Stereo Vision Aided Inertial Navigation in Forested Areas to Create 3D Models

Axel Andersson (axan0003@student.umu.se)
Linnea Kyrö Stenlund (liky0002@student.umu.se)
Anna Lundkvist (anlu0210@student.umu.se)
Hanna Markgren (hama0085@student.umu.se)
Konrad Steinvall (kost0004@student.umu.se)
Dennis Svedberg (desv0006@student.umu.se)
Mattias Tjernqvist (matj0016@student.umu.se)

22nd January 2017

Abstract

In order to get accurate 3D-models of forests in an efficient manner using a laser scanner attached to a backpack, the position of the laser scanner must be known at all times. In this project, the trajectory of a personnel carried backpack was tracked in a forested environment using visual stereo odometry in combination with an Inertial Navigation System (INS).

The systems were fused together in a somewhat binary manner. When the variance of the INS was above a certain threshold, the local odometry trajectory - supported by the directional angles of the INS - was imposed on the INS trajectory, and if the variance of the INS was below the threshold, the INS trajectory was used directly.

Using this method, the fused trajectory reduced the distance to known reference points by as much as 80% compared to the INS alone and by 87% compared to the odometry trajectory. But, the fusion method caused discontinuities where the fused trajectory went from using the stereo odometry to using the INS and vice versa.

In the future, further development of the fusion method is most likely required to in order to get accurate 3D-models from a laser scanner, especially in the discontinuity regions.

Contents

1 Introduction

Highly accurate 3D models of forests at ground level are very attractive for both commercial and scientific purposes. The modern 3D models are made using stationary laser scanners placed at known locations in the forest. This, however, is very time consuming. To make the collection of laser data more convenient, the idea of putting the laser scanner on top of a backpack was born. An important precursor to be able to use the laser data is knowing the exact position where the data was collected. But, in forest areas it is difficult to accurately track position using satellite based systems exclusively; as the signal can interfere with the tree canopy. It is therefore of great interest to develop a less time consuming, easier to carry and accurate positioning system that does not solely rely on satellite connection.

It is also possible to estimate the position by using stereo cameras. By matching points from cameras with a known distance between each other, the coordinates can be calculated using triangulation. The trajectory can then be calculated from a sequence of images from a stereo camera.

Thus, the aim of this research is to track the position using a stereo camera and a Inertial Navigation System (INS) and then fuse together the trajectory of the two systems to get a more accurate result when tracking the position in the forest.

2 Theory

2.1 Sensor Fusion Algorithm

Here we present a basic algorithm used for sensor fusion. Firstly, the open-source stereo odometry library `libviso2` was used to calculate the translation vectors t between the camera frames [1]. The translation vectors, t are in the cameras local coordinate system and can be transformed to global translation vectors, t^{global} by using roll φ, pitch θ, and yaw ψ angles from the INS according to Equation 1

$$t^{global} = R_x(\varphi)R_y(\theta)R_z(\psi)t \tag{1}$$

where R_x, R_y, and R_z are the usual 3×3 rotation matrices around the x, y, and z axes. Once all translation vectors are converted into a global coordinate system, the trajectory of the camera can be estimated. The position of the camera, x, at frame i, is given adding its global transition vector, t_i^{global}, with its position at the previous frame, namely x_{i-1}. Specifically, that is

$$x_i = x_{i-1} + t_i^{global}. \tag{2}$$

Using Equation 1 in Equation 2, the position of the camera at frame i is given by

$$x_i = x_{i-1} + R_x(\varphi)R_y(\theta)R_z(\psi)t_i \tag{3}$$

Thus, using Equation 3, it is possible to obtain an estimation of the trajectory of the camera. This trajectory will not drift over time. However, errors in the estimation of the transformation vectors will accumulate and cause the trajectory to drift over distance. The trajectory from the INS is calculated from both IMU (gyroscopes and accelerometers) and GNSS data. The IMU data will drift over time but the drift is for the INS trajectory corrected with GNSS data. However, the correction is only reliable in locations with good GNSS signals, for example in open areas. Also, the INS trajectory (position and orientation) is given in global coordinate systems whereas the trajectory from the camera is only given in coordinate system that is relative to other frames in the sequence of frames. The strategy was therefore to use INS data in areas where reliable data were produced from the INS and otherwise use the camera translation vector with orientation values from the INS (according to Equation 3).

In practice, this was achieved by investigating the size and changes of the standard deviation of the position, given by the INS log. If the standard deviation was above the threshold σ_t, or if the average (moving average) change of the standard deviation was above the threshold $(\Delta\sigma)_t$, the trajectory would be updated using Equation 3 and with the local translation vectors, t, obtained from `libviso2`

[1]. Otherwise, the trajectory would be updated with the position given by the INS. Algorithm 1 shows a pseudo code of the algorithm presented.

Data: For each frame take: the local transformation vectors t, the INS position x^{INS}, the standard deviation σ, the moving average range N, the roll angle φ, the pitch angle θ, the yaw angle ψ and the angle of the camera relative the Inertial Measurement Unit (IMU) α_{rel}.

Result: A fused trajectory x.

for k = *each frame* **do**

  if $k < 200$ **then**

    $(\Delta\sigma)_k = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{2\Delta t}(\sigma_{i+1} - \sigma_{i-1})$;

  else

    $(\Delta\sigma)_k = \frac{1}{N+1} \sum_{i=k-N}^{k} \frac{1}{2\Delta t}(\sigma_{i+1} - \sigma_{i-1})$;

  end

  if $\sigma_k > \sigma_t$ ***or*** $(\Delta\sigma)_k > (\Delta\sigma)_t$ **then**

    $x_k = x_{k-1} + R_x(\varphi_k)R_y(\alpha_{rel} - \theta_k)R_z(\psi_k)t_k$;

  else

    $x_k = x_k^{INS}$;

  end

end

Algorithm 1: The Algorithm shows how the INS trajectory is fused with the camera trajectory.

