can be seen, it does not result in a good absolute path, as the bias changes over time, but the local features of the path are all visible with high temporal resolution.

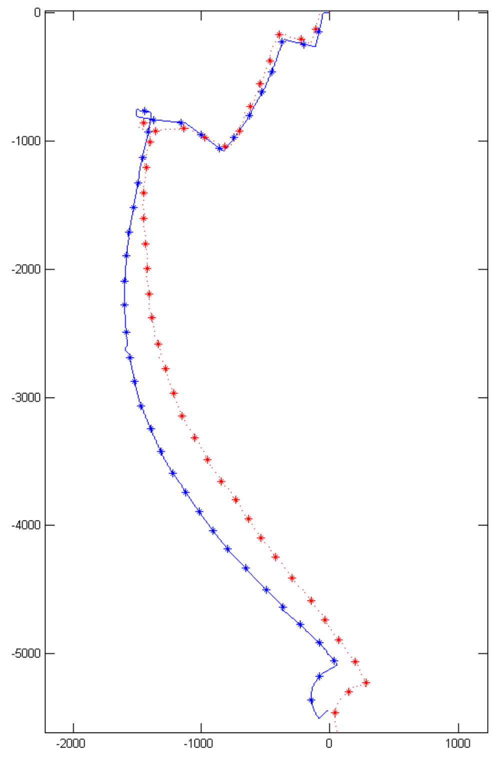

Fig. 2. Reconstructed path after the first optimization step. Red (dotted): GPS path, blue (solid): reconstructed path. Scale is in meters.

Fig. 3. shows the path after the second optimization step, when the bias is considered as a changing variable over time. The optimization was done using the gradient method.

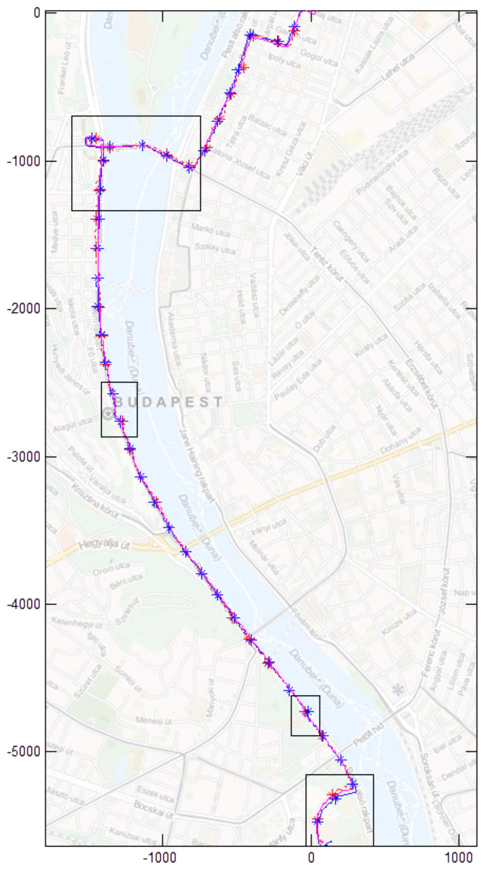

Fig. 3. Reconstructed path after the second optimization step. Red (dotted): GPS path, blue (solid): reconstructed path, magenta: ground truth. Scale is in meters.

As it can be seen, the deviation between the calculated path and the GPS path is quite small, it is 8.09 meters in average, which is around the accuracy of the GPS. It should be noted that the GPS also has a certain inaccuracy, which should not be followed with the calculated gyroscope path, as it wasn't the true motion trajectory.

What is more important, the deviation does not increase over time, which means that an arbitrary long measurement could be carried out with this method. The absolute position measurements of the GPS module makes it possible to determine the gyroscope bias all along the measurement samples.

In Fig. 4. some features of the path can be seen magnified from Fig. 3. At all of these figures, the true trajectories of the bicycle look much more like the reconstructed path segments than the GPS path. On Fig. 4.b. and d. the GPS signal is blocked by a tunnel and heavy foliage, making it even worse quality than at other parts of the path. However, the reconstructed path looks just like everywhere else.

