Application of the MSCKF algorithm on the Cheddar Gorge Wildcat Dataset

Anastasios I. Mourikis and Tue-Cuong Dong-Si
Dept. of Electrical Engineering, University of California, Riverside
Corresponding author e-mail: mourikis@ee.ucr.edu

November 30, 2010

1 Introduction

This report describes the results of applying the Multi-State-Constraint Kalman Filter (MSCKF) algorithm for estimating the trajectory of the vehicle in the dataset collected by the Wildcat platform at the Cheddar Gorge, Somerset area. The derivation and mathematical details of the MSCKF algorithm are described extensively in previous publications [1–3], and therefore this report focuses on the implementation details and modifications to the basic algorithm that were required for processing the Cheddar Gorge dataset. In the following sections, we first describe the steps taken for data pre-processing, the selection of parameters for the MSCKF, filter initialization, and the details of the feature extraction and matching algorithms. Subsequently, we present the estimation results and time requirements of the MSCKF. Finally, the report discusses the lessons learned, and presents suggestions for collecting future datasets.

2 Data preprocessing

The dataset provided for this project comprised the measurements of the Xsens MTi-G-28 inertial measurement unit, the stereoscopic images recorded by a PointGrey Bumblebee 2 camera rig, and the output of an OXTS RT3044 inertial navigation unit, used for ground truth. In the following, we describe the preprocessing steps applied on each sensor's data.

2.1 Xsens data

The Xsens output includes the measurements of several sensors (gyroscope, accelerometer, GPS, magnetometer, air pressure, and temperature). For the purposes of this project, only the measurements of the gyroscope and accelerometer were used, since the goal of the work is to evaluate the performance of vision-aided inertial navigation in the Cheddar Gorge experiment. The Xsens output was recorded in the "raw" format, which directly provides the readings of the sensor's analog-to-digital converters. This output is uncalibrated, which means that the effects of sensor misalignment, biases, temperature-related variations in the biases and scale factor, and g-sensitivity are not compensated for.

Prior to using the IMU measurements in the MSCKF, we use the available calibration information to compensate for the misalignment between the IMU axes, as well as for constant bias and scale factors. Unfortunately, the remaining effects (e.g., temperature dependence of the biases and scale factors) are impossible to compensate for offline. Instead, the additional uncertainty due to these effects is modelled in the noise parameters used in the MSCKF (see Section 4).

2.2 Bumblebee data

The stereoscopic images are captured in "raw" (i.e., unrectified) form, at a rate of 20Hz. Prior to processing the images in the filter, we use the Bumblebee calibration parameters to produce grayscale rectified images at a resolution of 640×480 pixels. The original MSCKF algorithm works using monocular image sequences, and therefore for the results presented in Section 7 we only process the images of one camera.

2.3 Sensor synchronization

The dataset provides timestamps for each of the IMU measurements and each of the Bumblebee image pairs. To evaluate the precision of these timestamps, we inspected the difference between the timestamps of consecutive readings from each sensor. For the Bumblebee, the time difference between consecutive timestamps is remarkably stable, and equal to 50ms, which corresponds to the nominal 20Hz frame rate. This leads us to believe that the image timestamps are reliable.

On the other hand, a similar inspection for the IMU timestamps reveals a different situation. Specifically, the measurements appear to be recorded in "batches", with periods of where no measurement is recorded, followed by a burst of a few dozen measurements with almost identical timestamps. This is demonstrated in Fig. 1. Inspection of the IMU packet counters shows that no IMU packets are lost. This indicates that since the timestamps are recorded using the computer's operating-system (OS) time, the "bursty" appearance of the timestamps is due to the way in which the OS schedules the different tasks, and uses the computer's I/O buffers.

Since no IMU packets are lost, the timestamps of the IMU can be easily corrected, using the knowledge that the IMU measurements occur at a rate of 100Hz. Prior to using the Xsens data in the filter, we create new timestamps for all the measurements, such that the time difference between consecutive measurements is constant.

In addition to correcting the time difference between consecutive IMU measurements, it is also necessary to account for the delay that exists between the time a sensor records a measurement, and the time this measurement is assigned a timestamp from the OS. Since this delay may generally be different for each sensor, this can result in an offset between the timestamps of the different sensors (in our case, the Xsens and the Bumblebee), which needs to be corrected. To compute this time offset (assuming it remains constant throughout the dataset), we have computed separate rotation estimates using the Xsens gyro measurements only, and using the Bumbebee images only. For the Xsesns, this is done by simple integration of the rotational velocity, while for the Bumbebee, the rotation is computed using least-squares image-based motion estimation between consecutive frames. By alignging the rotation estimates in time, we have estimated the time offset between the two sensor streams, and adjusted the IMU timestamps accordingly.

3 Image processing for feature extraction and matching

The images provided by the Bumblebee are processed to extract point features, which are subsequently used by the filter. In our current implementation, we are using Harris corner features [**?**,4]. First, the "cornerness" measure (i.e., the smallest eigenvalue of the averaged gradient outer product matrix) is computed in the entire image large. Those points that are local maxima of the cornerness measure, and whose value is above a preselected threshold, are chosen as corners. To ensure a good distribution of features in the image, we separate the image into a 16×16 grid, and we keep only the best two features in each of the grid cells. This allows for a maximum of 512 features (i.e., $16 \times 16 \times 2$) in each image.

To perform feature matching, we use guided matching and normalized cross-correlation. Specifically, given the features detected in two consecutive images, I_k and I_{k+1}, we first find those features that are *potential* matches by using the epipolar geometry. Given the estimate of the camera motion, which is

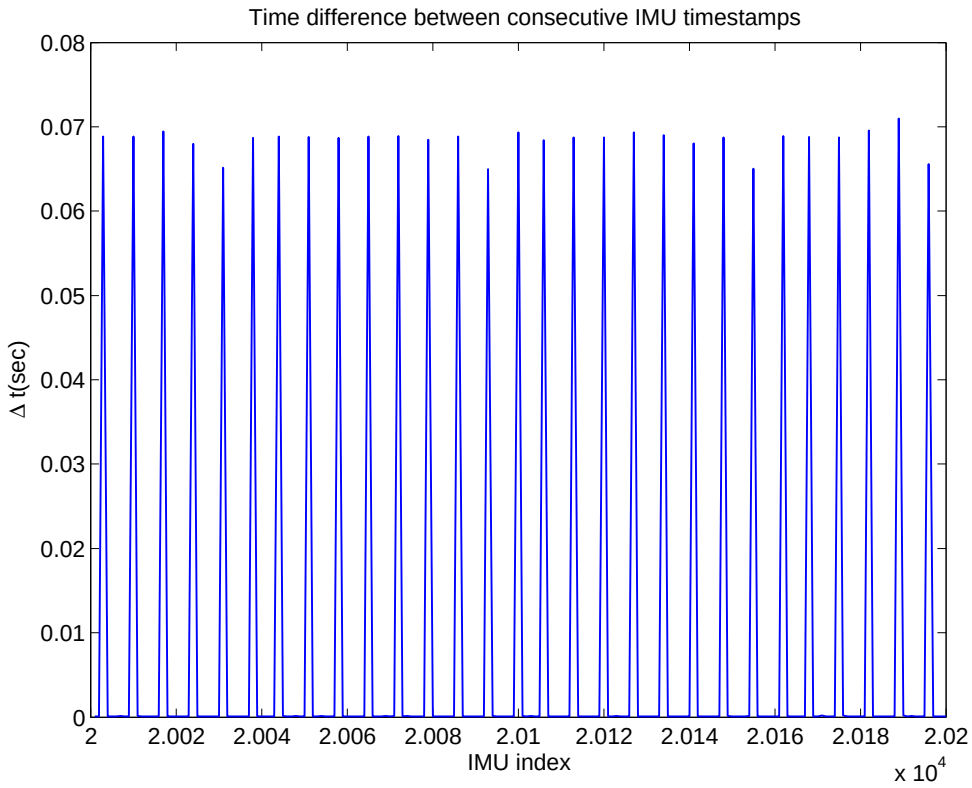

Figure 1: Difference between consecutive Xsens timestamps, illustrating the "bursty" nature of the timestamps.

available from the filter, we can compute for each feature in image I_k the corresponding epipolar line in image I_{k+1}. Only features close (within 5 pixels) of this epipolar line are considered potential matches. For those features, we define a 19×19 image patch centered at the feature locations, and compute the normalized cross-correlation. In this way, for each feature in image I_k we find the best-matching feature in image I_{k+1}. Subsequently, we repeat the same process in the reverse direction, finding the best match in image I_k for each feature in image I_{k+1}. To remove (most) erroneous matches, we then perform a mutual consistency check. Specifically, we declare a feature f_i in image I_k to be matching a feature f_j in I_{k+1} only if (i) out of all the potential matches in image I_{k+1}, f_j has the best normalized cross-correlation score to f_i, (ii) out of all the potential matches in image I_k, f_i has the best normalized cross-correlation score to f_j, (iii) the normalized cross-correlation score is above 0.8. An example image with the detected features superimposed is shown in Fig. 2.