3 Method

3.1 Concept

The idea was to create a personnel carried system to accurately track our trajectory in a forest environment. In an open area a Global Navigation Satellite System (GNSS) is usually sufficient for position tracking. But, in a forest environment accurate position tracking becomes challenging - due to trees causing either poor or no connection at all - and the GNSS would have to be complemented by additional sensors. Thus, our system is comprised of an INS (consisting of an IMU and a GNSS) and a stereo camera. Employing stereo visual odometry a trajectory can be calculated by sequential camera images.

By using the algorithm presented in Section 2.1, the sensors could be fused together into a coupled system where they could complement each other. When the GNSS failed, the camera took over and vice versa. However, the angles given by the camera trajectory were unreliable and contained large errors, so that the trajectory in the global system became a very poor estimate of the actual trajectory. This was solved by supporting the camera trajectory with the angles given from the INS to get a better transformation between the local and global coordinate systems, and hence a more accurate trajectory.

3.2 Procedure

A stereo camera was programmed to capture images at 16 frames per second and time-stamp the images at the moment of shutter closure. The images were then processed in a visual odometry application, resulting in a motion trajectory.

An INS was used to record the position given by GNSS, and the linear accelerations as well as the angular velocities from the included IMU. The resulting INS data was treated in `Inertial Explorer`[2], and was improved by the SWEPOS ground station data [3].

The data from the camera and INS were then fused using the method described in Section 2.1, using the threshold parameters $\sigma_t = 0.5$ [m] and $(\Delta\sigma)_t = 6.4 \cdot 10^{-3}$ [m/s], and a moving average range of $N = 200$.

3.3 Equipment

The equipment used consisted of a stereo camera (Section 3.3.1), an INS (Section 3.3.2), and a computer (Section 3.3.3).

3.3.1 Stereo Camera

The camera used was a Bumblebee XB3 BBX3-13S2M IEEE1394b stereo camera by `Point Grey Research Inc` [4]. The Bumblebee XB3 consists of three separate cameras, each referred to as C_{left}, C_{mid}, C_{right}. The baseline between C_{left} and C_{right} was 240 mm, whereas the baseline between C_{left} and C_{mid}, and C_{right} and C_{mid}, was 120 mm in both cases. In the results presented below, the odometry used the C_{left} C_{right} camera pair, as this gave the best results.

Using `FlyCapture 2.1.6` [5] - a software development kit used to control and acquire images from Point Grey cameras - it was possible to acquire gray-scaled images from the three cameras simultaneously. Using the file-extension `.bmp`, the camera managed to acquire three simultaneous images, with a resolution of 1280×960 Pixels, at a frequency of approximately 16 Hz. `FlyCapture` also provided the feature of clocking the time when the shutter of the camera closed. This feature was used to accurately time-stamp the images.

3.3.2 Inertial Navigation System

The INS used was a SPAN-IGM-S1 made by `NovAtel` [6]. It features the Sensonor's STIM300 MEMS Inertial Measurement Unit combined with the OEM615 dual-frequency Global Navigation Satellite System (GNSS) receiver [7]. The antenna used was a GPS-702-GG, also made by `NovAtel` [8].

3.3.2.1 IMU

The STIM300 contains three micro-machined electromechanical systems (MEMS) gyroscopes and three solid state accelerometers [9].

MEMS gyroscopes use the Coriolis effect to calculate the angular velocity. The Coriolis effect is that a mass m moving with a velocity v in a frame of reference rotating at angular velocity ω experiences a force

$$F_c = -2m(\omega \times v) \tag{4}$$

A MEMS gyroscope contains vibrating elements that measure the Coriolis effect. Some advantages for MEMS gyroscopes are that they are small, weigh little and have low power consumption [10].

3.3.2.2 GNSS

The INS used two global navigation satellite systems, GPS and GLONASS. GPS, short for Global Positioning System is developed by the United States, while GLONASS, short for GLObal NAvigation Satellite System, is Russia's version.

The systems have three main components; a space segment, control segment and the user segment [11].

The space segment contains the satellites, of which GPS has arranged into six orbital planes around the Earth. The signal generated on the satellites is based on the fundamental frequency 10.23 MHz, generated by atomic clocks. There are two carrier frequencies, L1 and L2 that are generated by multiplying the fundamental frequency with integers 154 and 120.

On these carrier frequencies pseudo-random noise codes, satellite ephemerides, ionospheric modeling coefficients, status information, system time and satellite clock corrections are overlaid.

The travel time for the signals between the satellites and the receiver is used to compute the distance, which is called the pseudo-range, since it is affected by errors and is not the true range [12].

The control segment is a global network of facilities that monitors the transmissions of the satellites. In Colorado there is a master control station that controls the satellite constellation. The other stations are monitor stations that track the satellites and send their observations to the master control station. If one of the satellites were to fail, the master control station should reposition the satellites for best possible coverage [13].

The user segment is the receiver of the users, in our case the SPAN-IGM-S1. It converts the signals into time, position and velocity estimates.

The signal processor in the receiver reconstructs the carriers and extracts the navigation messages. Then the receiver compares the received signal with a reference signal since the motion of the satellite has Doppler shifted the signal. By using code correlation between the two signals the received signal is shifted with respect to time.

Two reference systems are used. One for describing the motion of the satellites, that is space-fixed and inertial, and one that is earth-fixed for the position of the observation stations. It is to the latter that the position of the receiver is computed. The earth-fixed reference system has by convention three axes, the Z-axis goes with the rotation axis of the earth, the X-axis is along the Greenwich meridian and Y-axis is orthogonal with them both.