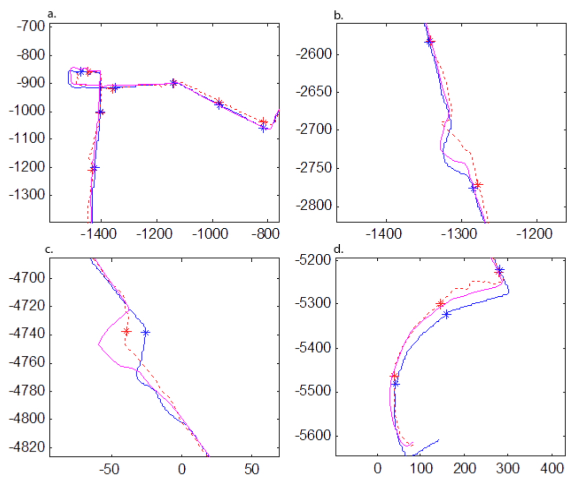

Fig. 4. Segments of the reconstructed path. Red (dotted): GPS path, blue (solid): reconstructed path, magenta: ground truth. Scale is in meters.

The average angular velocity (cca. 30 degree in 15 minutes) at the curvature of the Danube at the average speed of the bicycle at the middle of the path is less than 1 LSB (0.00875 °/s), which is also much less than the bias drift during this time interval. Yet our method was able to precisely reconstruct that section of the path. Measuring less than 1 LSB angular velocity is only possible due to the Gaussian noise on the gyroscope output, which dithers the analog signal before A/D conversion. A built-in high-pass filter in the sensor (which is a common feature of modern MEMS gyroscopes), if used, would make such a measurement impossible.

A. Comparison of the minimization methods

The main benefit of using the gradient method is that one iteration step can be calculated very fast, since only the gradient must be returned, which can be computed as (20). The result of the gradient method is also smooth, which makes it an ideal optimizer for estimating the samples of a smooth function, here namely the gyroscope bias, see Fig. 5.

The main drawback of the gradient method is that many iterations might be needed to converge to a local minimum. The convergence might be very slow if the parameter vector is on a plateau of the objective function surface, where the gradient is close to zero.

LM optimizer and Quasi-Newton optimizers in general are almost the exact opposites of the simple gradient methods. They can converge to a local minima in only a few iterations, but these iteration steps could take much more time than in the simple gradient methods. Partly because the Jacobian matrix of the objective function must be returned, which takes more time to compute than the gradient, and mainly because solving a linear system takes place in each step. Usually, this linear system is near singular, which causes the solution that is added to the parameter vector to have a high variance with many outliers, see Fig. 5.

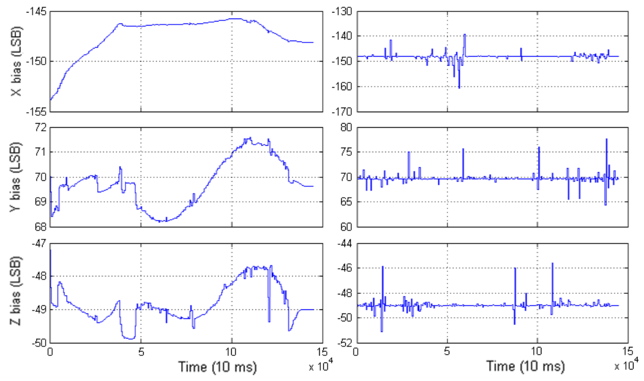

Fig. 5. Gyroscope bias after optimization, gradient method (left) and LM method (right)

We only had full success with the gradient method. The LM optimizer also started to reduce the error, but after some iterations the error began to increase, and the optimization became divergent. It might be worth it to investigate a combination of the two optimization methods to achieve increased performance.

B. *Comparison with ground truth and alternative methods*

We have compared the reconstructed path with the ground truth path. Unfortunately, there was no other measurements available of the true path of the bicycle than the measurements that we have recorded, so we had to create it by carefully drawing the most probable path on a satellite image. This might not be as precise as one would want, but considering the resolution of the available satellite image, it is quite certain that the average accuracy of the ground truth path obtained this way is less than 3 meters. There might be a constant bias between the satellite image coordinates and the measured GPS points, so we compensated this by translating the ground truth path so that the distance between the path and the GPS points would be minimal.