The use of guided matching in the above process (i.e., the use of the epipolar constraint to identify potential matches) results in an algorithm with low computational cost, while the use of the mutual consistency check ensures that very few outliers survive the matching process and are passed on to the filter. To ensure that these outliers are not processed by the filter, we further employ a Mahalanobis gating check in the MSCKF, prior to using a feature's measurements. Moreover, features whose disparity is very small (close the image noise level) are discarded. Overall, in the Cheddar Gorge dataset, 262 features are detected on average in each image, and 137 features are successfully matched. Each feature is tracked for an average of 4.9 frames. Out of the successfully matched features, approximately two thirds pass the disparity and Mahalanobis tests and are processed for MSCKF updates.

Figure 2: An example image from the dataset, with detected features superimposed. Green features are those that are successfully matched, red are those that were not matched.

4 Algorithm parameter selection

The MSCKF is a Kalman-filter based estimator, and therefore it requires prior knowledge of the covariance matrix of the noise in the sensor measurements. Specifically, it requires knowledge of the covariance of the noise in the image feature measurements, the variance of the noise in the IMU rate and acceleration measurements, as well as the variance of the random walk processes that describe the IMU biases. These are discussed in what follows.

4.1 Image noise

Regarding the image noise, we model the compounded effects of the image noise and quantization (we are not using subpixel feature detection) as isotropic, zero-mean, white Gaussian noise with standard deviation equal to one pixel along each direction.

4.2 IMU rate and acceleration measurements

The Xsens manual provides values for the standard deviation of the IMU rate and acceleration measurements. These values are 0.05 deg/sec/$\sqrt{\text{Hz}}$ for the gyroscopes, and 0.002 m/sec^2/$\sqrt{\text{Hz}}$ for the accelerometers. However, we have found these values not to be appropriate for processing the data recorded in this particular dataset. First, these values correspond to calibrated output, which is not available here, and it is unclear if the noise characteristics of the raw data are the same. Secondly, the sensor appears to be rigidly mounted on a metal surface on the vehicle, which means that the vibrations caused by the engine rotation (and their harmonics) directly impact the sensor. A frequency analysis of the IMU data reveals a very strong peak at 30 Hz, which is probably due to the engine (this corresponds to 1800RPM). It should be noted that

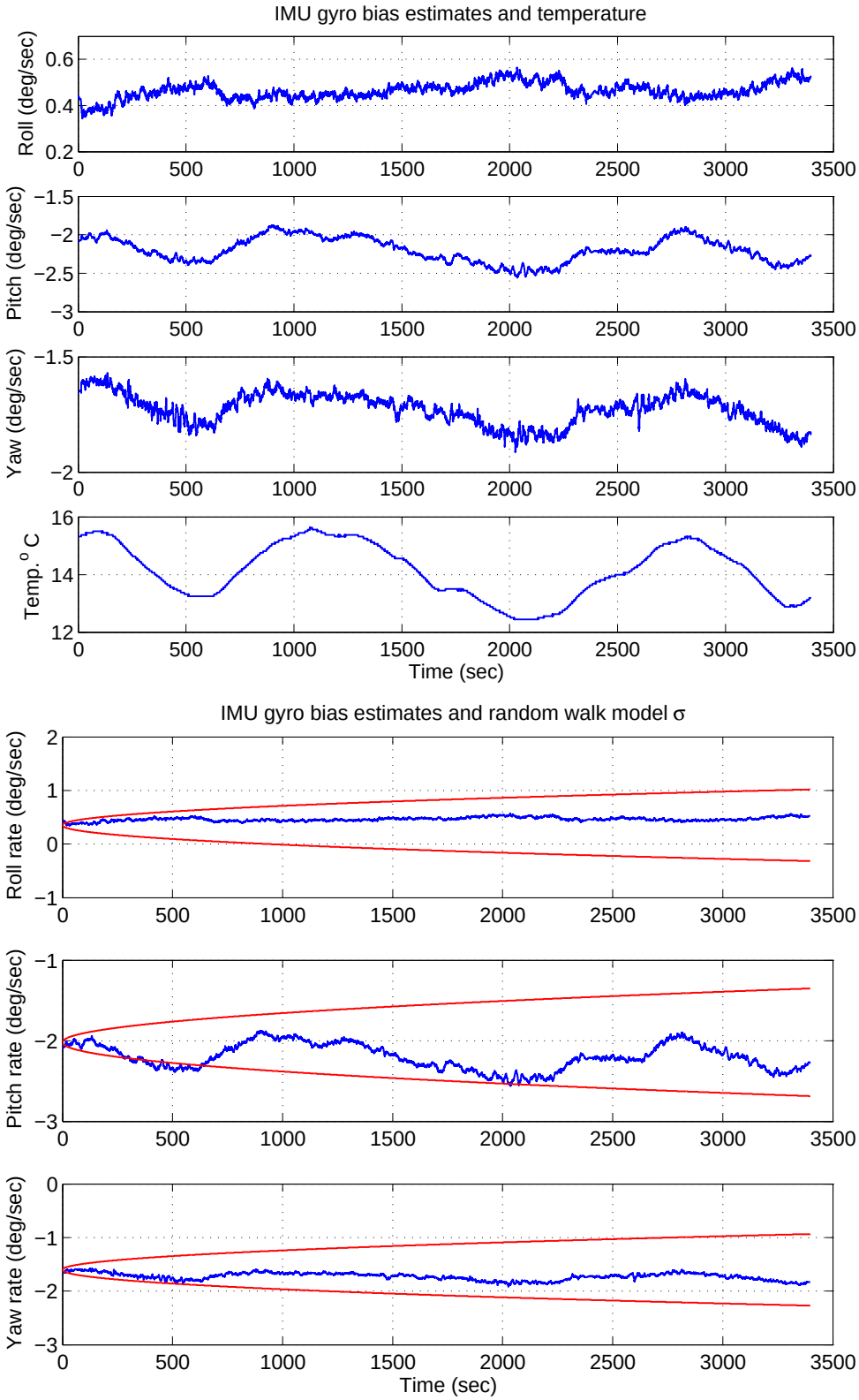

Figure 3: Top: Estimated gyroscope biases and IMU temperature for the duration of the dataset. Bottom: Estimated gyroscope biases vs. 95% confidence region of the random walk model.

30Hz is the bandwidth of the accelerometers, and therefore these vibrations and their harmonics could cause the accelerometer measurements to have accuracy less the nominal one (this is discussed in Section 3.4.2 of the Xsens manual).

For the reasons discussed above, we have computed the standard deviations of the rate and acceleration measurements from the data itself. Specifically, at the start and at the end of the dataset the vehicle is stopped, and therefore during those time periods the accelerometers and gyroscopes record constant signals (bias and earth rotation for the gyroscope, bias and local-frame gravitational acceleration for the acceleromenter) with the addition of noise. We have used the time interval in the beginning of the dataset to compute the standard deviation of the noise for each sensor, and used these values in the MSCKF. It is worth noting that the value for the standard deviation of the noise in the acceleration measurements is approximately one order of magnitude larger than the value reported in the manual.

4.3 IMU bias random walk

Following standard practice, the IMU biases are modelled as Gaussian random walk processes. The change in the value of a random walk during a time interval Δt is a Gaussian random variable with variance $\sigma^2 \Delta t$, where σ^2 is the power of the white noise "driving" the random walk. For the MSCKF, we require an estimate for the power of this white noise, both for the gyroscope and for the accelerometer biases.

We should note that, as discussed earlier, the IMU data was recorded in raw form, and is not temperature-compensated. The most significant effect of this is that the IMU biases exhibit significant, temperature-dependent, variation in time. Specifically, in Fig. 3(Top) we plot the estimated values of the gyroscope biases over time, as computed by the MSCKF, as well as the temperature of the IMU housing, reported in the raw data. It becomes clear that a correlation is present, both between the biases and the temperature, as well as between the biases in each of the three gyro axes. Therefore, modelling each of the biases as an independent random walk process (which is standard practice, and the approach followed in the MSCKF) is not entirely appropriate. However, it is an acceptable approximation in the current instance, where a model for the temperature-dependence of the biases is not available.

In order to compute an appropriate value for the power of the white noise driving the accelerometer and gyroscope random walks, we plot the estimates of the filter against a $\pm 2\sigma\sqrt{\Delta t}$ envelope (the 95% confidence region of the random walk process), and seek an appropriate value of σ, such that the computed random walk processes "match" their confidence regions (see bottom Fig. 3). The values chosen in this manner are 5.7×10^{-3} deg/$\sqrt{\text{sec}^3}$ for the gyroscopes, and 10^{-4} m/$\sqrt{\text{sec}^5}$ for the accelerometers. We should point out that even though using the filter estimates to determine the noise parameters can, in general, cause the estimates to converge to incorrect values, in the particular application the IMU biases are observable given the camera measurements. We have experimentally verified that choosing different values for the standard deviation of the random walk has very little effect on the estimated values of the biases. Therefore, we are confident that the chosen values are appropriate.

5 MSCKF initialization

In order to initialize the MSCKF, estimates for the initial values of the IMU position, velocity, and orientation in ECEF, as well as for the biases are required. These are obtained as follows:

- **Position**: The first reading from the OXTS ground truth is used to obtain the initial position in the ECEF frame. Since the global position is not observable using the IMU and camera measurements alone, the filter would be unable to determine position without using an initial "bootstrapping" from ground truth.