One type of error is multipath error which is when the signal is reflected so that there are multiple reflections at the receiver. There are also many other types of errors that may affect the results [11].

3.3.3 Computer Specifications

The computer used to acquire data from the INS and camera was a Dell Latitude E6420 using Windows 7. The computer was chosen due to it having an express card slot which allows for a IEEE1394b connection between the camera and computer.

`Microsoft Visual Studio 10` together with `FlyCapture` were required in order to create a program that could capture and time-stamp images. `Python` was used to simultaneously run and log data from both the INS and the camera. `NovAtel Connect` was used to synchronize the computer clock with the GNSS time.

The software estimating trajectories was confirmed working on computers running either LINUX, Windows 7 or Windows 10 separately. A few dependencies were required to estimate the trajectories. First, `MATLAB R2016b` with an installed C++ compiler `MinGw 4.9.2` (for Windows), was required to run the program `LIBVISO2` (Library for Visual Odometry 2). Finally, the INS data was post processed using `Inertial Explorer`.

3.3.4 Inertial Explorer

`Inertial Explorer` is a program from `NovAtel`. It was used to align the INS and post process the data. Aligning the INS changes the coordinates from the local coordinate system to the global one. Unless this is done, the angles from the INS will all be zero. It is possible to align without using `Inertial Explorer`, but that is easiest done when moving in a straight line in at least 5 m/s. Since the INS is not mounted on a vehicle but carried by a person, this is quite difficult. Therefore `Inertial Explorer` was used instead.

When `Inertial Explorer` was used there was no need to have a special initialization process. But at the beginning and end, good satellite connection is needed to make sure that the aligning process will work correctly. It is also good to start the INS a few minutes before logging data. This is so that it will have time to find satellite connections.

When post processing in `Inertial Explorer` it is possible to add data from a base station to get more accurate results. To process the data `Inertial Explorer` uses Kalman filters [2]. In Appendix A.4.3 the process how to use `Inertial Explorer` to post process data is described.

4 Results

Trajectories were estimated using three different methods. The first method uses the camera incorporated with the angles from the INS, the second method uses solely the INS unit, and the third method uses the the algorithm presented in Section 2.1 to fuse the camera together with the INS unit.

4.1 Test 1: Walk Through a forest and Into a Building

This test was made near the SLU building in Umeå. The route started outside the main entrance of the building and continued along a pathway until a detour into a forest was made. The tree height in the forest was about 10 meters[14]. Later on the route came out on the pathway again and into the SLU building at another entrance, going down two flight of stairs on the way to the original starting point three levels below. In the building the satellite connection was lost completely. The walk continued through the whole building and ended outside the main entrance, where the starting point was. A representation of this route is presented in Figure 2. The estimated trajectories along with reference points for the three mentioned methods for the route described are showed in Figure 1 and Figure 3. The reference points were made with a high precision GPS, Trimble GeoXR 6000.

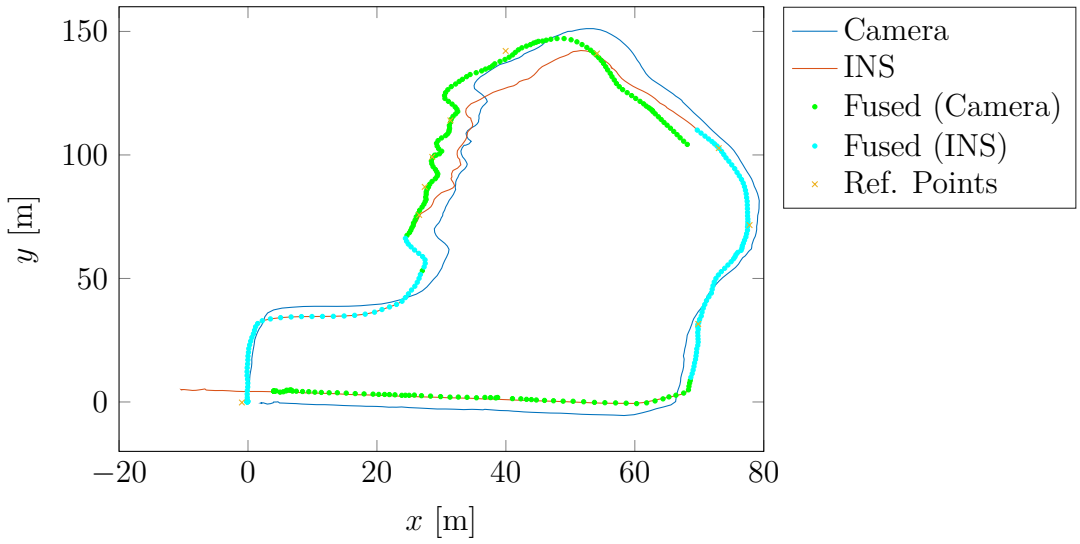

Figure 1 – The calculated trajectories for the three methods with reference points.