After we determined the ground truth, we calculated the closest distance of each point in the GPS data and each point in the reconstructed path from the ground truth polyline, without considering temporal information.

The average distance between the GPS points and the ground truth was 6.97 meters. The average distance between the points of the reconstructed path and the ground truth was 6.62 meters. This means that our method was able to slightly improve the accuracy of the GPS points.

The authors in [11] proposed two methods with KF and PF in a very similar application. They reported 7 to 25 meters accuracy in their measurements, depending on different circumstances.

V. Conclusion

In this paper, we have presented a method which is able to reconstruct the path, based on gyroscope and odometer measurements, by estimating the gyroscope bias using some external reference measurements, GPS in this case. The estimation of the bias is done by minimizing the difference between the reference path and the reconstructed path by tuning the bias values. An efficient calculation method was presented for computing the derivatives of the objective function, which is needed for the optimization.

The resulting path has higher resolution both in time and space than the GPS path in itself. It is also more accurate, as the gyroscope is able to sense very little changes in the heading, where the GPS cannot. It has long-term accuracy opposed to purely inertial IMU systems, where the error accumulates over time. It also might be possible that this method increases the accuracy of consecutive GPS measurements just as averaging would do, as it provides a connection between two GPS measurements, although this subject needs more investigation.

This method makes it possible to provide accurate position information in places for shorter time periods where GPS signal is blocked, as the user is passing through these areas. By using another kind of external reference measurements than GPS, like range finders on mobile robots, or even a-priori information, like a roadmap, new applications become possible. Such applications could be e.g. indoor localization for mobile robots or automotive navigation without the need of GPS satellites.

References

[1] Tailanian, M.; Paternain, S.; Rosa, R.; Canetti, R., "Design and implementation of sensor data fusion for an autonomous quadrotor," (I2MTC) Proceedings, 2014 IEEE International , vol., no., pp.1431,1436, 12-15 May 2014

[2] Templeton, Todd, et al. "Autonomous vision-based landing and terrain mapping using an MPC-controlled unmanned rotorcraft." Robotics and Automation, 2007 IEEE International Conference on. IEEE, 2007.

[3] Ning Yang; Wei Feng Tian; Zhi Hua Jin; Chuan Bin Zhang, “Particle filter for sensor fusion in a land vehicle navigation system” Measurement Science and Technology, Volume 13, Number 3, January 2005

[4] Zhao, L., Ochieng, W. Y., Quddus, M. A., & Noland, R. B. (2003). An extended Kalman filter algorithm for integrating GPS and low cost dead reckoning system data for vehicle performance and emissions monitoring. The Journal of Navigation, 56(02), 257-275.

[5] Francois Caron, Emmanuel Duflos, Denis Pomorski, Philippe Vanheeghe, GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects, Information Fusion, Volume 7, Issue 2, June 2006, Pages 221-230, ISSN 1566-2535

[6] Grzonka, Slawomir, Giorgio Grisetti, and Wolfram Burgard. "Towards a navigation system for autonomous indoor flying." Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. IEEE, 2009.

[7] Hoflinger, F.; Rui Zhang; Reindl, L.M., "Indoor-localization system using a Micro-Inertial Measurement Unit (IMU)," European Frequency and Time Forum (EFTF), 2012 , vol., no., pp.443,447, 23-27 April 2012

[8] Kuznietsov, A., "Inertial measurement system for performance evaluation of track and field sprinters," (I2MTC), 2012 IEEE International , vol., no., pp.1681,1686, 13-16 May 2012

[9] Thrun, S., Burgard, W., & Fox, D. (2005).Probabilistic robotics. MIT press, 2005.

[10] Woodman, O. J. (2007). An introduction to inertial navigation. University of Cambridge, Computer Laboratory, Tech. Rep. UCAMCL-TR-696, 14, 15.

[11] Georgy, J., Noureldin, A., Korenberg, M. J., & Bayoumi, M. M. (2010). Modeling the stochastic drift of a MEMS-based gyroscope in gyro/odometer/GPS integrated navigation. Intelligent Transportation Systems, IEEE Transactions on, 11(4), 856-872.