- **Velocity**: The initial velocity of the IMU is set to zero in the ECEF frame.
- **Orientation**: The yaw of the IMU with respect to the local NED frame is not observable (gyrocompassing using the Xsens IMU is not expected to yield estimates of meaningful accuracy), and therefore the first reading from the OXTS is used to obtain the initial yaw. For the initial pitch and roll, an initial estimate is produced using the accelerometer measurements in the first few IMU timesteps. These estimates are subsequently transformed to the ECEF frame and passed to the filter.
- **Gyroscope biases**: Since at the start of the dataset the vehicle is not moving, the rotational velocity measurements during this period provide us with estimates of the initial values of the biases (after subtracting the earth rotation vector).
- **Accelerometer biases**: Without motion, the accelerometer biases cannot be estimated. Therefore, we provide the filter with an initial guess for the accelerometer biases, which is computed as an average of the accelerometer bias estimates throughout the trajectory. This is reasonable, since in a realistic setting, one would have a good guess for the biases from a previous run (and the biases are not expected to vary significantly when calibrated output is used).

The initial covariance for the position, velocity, and yaw are set to zero, while for the other variables large uncertainty values are used. To obtain better estimates, as well as to compute the correlations between the different estimates (e.g., the initial pitch and roll are heavily correlated with the initial accelerometer biases), we use the first few IMU timesteps where the vehicle is stationary to carry out zero-velocity updates. Specifically, by inspection of the IMU data we have determined that after an initial jolt, the vehicle remains motionless between $t_1 = 11$ sec and $t_2 = 20$ sec. This is the period during which zero-velocity updates take place, while the first 11 sec of the dataset are not used.

6 Estimating the camera-to-IMU transformation

The accuracy of the estimates produced by the MSCKF depends crucially on the accuracy with which the camera-to-IMU transformation (rotation and translation between the camera and IMU coordinate frames) is known. This transformation is not possible to compute with sufficient precision using the measurements of the sensors' housing, because the sensor frames are not necessarily aligned with their housing (e.g., the camera optical axis depends on the lens distortions, among other parameters). For this reason, we have modified the original MSCKF algorithm described in [1–3] to include the rotation quaternion and the translation vector between the camera and IMU in the filter state vector. During filter propagation, the camera-to-IMU transformation estimate does not change, and during each filter update, the estimate is corrected using the measurement information. The initial estimate of this transformation is obtained from [5]. We have experimentally found that including the camera-to-IMU transformation in the filter results in a significant improvement in the accuracy of the trajectory estimates.

7 Estimation results

This section describes the estimation results of the MSCKF on the Cheddar Gorge dataset, and the time requirements of the filter. We first present in detail the results of the MSCKF when only the images of the right camera are processed. Subsequently, we discuss the filter's results in different scenarios (e.g., with different numbers of features used, or using the left camera instead).

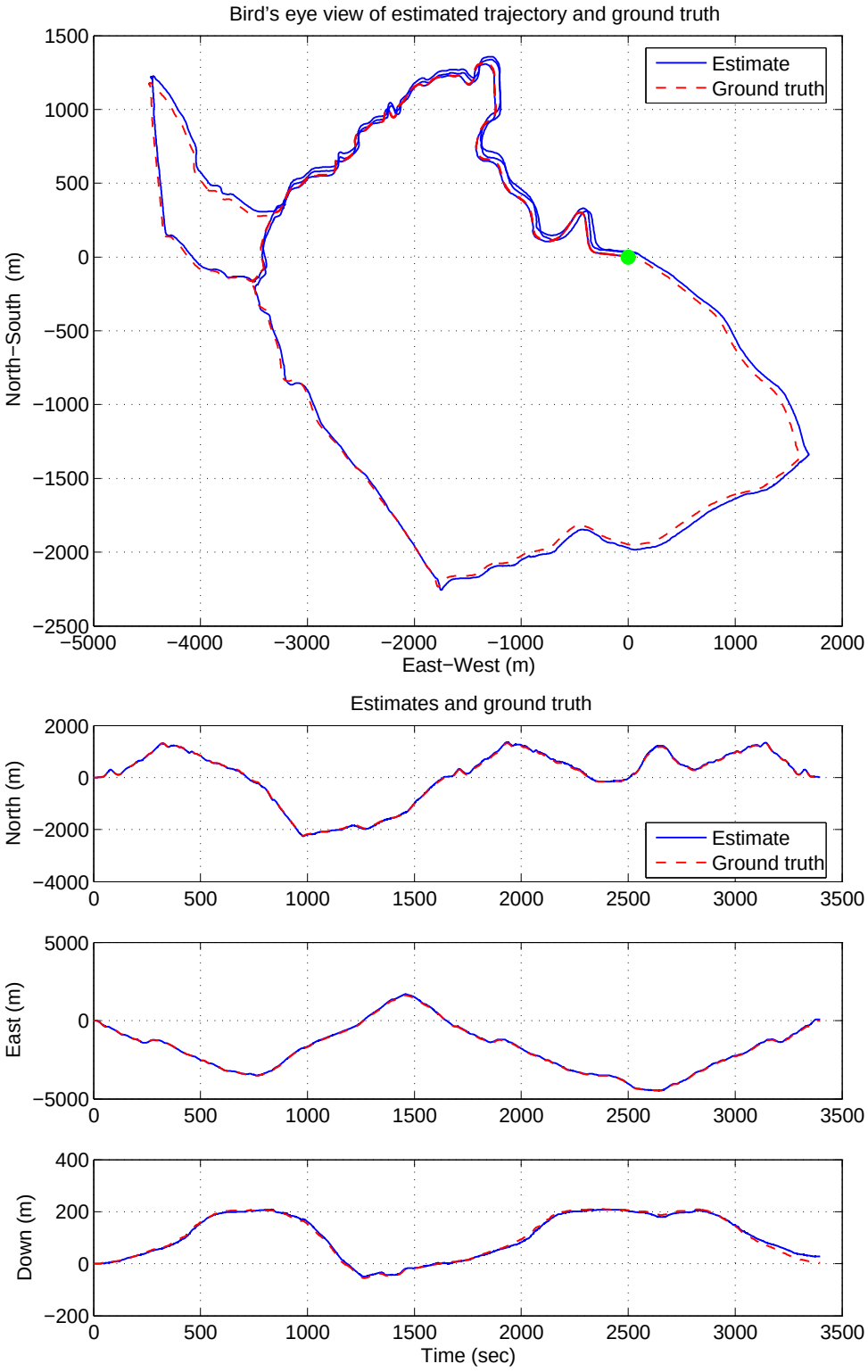

Figure 4: Top: Bird's eye view of estimated trajectory (solid blue line) vs. ground truth (red dashed line). The starting position is denoted by a green dot. Bottom: North-East-Down coordinates of estimated trajectory vs. ground truth.

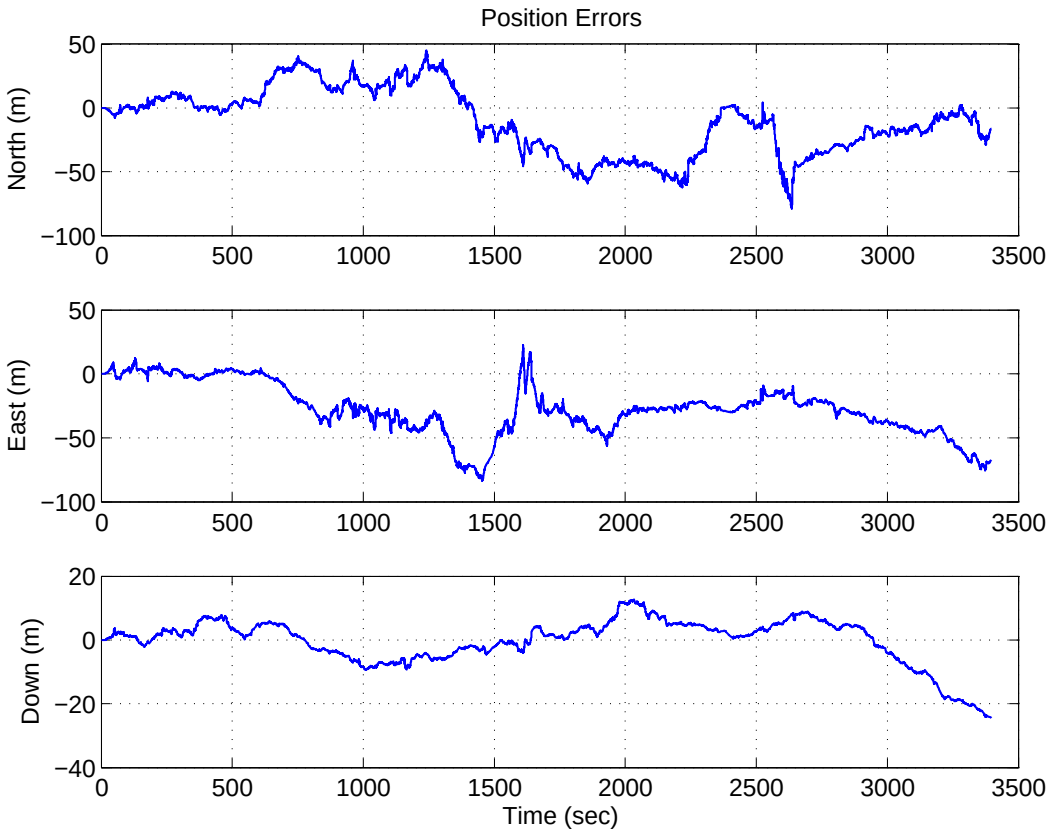

Figure 5: Position error for the MSCKF estimates. The three subplots correspond to the North, East, and Down coordinates.