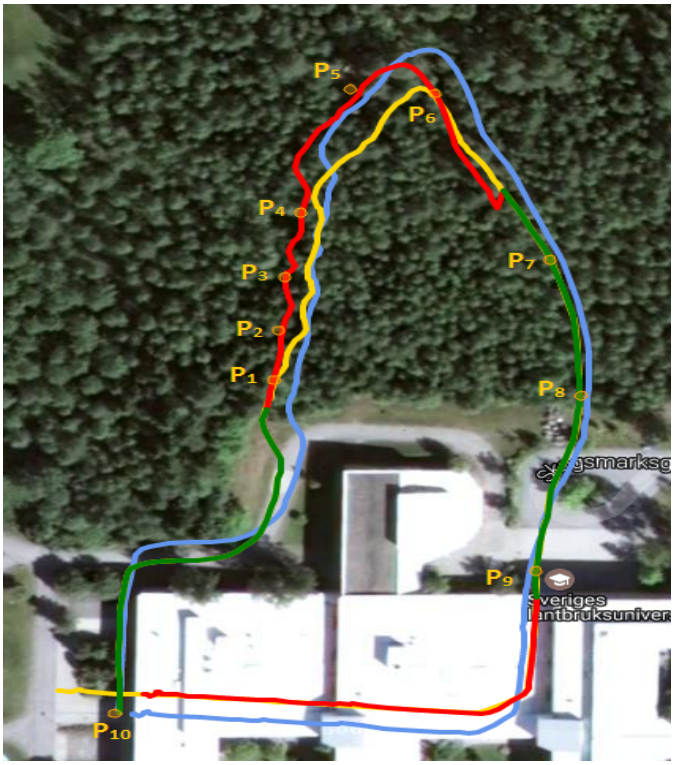

Figure 2 – A Google Maps representation of the path walked. The orange circles indicate reference points, the blue line the camera, the yellow the INS and the green-red line the fused trajectories. The red parts indicates that the camera is used and the green parts the INS.

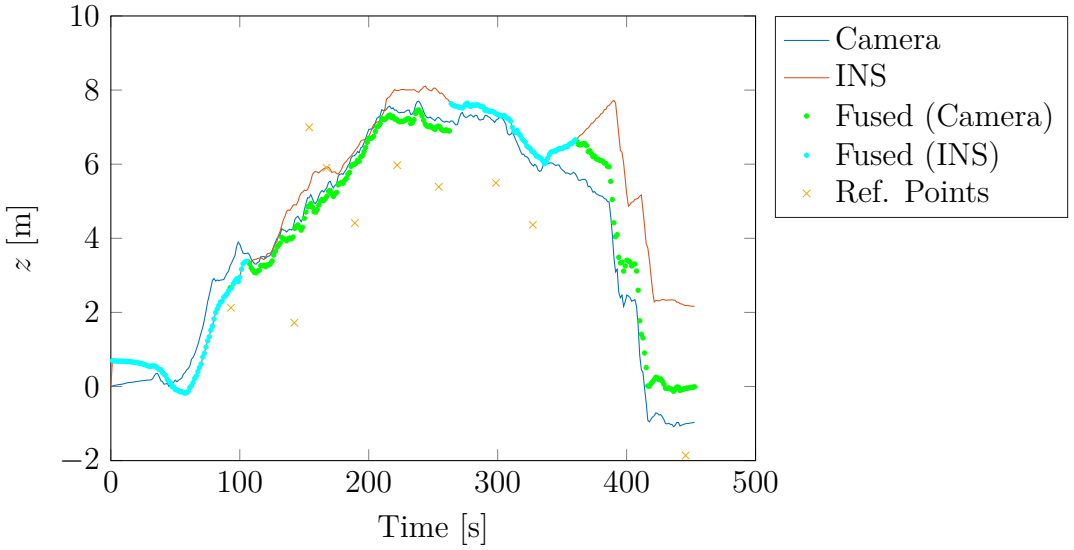

Figure 3 – The altitude for the three different methods together with reference points.

The standard deviation and its rate of change can be seen in Figure 4 and in Figure 5 respectively, with the fusion threshold visualized.

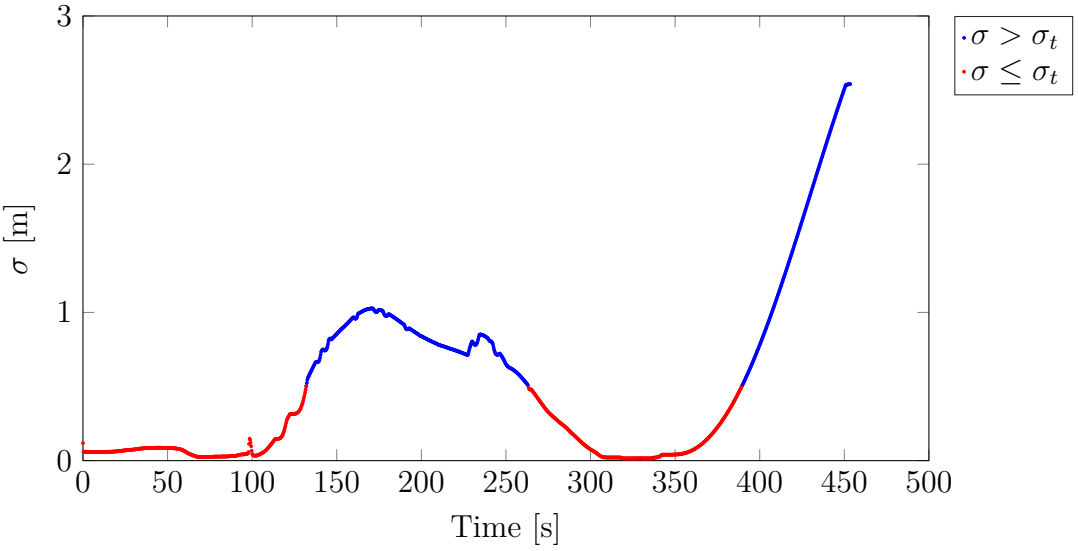

Figure 4 – The standard deviation of the position obtained by the INS, after post-processing in Inertial Explorer, is graphed against time. In the blue region the criteria $\sigma > \sigma_t$ is fulfilled, and the fused trajectory is updated using the camera.

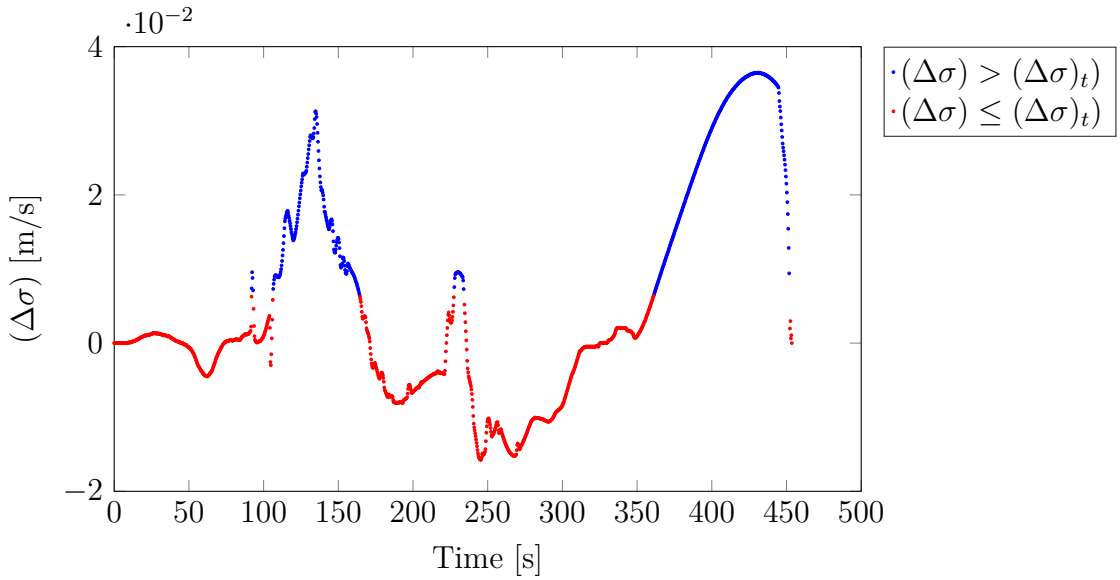

Figure 5 – The change of the standard deviation of the position obtained by the INS, after post-processing in Inertial Explorer, is graphed against time. In the blue region the criteria $(\Delta\sigma) > (\Delta\sigma)_t$ is fulfilled, and the fused trajectory is updated using the camera.

The difference between the three methods can also be seen in Table 1 where the minimum distances between the reference points and the different trajectories are shown. It can be seen that the fused method produces a lower error compared to the INS.

Table 1 – The Table contains the minimum distances in the x-y plane between the respective trajectories relative the exact point points $P_1 \dots P_{10}$. The trajectories start and end at P_{10}, going in a clockwise direction as illustrated in Figure 2.

Ref. Pts.	P_{10}^{start}	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}^{end}
Camera [m]	0.86	3.57	4.73	4.57	4.32	3.04	5.23	1.83	1.06	0.96	3.11
INS [m]	0.86	0.13	3.50	3.86	3.06	8.30	0.29	0.01	0.25	0.02	11.4
Fused [m]	0.86	0.25	0.70	0.19	0.18	1.53	0.07	0.01	0.25	0.02	6.9

As shown in Table 2 the relative error between points in difficult forest conditions ($P_2 \dots P_6$) can be very high for the INS. Our fusing method shows greatly reduced relative errors in this type of environment. The error at P_{10} shows the need to use another method for aligning the camera path, such as a digital compass. Here the incorrect yaw estimation impacts the fused path negatively.

Table 2 – Relative error between waypoints. Trajectories were reset at each waypoint thus only measuring error between each waypoint. Percentages represent error divided by distance traveled.

	Camera [m]	INS [m]	Fused [m]
P_1	5.23 (5.4%)	1.27 (1.4%)	0.43 (0.5%)
P_2	1.24 (8.1%)	2.95 (25.6%)	1.24 (9.6%)
P_3	0.56 (4.8%)	3.86 (34.5%)	0.56 (4.8%)
P_4	0.46 (2.9%)	1.85 (8.5%)	0.46 (2.9%)
P_5	1.47 (4.4%)	4.17 (14.3%)	1.47 (4.5%)
P_6	1.63 (8.7%)	6.38 (29.5%)	1.62 (8.6%)
P_7	7.14 (14.4%)	3.44 (7.5%)	0.13 (0.27%)
P_8	2.06 (5.8%)	0.49 (1.4%)	0.49 (1.6%)
P_9	2.52 (5.9%)	0.65 (1.5%)	0.65 (1.5%)
P_{10}^{end}	5.36 (5.4%)	10.81 (10.0%)	6.78 (7.14%)

4.2 Test 2: Walk Through a Forest

In an area called Gammlia in Umeå there is a much more older and denser forest compared to the forest in test 1, therefore it was chosen for the next test. Here the heights of the trees varied between 10 and 27 meters[14]. Reference points in this area had already been measured with a RTK GNSS in combination with a Trimble total station. A representation of the route walked is shown in Figure 6, where the reference points 504, 501 and 503 are marked with orange circles. The route started in the open area on the right before going into the forest and ended at the same spot.