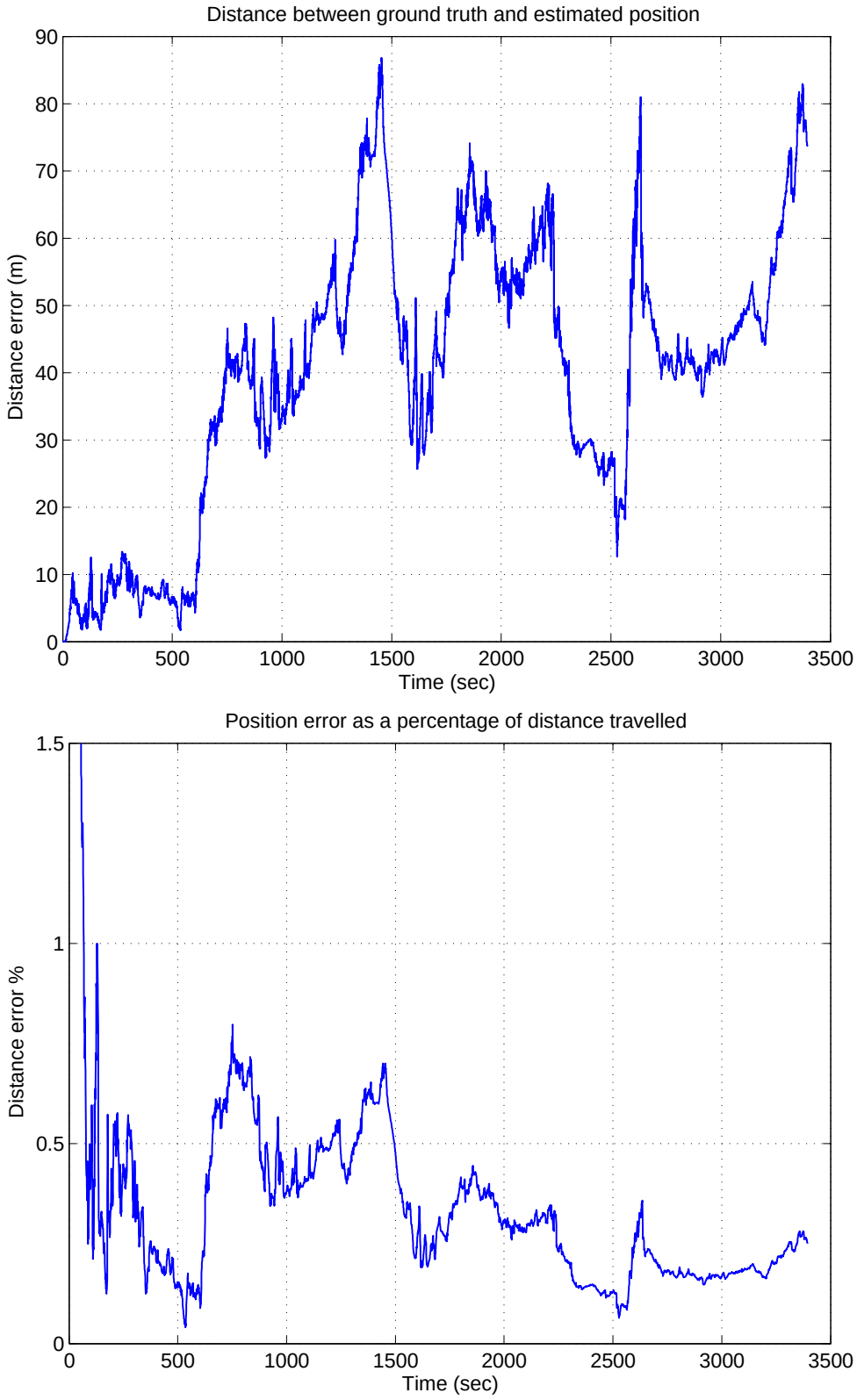

Figure 6: Distance between the true and estimated trajectory (top), and the distance as a percentage of the travelled distance (bottom).

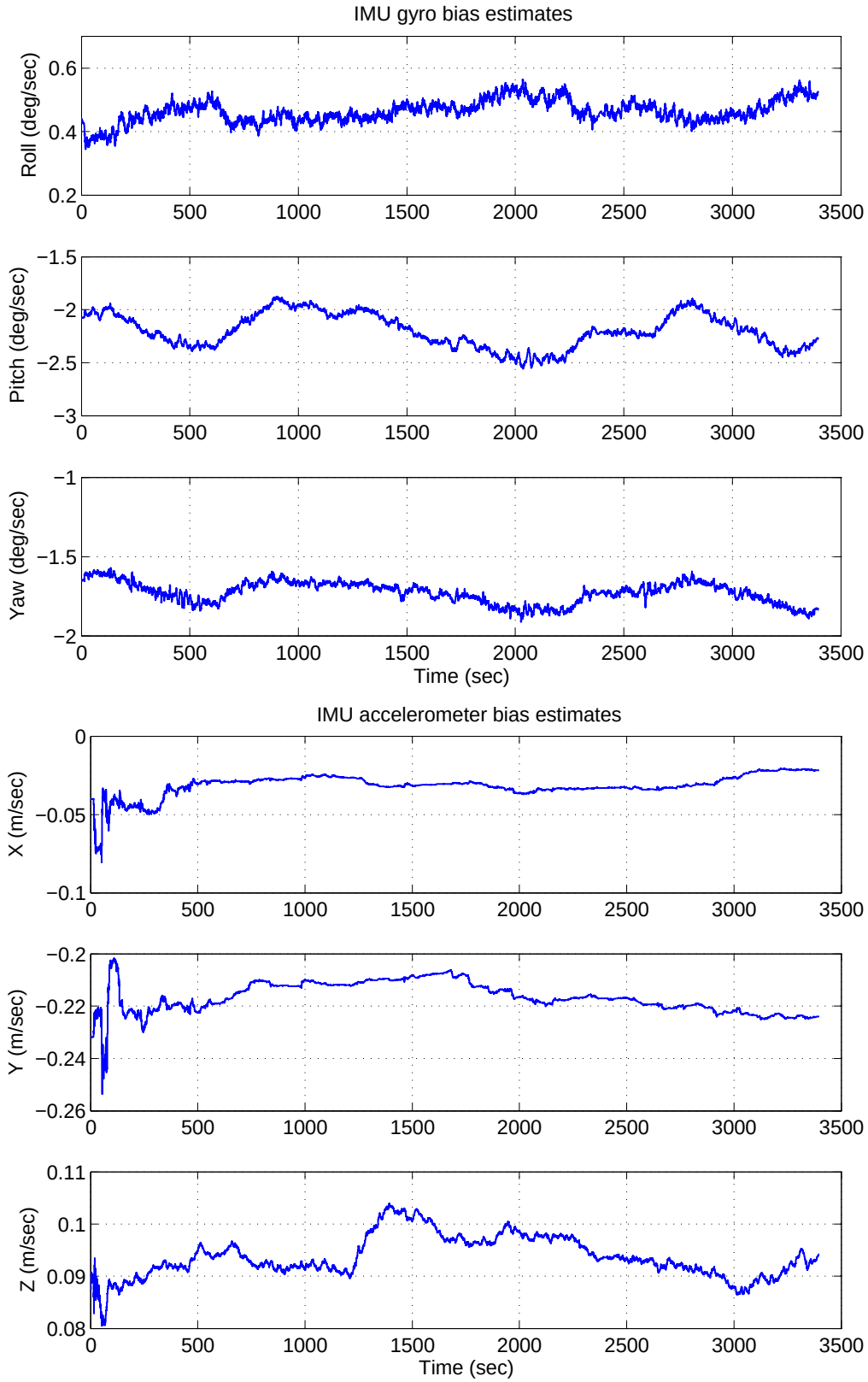

Figure 7: The MSCKF estimates for the gyroscope biases (top) and accelerometer biases (bottom).

7.1 Position estimation

In Fig. 4 we present the position estimates produced by the MSCKF, and compare them to the ground truth estimates obtained by the OXTS GPS/INS unit. The top figure shows the bird's eye view of the trajectory (i.e., the trajectory projected on the local horizontal plane), while the bottom figure presents the north-east-down coordinates as a function of time. Note that the MSCKF estimates the position in the ECEF frame, and the transformation into the NED frame is done only for presentation purposes.