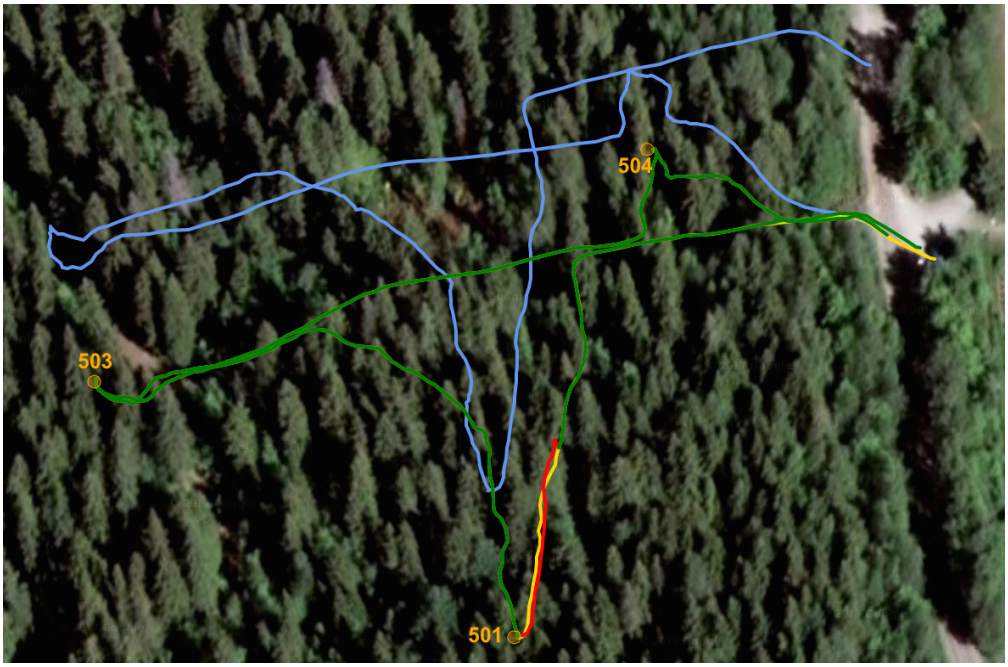

Figure 6 – A Google Maps representation of the path walked. The orange circles indicate reference points, the blue line the camera, the yellow the INS and the green-red line the fused trajectories. The red parts indicates that the camera is used and the green parts the INS.

Figure 7 and Figure 8 show three similar trajectories through an older and more dense forest alternating between following a wider trail and none at all.

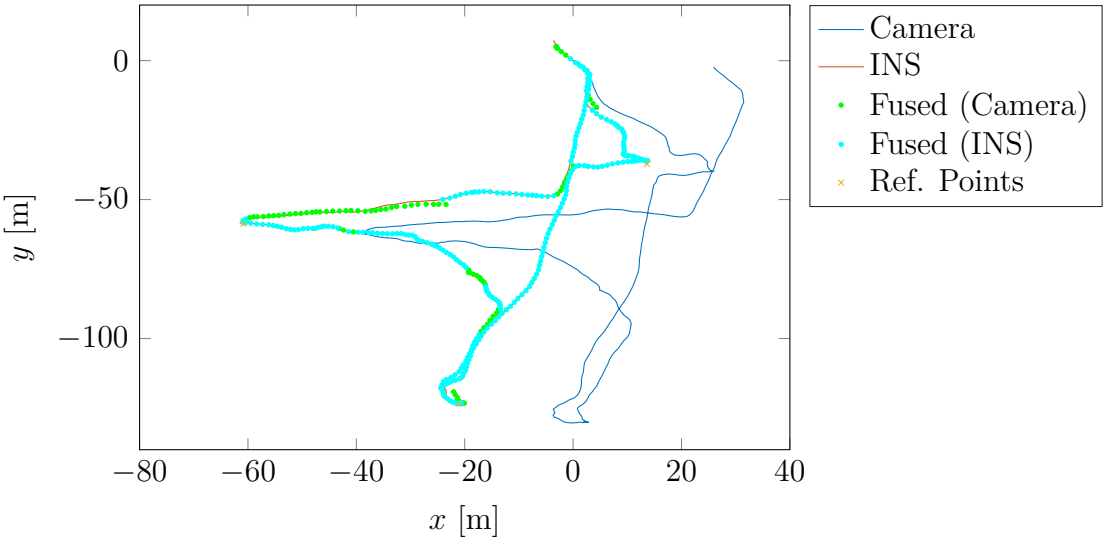

Figure 7 – The calculated trajectories for the three methods with reference points.

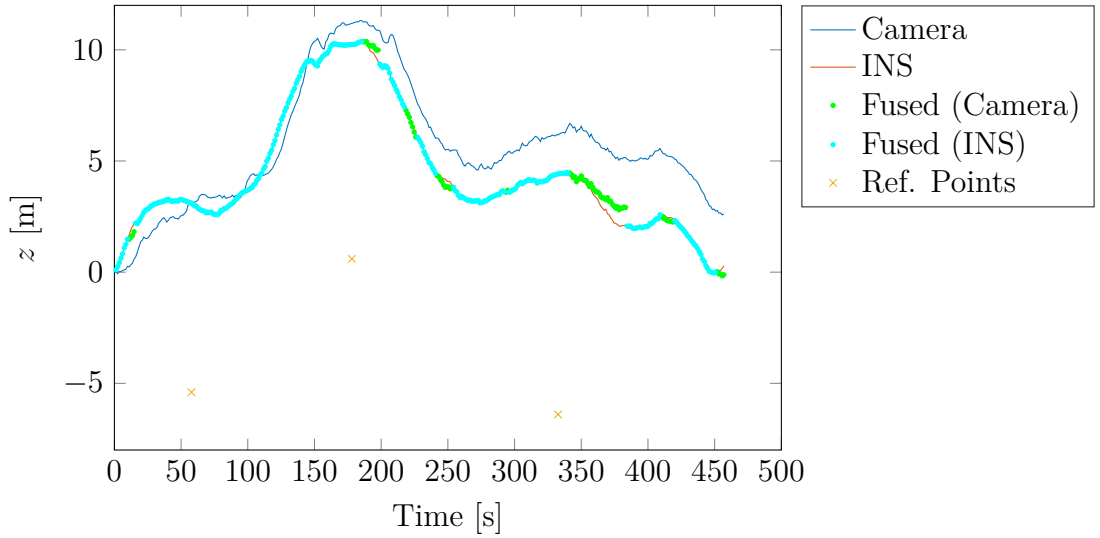

Figure 8 – The altitude for the three different methods together with reference points.

In these figures the standard deviation of the INS positions was very low, as seen in Figure 9 and in Figure 10, resulting in that the fused trajectory largely overlapped with the INS trajectory.

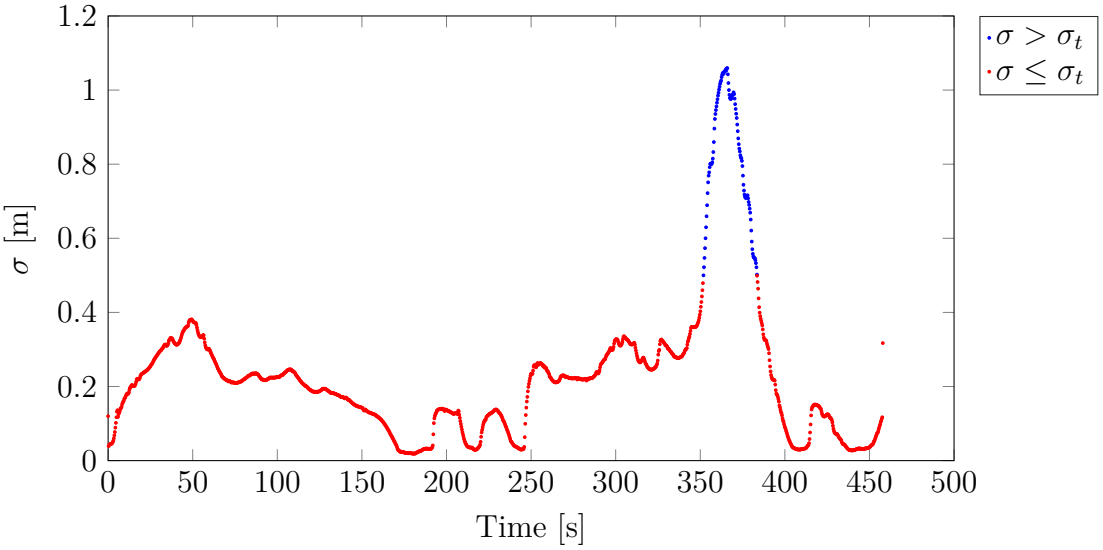

Figure 9 – The standard deviation of the position obtained by the INS, after post-processing in Inertial Explorer, is graphed against time. In the blue region the criteria $\sigma > \sigma_t$ is fulfilled, and the fused trajectory is updated using the camera.

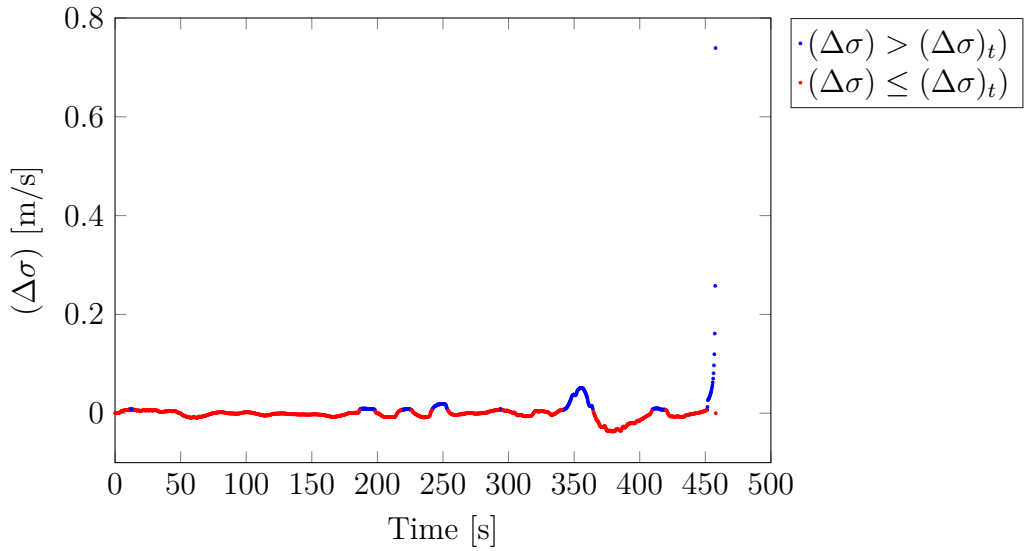

Figure 10 – The change of the standard deviation of the position obtained by the INS, after post-processing in Inertial Explorer, is graphed against time. In the blue region the criteria $(\Delta\sigma) > (\Delta\sigma)_t$ is fulfilled, and the fused trajectory is updated using the camera.

The difference between the three methods can also be seen in Table 3 where the minimum distances between the reference points and the different trajectories are shown. It can be seen that difference in error between the INS and the fused method is very small.

Table 3 – The Table contains the minimum distances in the x-y plane between the respective trajectories relative the exact point points 504, 501 and 503.

Ref. Pts.	504	501	503
Camera [m]	4.65	22.67	17.42
INS [m]	0.94	0.39	0.07
Fused [m]	0.94	0.41	0.07

5 Discussion

5.1 Methods

5.1.1 Choice of Camera

The stereo camera used was a Bumblebee XB3 multi-baseline stereo camera. It was chosen due to its flexibility, its multiple baselines, and the fact that the camera was already calibrated by the manufacturers to grab images at the exact same time.

The advantage of two different baseline is that a longer baseline provides more precision when performing stereo-odometry on objects at longer distances whereas the narrower baseline provides more precision on closer objects.

After some investigation we found that the camera parameters calibrated by the manufacturers were off. For instance, rectification of images using either Matlab's Stereo Camera Toolbox or Triclops with the the manufacturers camera parameters gave poor results.