In Fig. 4 we observe that the MSCKF filter estimates are very accurate, and that the estimated trajectory follows the ground truth trajectory very closely. To examine the quality of the position estimates in more detail, in Fig. 5 we plot the time evolution of the position errors along the north-east-down axes. From this figure we conclude that the errors on the local horizontal plane fluctuate in a range of approximately 80m around zero, while the errors for the altitude remain significantly smaller, less than about 25m for the entire trajectory. The RMS values of the position errors in each of the three axes is $[25.8 \;\; 20.3 \;\; 7.1]\; m$, respectively. These results for the position errors are explained by the fact that the accelerometer bias in the direction along the gravity vector is estimated very accurately, and thus the displacements in this direction can be estimated with high precision. On the other hand, the errors in the horizontal plane are strongly dependent on the errors in the yaw (shown in Fig. 11). Since the yaw is not observable given the IMU and camera measurements, the position accuracy in the horizontal plane is expected to be lower than the accuracy in the altitude.

The top plot of Fig. 6 illustrates the distance between the ground-truth and estimated trajectories (i.e., the norm of the position error vector) as a function of time, while the bottom plot shows this as a percentage of the distance travelled at each point in the trajectory. As expected, since the MSCKF is a visual/inertial odometry estimator, the position errors show a generally increasing trend. Interestingly however, the error as a percentage of the distance travelled remains small, and below 0.5% for most of the trajectory. Specifically, in the first 100 sec the error is a larger percentage of the distance travelled, due to the fact that in this initial phase the accelerometer biases are not yet well-estimated. Once the vehicle goes through the first few turns, the measurement information allows the biases to be estimated with good precision, and the error percentage drops significantly. Fig. 7 shows the estimates for the IMU biases over time. We can observe that in the first few seconds the accelerometer biases in the x and y axes undergo significant corrections, which shows that in this phase, their estimates are still uncertain. The corrections to the biases become subtler as time progresses.

7.2 Velocity estimation

We next turn our attention to the velocity estimates produced by the MSCKF. In Fig. 8 we plot the estimated Xsens velocity in the local vehicle frame (forward-right-down), and compare it to the ground-truth estimates of velocity. Internally, the MSCKF estimates the velocity in the ECEF frame, and the transformation into the local frame is done for presentation purposes only. As expected, the velocity vector is mostly along the forward axis, with smaller components in the right and down directions. In Fig. 9 we plot the difference between the ground-truth velocity and the estimated velocity, along the three axes of the local frame. From this plot, we observe that the difference in the estimates remains smaller than 0.2 m/sec for most of the trajectory, with "spikes" in the errors occurring around 500 sec, 1200 sec, and 1600 sec. The RMS values of the velocity errors are $[0.52 \;\; 0.09 \;\; 0.06]$ m/sec.

To examine the reason why the "error spikes" appear in Fig. 9, in Fig. 10 we plot the reported accuracy of the OXTS position and orientation estimates for the duration of the dataset (no data for the velocity accuracy are available). Interestingly, during the times when the difference between the MSCKF and OXTS velocity estimates is large, the OXTS reports poor accuracy. We therefore believe that during those times, the MSCKF estimates are *better* than the estimates reported by the GPS/INS navigation system. The large

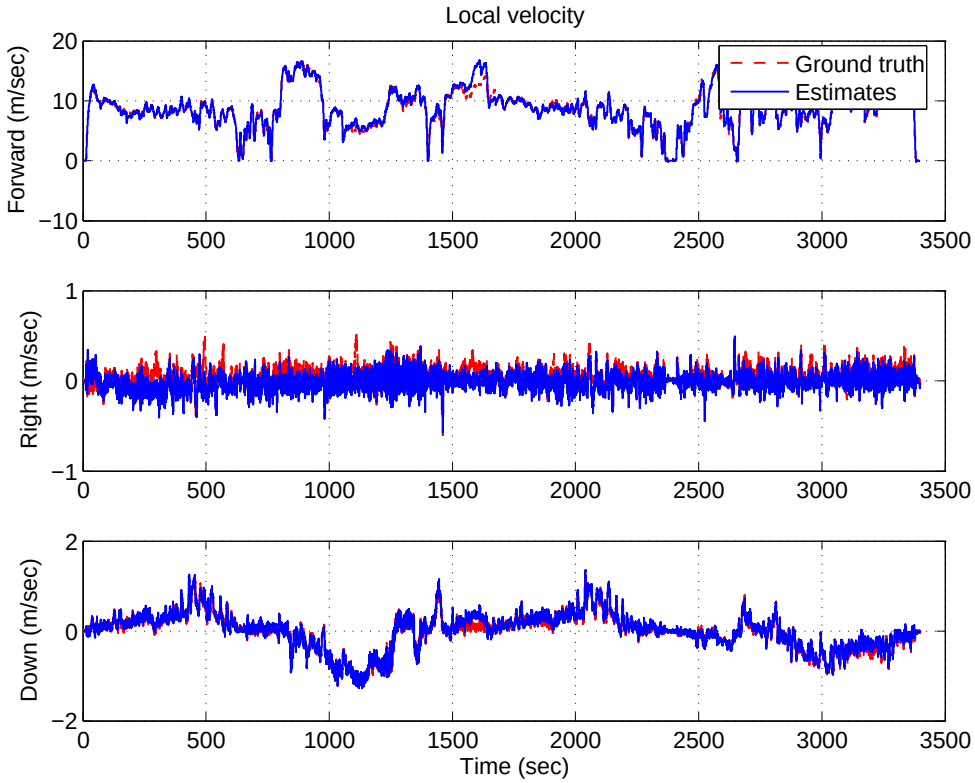

Figure 8: Velocity estimate (solid blue lines) vs. ground truth (dashed red lines) in local frame. The three subplots correspond to the forward, right, and down directions

errors in the GPS/INS estimates are apparently caused by dense foliage and/or the canyon walls obstructing the GPS signals.

As a final remark, we point out that the errors plotted in Fig. 9 include the effects of (i) errors in the velocity estimates of the MSCKF, (ii) errors in the velocity estimates of the OXTS unit, and (iii) errors due to the rotational velocity estimates of the OXTS unit. Since the Xsens and OXTS units are mounted in different parts of the vehicle, their velocity vectors differ when the rotational velocity of the vehicle is not zero (which is almost always the case, due to small vibrations, vehicle turns, and the vehicle's suspension). Therefore, to compute the ground-truth velocity of the Xsens we need to take into account the rotational velocity of the vehicle. This is expected to introduce additional accuracies in the ground truth estimates for the Xsens velocity vector.

7.3 Orientation estimation

In Fig. 11 we plot the difference between the MSCKF and the OXTS estimates of the roll, pitch, and yaw angles. Internally, the MSCKF estimates the orientation using a unit quaternion representation of the rotation from the ECEF to the Xsens frame, and the transformation to the local roll, pitch, and yaw is done for presentation purposes only. From the plots we can observe that the differences in the roll and pitch estimates are small, and remain below 0.5^o for most of the trajectory. The large spike appearing around 1600 sec coincides with the spike in the reported uncertainty of the OXTS unit, and therefore we believe that this large difference is due to large errors in the GPS/INS estimate, not in the MSCKF estimate. We point out that, as in the case of the velocity estimates, the differences plotted in Fig. 11 include the errors of *both* the MSCKF and the GPS/INS estimates. Specifically for the roll and pitch angles, the estimates of

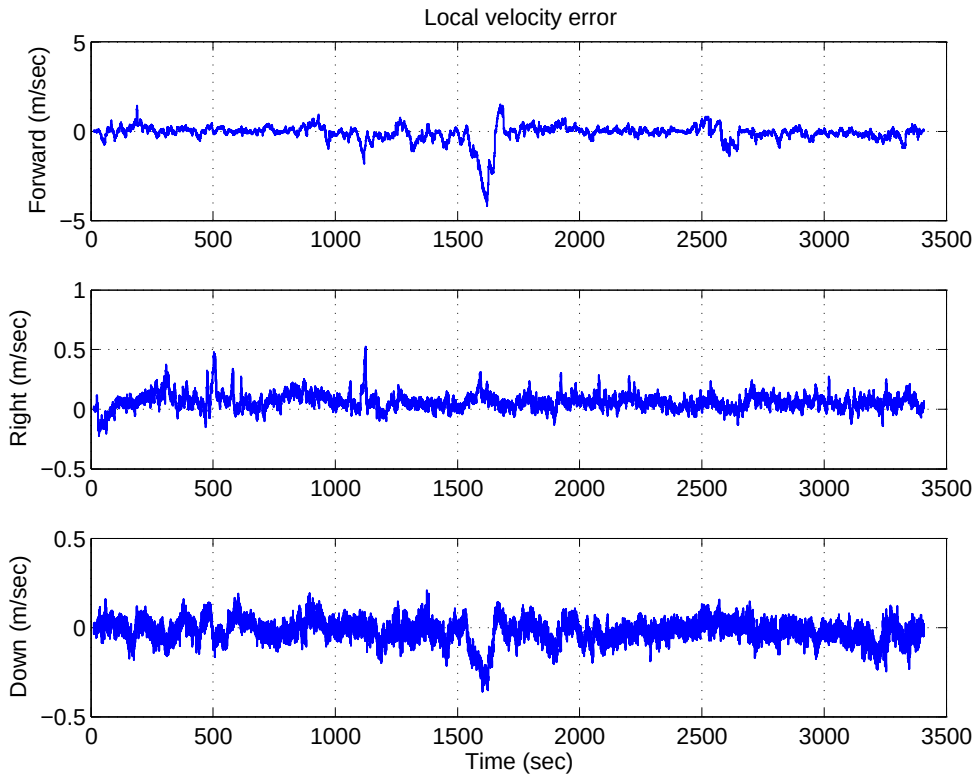

Figure 9: Velocity errors in the local frame. The three subplots correspond to the forward, right, and down directions

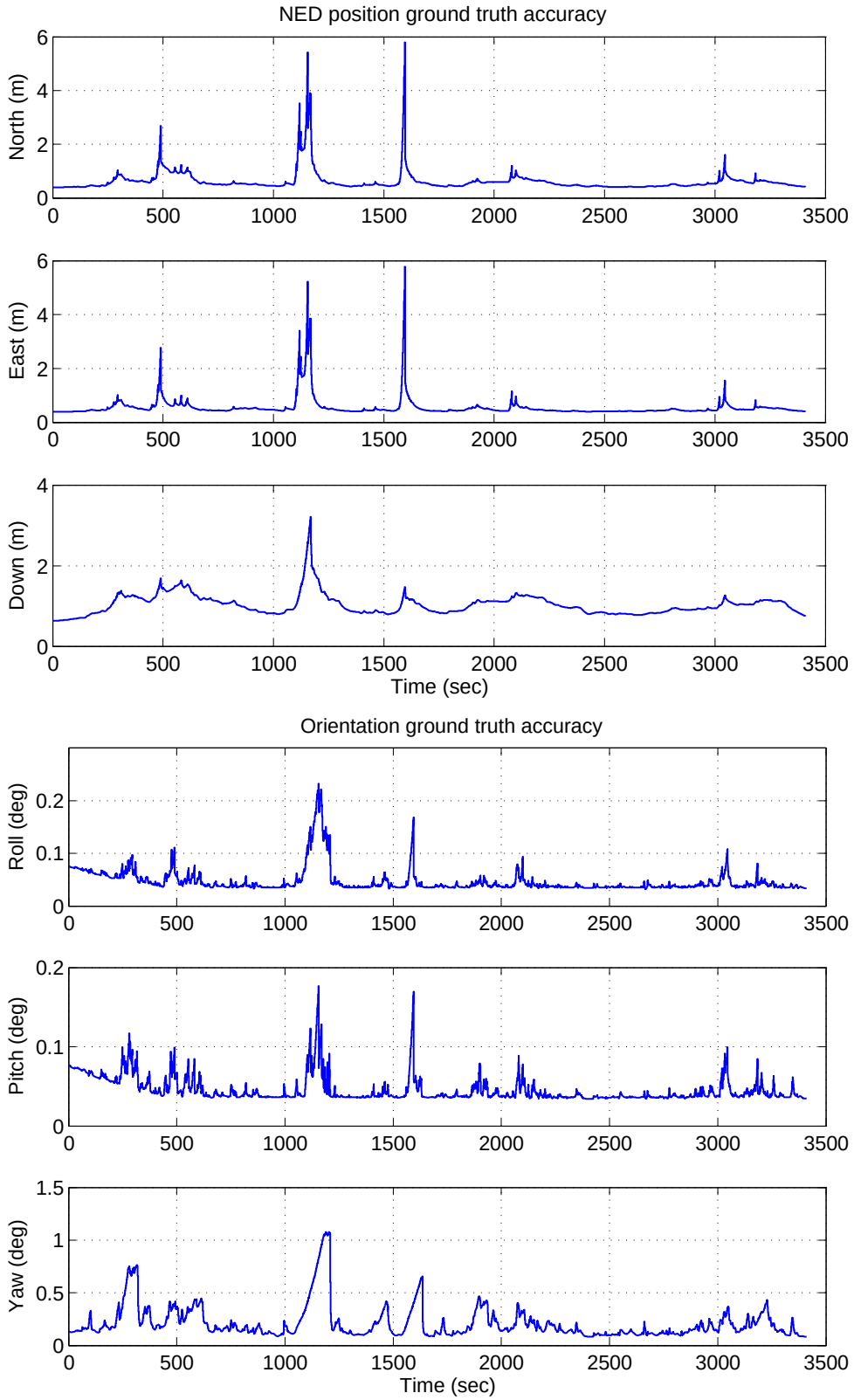

Figure 10: The reported OXTS ground truth accuracy. Top: position accuracy, with the three subplots corresponding to the north, east and down coordinates. Bottom: orientation accuracy, with subplots for roll, pitch, and yaw.

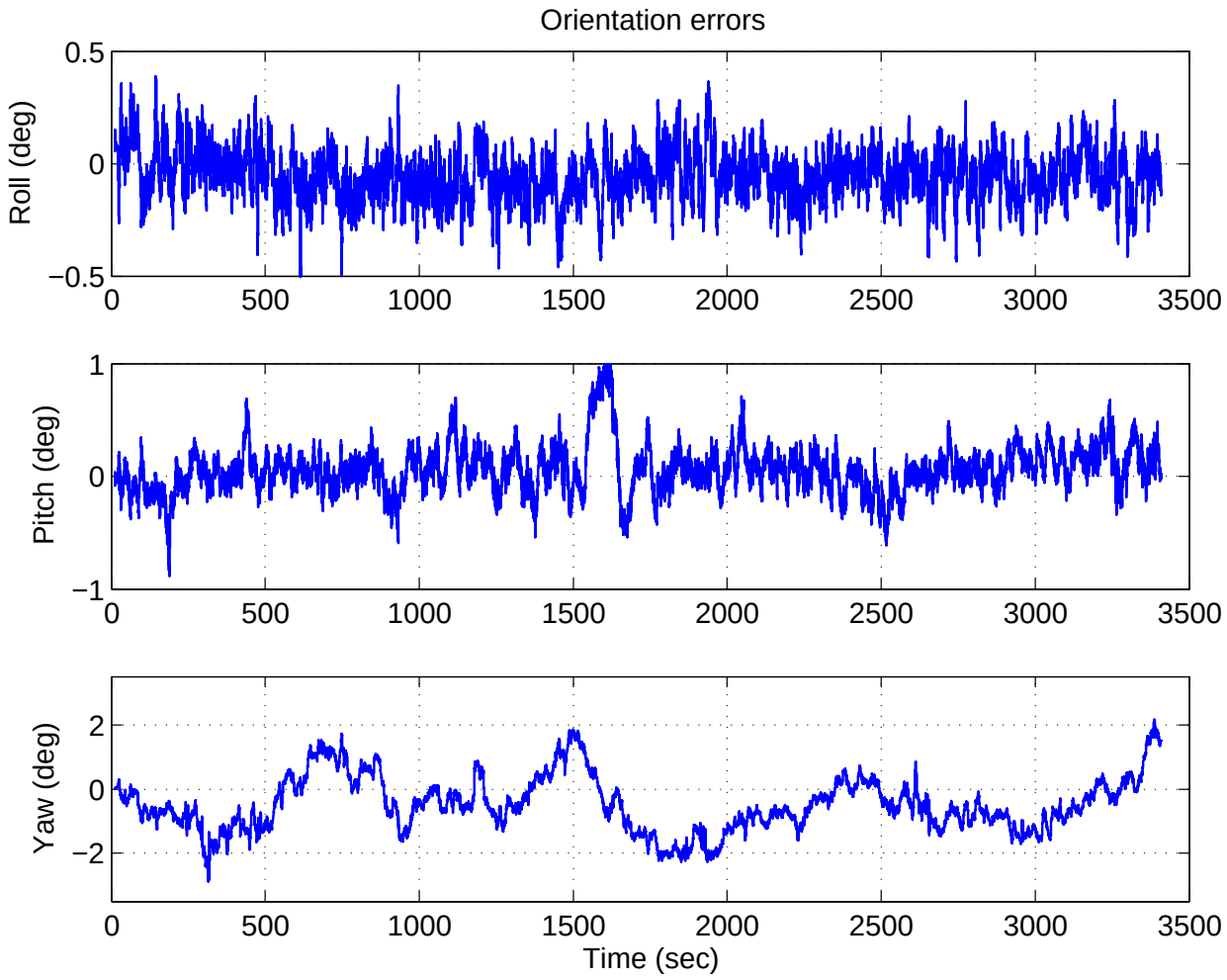

Figure 11: Orientation errors in the local frame. The three subplots correspond to the roll, pitch, and yaw.

both filters have very similar reported accuracies, and thus the contribution of the errors in the MSCKF and OXTS estimates to the differences plotted in Fig. 11 is expected to be comparable.

Regarding the local yaw estimates, Fig. 11 shows that the errors in these estimates range between approximately -2^o and 2^o for the duration of the experiment. As expected, the errors in the yaw are larger than those in the roll and pitch, since yaw is not observable given the available information. The RMS values of the difference between the MSCKF and GPS/INS orientation estimates are $[0.13 \;\; 0.22 \;\; 0.98]^o$.

7.4 Accuracy reported by the MSCKF

In Fig. 12 we plot the accuracy (1 standard deviation of the errors) reported by the MSCKF. The top three plots correspond to the accuracy reported for the roll, pitch, and yaw, while the bottom three plots correspond to the accuracy for the position in the north-east-down coordinates. We first note that the reported accuracy for the roll and pitch is approximately constant for the duration of the experiment, fluctuating at or below 0.1^o. As previously discussed, this is due to the fact that the rotations about axes in the horizontal plane are observable. On the other hand, the rotations about the gravity vector are not observable. Therefore, the reported accuracy in the MSCKF yaw estimates continuously increases for the first 1000 sec, and subsequently "stabilizes" around 0.6^o. This behavior of the orientation uncertainty can be attributed to

the fact that Extended-Kalman-filter based estimators tend to report lower covariances than the actual ones, when used for estimating unobservable states. This is a well-documented behavior, which has been studied primarily in the context of 2D Simultaneous Localization and Mapping (SLAM) [6–9].

The position uncertainty reported by the MSCKF exhibits a gradual increase, with reported σ for the position in the horizontal plane in the order of 50 m at the end of the trajectory, and approximately 5 m for the vertical axis. These reported accuracies reflect the fact that the position in the vertical axis is estimated with a significantly higher accuracy, compared to the position in the horizontal plane, which agrees with the errors reported in Section 7.1. Similarly to the behavior observed in the reported yaw accuracy, we believe the reported uncertainty for the position to be an underestimation of the actual uncertainty, due to the fact that the position is not observable given the measurements provided to the filter.

7.5 Algorithm time requirements

In Fig. 13 we present the time requirements of the MSCKF. Specifically, the top plot shows the time (in milliseconds) required for the MSCKF update for each image, while the bottom plot shows the time required for both the MSCKF update and for feature extraction and matching. These times were recorded on a desktop computer with an Intel i7 CPU at 2.93GHz, and with a single-threaded C++ implementation. On average, the MSCKF requires 7.8 msec per update, while the average total processing time per image (including image processing and filter updates) is 15.3 msec. Since the images are recorded at 20Hz, the time requirements of the MSCKF easily allow for a real time implementation. Out of more than 65000 images processed, only in 29 cases did the total processing time cross the 50 msec threshold required for real-time performance. If lower time requirements are desired, one could opt to implement the feature extraction and matching code on a GPU, and use a multi-threaded matrix library for the MSCKF.

7.6 Applying the MSCKF with different settings

Given a single dataset, it is not possible to collect statistics about the "expected" performance of the MSCKF in the given environment, and with the specific sensors available. However, we can get some sense of the stability of the results by running tests using different parameter settings. For the results presented so far, the MSCKF processed the data from the right camera only, and used a maximum of 2 features per cell (fpc) in a $16\times$ 16 cell grid in the image (see Section 3). We here present the results for the position drift under two additional scenarios: (i) using the images of the left camera only, and (ii) using a maximum of 3 features per cell in the image. Fig. 14 presents the position drift in these two situations, while plots for the remaining state estimates can be generated using the files accompanying this report. Note that using the left camera instead of the right one means that the noise in the image measurements is different in each case, and allows us to test the stability of the estimates with respect to the image noise realization.

The key observation is that the overall performance under all three scenarios is similar. The average position drift is 0.32% when using the right camera with 2 fpc, 0.37% using the left camera with 2fpc, and 0.49% using the right camera and 3 fpc. While towards the end of the trajectory the first set of parameters results in better accuracy, in the earlier parts of the trajectory the performance of the filters is similar, and in some cases the first set of parameters yields slightly worse performance. We conducted tests under a number of different scenarios (using the left or right camera, with or without the camera-to-IMU transformation in the estimated state vector, with different numbers of features in the images), and in all of them, the performance is similar to the results shown here. We thus conclude that the filter is quite robust to the choice of different settings and different realizations of the image noise.

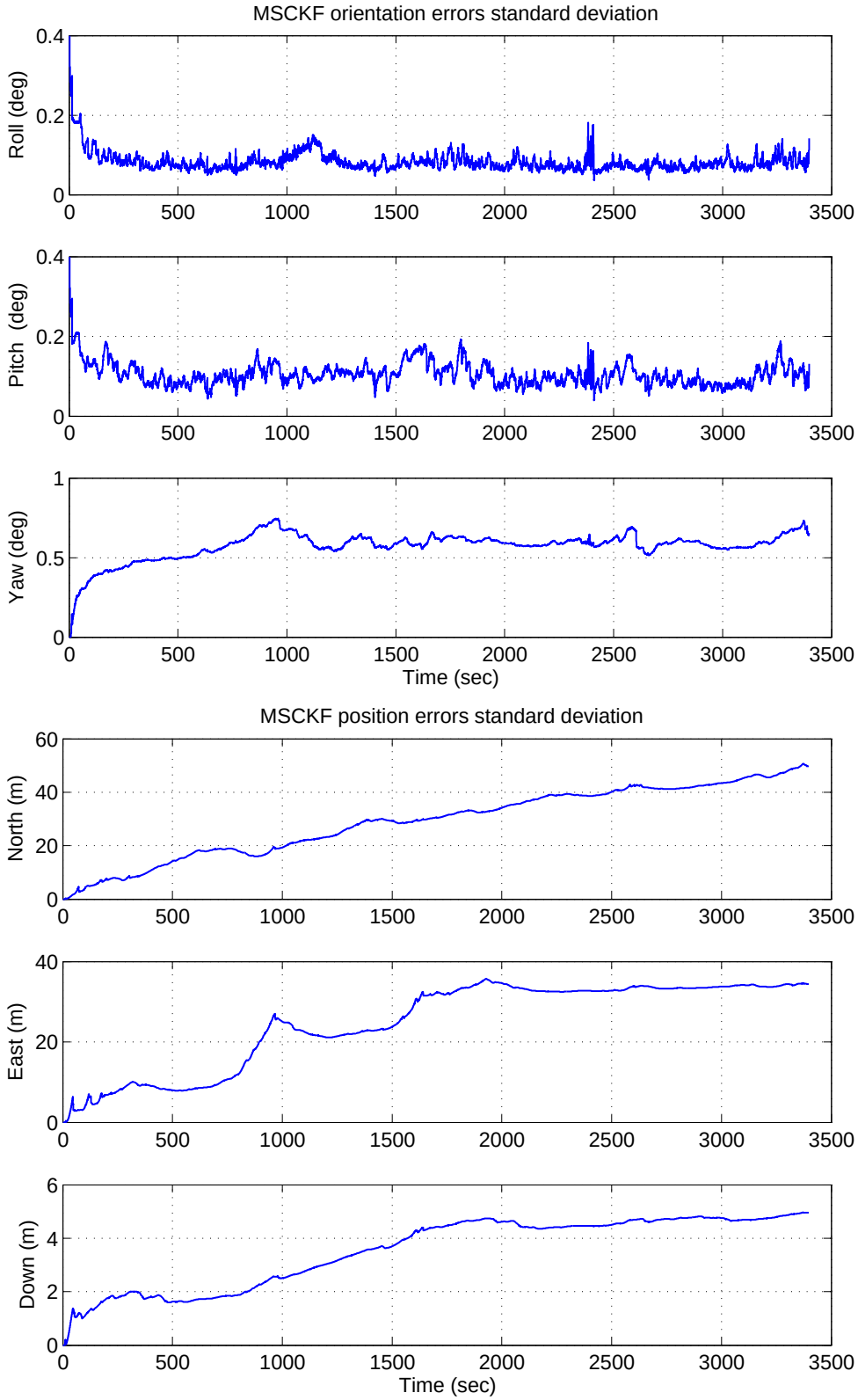

Figure 12: The estimation accuracy (1σ) reported by the filter. Top: orientation accuracy (roll-pitch-yaw), Bottom: Position accuracy in NED coordinates.

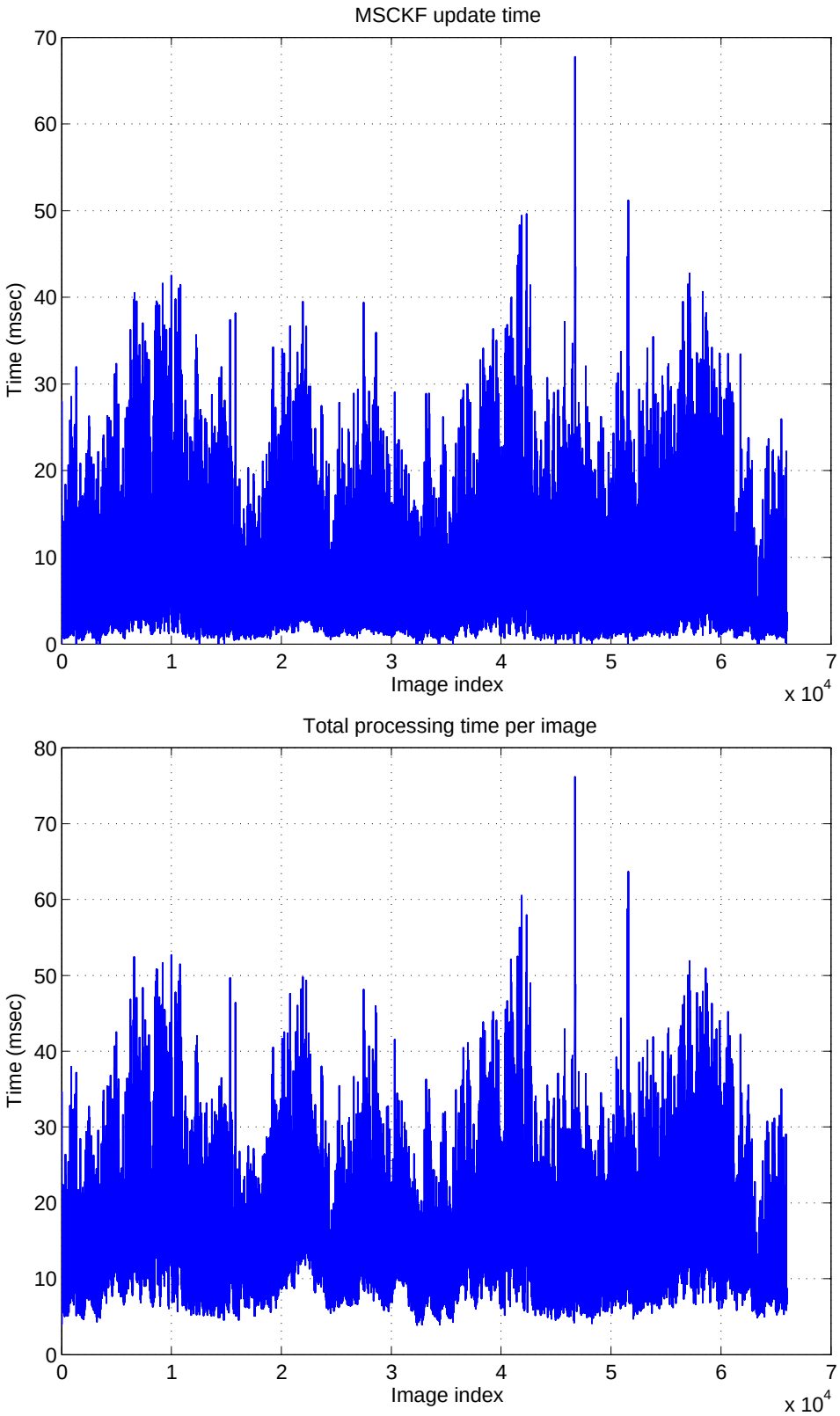

Figure 13: Top: The time required for each MSCKF update. Bottom: The combined time required for feature extraction and matching, and for the MSCKF update for each image.

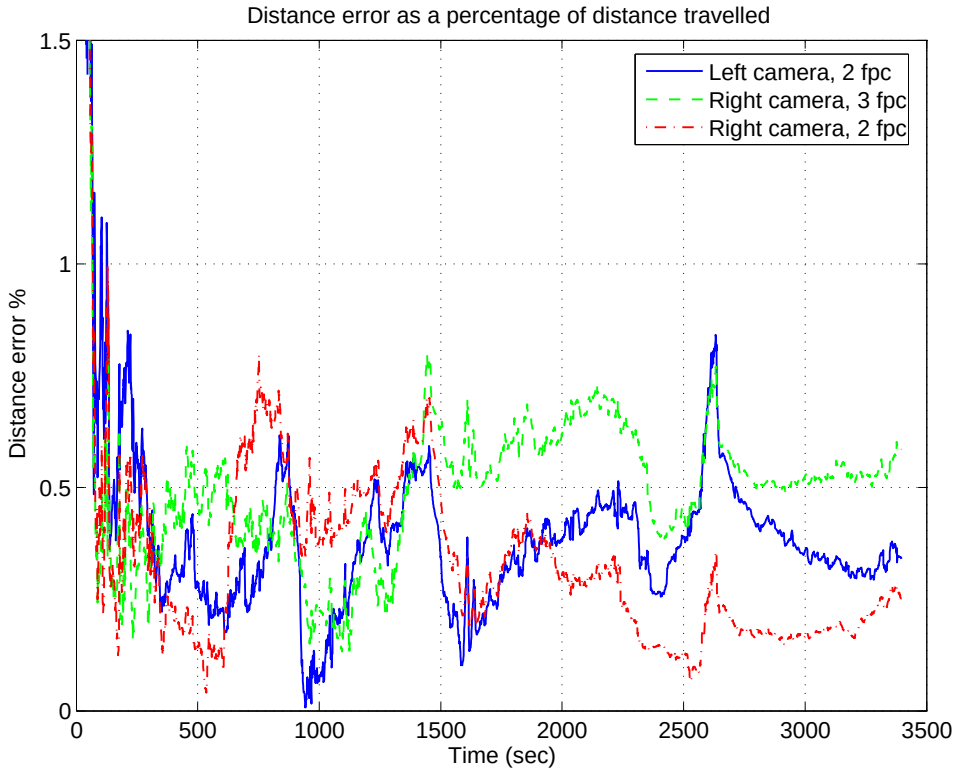

Figure 14: The position drift of the MSCKF using different settings.

8 Conclusions

This project has demonstrated that the MSCKF attains very good estimation performance in the Cheddar Gorge dataset. The position drift is in the order of 0.5% of the travelled distance, the yaw errors remain below 2.5^o, while the accuracy of the roll, pitch, and velocity estimates is comparable to that of the available GPS/INS ground truth. In fact, the MSCKF is capable of providing estimates with better accuracy than the GPS/INS system, in areas where dense foliage or the canyon walls obstruct the view to the GPS satellites. In contrast, the accuracy of the MSCKF does not exhibit any "blackout" periods, since the feature detection algorithm is able to find at least a few tens of features in all the images. Moreover, the MSCKF time requirements were shown to be well-suited for real-time operation.

8.1 Using additional sensor information in the MSCKF

Given the time available for this project, we were only able to test the MSCKF algorithm using a monocular image stream. However, use of the stereo images is possible in the MSCKF framework, and is expected to result in improved estimation accuracy, with small processing overhead (most of the additional time requirements would be in feature extraction and matching, which can be easily parallelized if needed). Additional sources of information that could be used, if needed, include the magnetometer measurements, and the non-holonomic motion constraints of the vehicle. The magnetometer readings provide absolute orientation information, and therefore render the yaw of the vehicle observable. Therefore, using magnetometer readings would result in a significant improvement of the accuracy, assuming these readings are free of systematic biases due to calibration errors. On the other hand, given that in this dataset the camera and IMU are mounted on a non-holonomic vehicle, we know that the lateral velocity of the IMU will be close to zero.

This can be translated as a zero lateral-velocity measurement, and can be employed in the filter for updates, even without the use of wheel encoder readings. We have opted not to use this constraint in our implementation, in order to demonstrate the performance of the pure visual/inertial odometry estimator, without any assumptions on the type of motion.

8.2 Suggestions for the data capturing system

Finally, this work has pointed out some possible directions for improving the quality of the captured sensor data, which would, in turn, improve the performance of any estimator used to track the vehicle trajectory. First, it is preferable to capture the IMU data in calibrated form, to avoid the presence of fast-changing, temperature-dependent biases, and of unmodelled effects such as scale factor changes, nonlinearity, and g-sensitivity. Second, the sensor package should be isolated from the vibrations caused by the motor as much as possible. Our experience with the dataset showed that the IMU readings have noise characteristics worse than their nominal ones, which is likely attributed to the vibrations affecting the sensor. However, given the requirements for having a rugged system, achieving better vibration isolation may be a difficult task. Lastly, in real-time trajectory estimation the time-offset between the IMU and camera images would have to be known, and the rate at which the data from each sensor becomes available should be as stable as possible. Our initial efforts showed that if the time offset between the sensors is not accounted for, the estimation results are significantly poorer.

References

[1] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Rome, Italy, Apr. 2007, pp. 3565–3572.

[2] ——, "A multi-state constraint Kalman filter for vision-aided inertial navigation," Dept. of Computer Science and Engineering, University of Minnesota, Tech. Rep., 2006, www.ee.ucr.edu/∼mourikis/tech_reports/TR_MSCKF.pdf.

[3] A. I. Mourikis, "Characterization and optimization of the accuracy of mobile robot localization," Ph.D. dissertation, University of Minnesota, June 2008.

[4] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[5] R. Simpson, J. Cullip, and J. Revell, "The BAE Systems Wildcat data sets," in *5th SEAS DTC Technical Conference*, Edinburgh, UK, 2010.

[6] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended Kalman filter based SLAM," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 1036–1049, Oct. 2007.

[7] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Analysis and improvement of the consistency of extended Kalman filter-based SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Pasadena, CA, May 2008, pp. 473–479.

[8] G. Huang, A. Mourikis, and S. Roumeliotis, "Observability-based Rules for Designing Consistent EKF SLAM Estimators," *The International Journal of Robotics Research*, vol. 29, no. 5, p. 502, 2010.

[9] S. Julier and J. K. Uhlmann, "A counter example to the theory of simultaneous localization and map building," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001, pp. 4238–4243.