The camera was mounted on the backpack such that it pointed towards the ground with an angle of $\alpha_{rel} = 18.09°$. We also investigated how our stereo odometry code would cope with the camera angles $\alpha_{rel} = 35°$, and $\alpha_{rel} = 5°$. It turned out that in both cases the number of matches would be significantly lower than in the cases of $\alpha_{rel} = 18.09°$. It was also very important to accurately measure α_{rel}. An error of $\pm 1°$ results very significant errors in the camera trajectory's z-direction.

5.1.2 Choice of Camera Software

One of the first decisions we had to make was to choose which method of image acquisition we were going to use. The two main choices were to use either `Triclops` or `FlyCapture`. After some investigation we found that `Triclops` could be used to, for instance, save rectified images directly - and depth-maps as well - but when doing so the image capture rate was decreased drastically as the computer needed to process every image over a longer period of time. It could be used to capture raw images as well, but it was slightly more problematic than `FlyCapture`. One of the advantages of `FlyCapture` was the simplicity of adding timestamps to the images and the possibility of altering other settings of the camera easily.

We chose to use `FlyCapture` as the main image capturing program instead of `Triclops` for the reasons mentioned above, but it was also unnecessary for us to generate depth-maps since our visual odometry code used raw images. We chose to save the images in `bmp` format because other file formats such as `png` or `jpg` compresses the image which - while reducing the image size - also reduces the frame rate. Since data storage was not an issue for us, the larger files with higher frame rate were chosen over the other alternatives.

5.2 Alternative Odometry Methods

Another approach investigated to solve the problem of trajectory estimation was `Google Cartographer`. While it probably could have been very useful, we felt we lacked the time needed to apply it to our problem. Also, as the software was very new there was not much documentation to aid us. Hence, we came to the conclusion that we focus our attention on other approaches. However, it could be interesting to investigate further in the future.

In the end, we chose to use the library `libviso2` as the basis for our stereo odometry code due to its robustness and lack of external dependencies. It has also been used in similar applications [15].

5.3 Alternative Fusion Methods

Another commonly used method for sensor fusion that we investigated thoroughly was the *Kalman filter*. We wanted to use Kalman filtering due to it being very useful in these sort of applications - especially for autonomous vehicles [16]–[18]. Unfortunately, we were unable to implement a working Kalman filter in the allotted time - so we had to settle for the method described in section 2.1. As Kalman filters work very well if the input parameters is correctly estimated and very badly if they are incorrectly estimated, the reason for not finishing a working Kalman filter in time was largely due to issues in estimating the input parameters

- mainly covariances. For future development a Kalman filter should probably be implemented for a more accurate trajectory estimation.

5.4 Fusion Algorithm

We chose to use the standard deviation, and the change of the standard deviation, of the INS positions as a quality check in our fusion algorithm. While the INS underestimated its standard deviation when it had a bad connection, it still gave a higher standard deviation than it had under a good connection. Thus, we looked at the size of the standard deviation and its change to gauge the accuracy of the measurement.

The fusion thresholds σ_t and $(\Delta\sigma)_t$ were chosen such that the camera would be used during the larger peaks in Figures 4 and 5.

As seen in Figure 4, the standard deviation threshold is reached at around $t = 140$ [s]. At this time, the INS would already have drifted away from the true path. In fact, the INS starts to deviate from the true path when the standard deviation begins to increase notably, at around $t = 110$ [s]. Therefore, it is of interest to investigate the rate of change of the standard deviation, as seen in Figure 5. This motivates our choice of using two thresholds.

The threshold values used were $\sigma_t = 0.5$ [m] and $(\Delta\sigma)_t = 6.4 \cdot 10^{-3}$ [m/s] were chosen to be below the peaks in their corresponding spectra. These threshold values needs to be altered to optimize the fused trajectory in different environments. To avoid that rapid fluctuations in $(\Delta\sigma)$ surpassed the fusion threshold, the decision of making a moving average of $(\Delta\sigma)$ was made. This removes some of the stochastic noise, leaving us with only the significant variations.

This method could be improved by also using the variance of the camera's trajectory to determine which instrument would be appropriate to use. Unfortunately, we were unable to find this variance which would also have allowed us to develop a Kalman filter. Another measure of quality could have been the number of matches between the time steps of the odometry, but we could not develop a satisfying way of taking this into account.

One of the main issues with this method is that when the trajectory goes from being camera dependent to becoming INS dependent, we get a discontinuity in the complete trajectory. This is obviously a big problem, and one way of solving it could be to do the same procedure backwards and trying to match the two trajectories near the discontinuity. This problem might also be solved by using a better directional support to the camera's trajectory that drifts less over time, such as a highly accurate digital compass.

5.5 Results Discussion

The reason why the INS gives inaccurate position in the forest is because of multi-path errors, i.e. the signals from the satellites are reflected on the trees and their foliage and this leads to that the signals reaches the receiver at different times. Since the signal is not completely blocked out by the trees the receiver estimates that it's positional error is smaller than it actually is in reality. Because of this the combined result of the IMU and GNSS is less accurate than it should be if it had trusted the IMU, that has smaller errors than the GNSS at such locations, more. For this reason, it is better to rely on just the camera and the angles from the IMU in a forest with dense foliage.

From Table 1, we calculate the absolute error of the fused method to be approximately 80% smaller than the error of the INS, and 87% smaller than the camera's error. However, in the second case, the fused method gave an increase in the absolute error of 2% compared to the INS. Since the camera was so far away from the reference points, a comparison of the absolute error would be meaningless.

In Table 2 we can also see that the difference in relative error between the INS and the fused path is quite noticeable. Especially for the $P_2, P_5, \text{and} P_6$ cases, where the difference in relative error is between 28.4-12.8%, which shows that our fusion method is surprisingly useful considering its simplicity.

But since we have only looked at the closest distance to the reference points, the error can be larger than it seems. The position at the time when we passed the reference point might not be the point that we compare the reference point with, since we do not have an accurate method to check the time when we pass the reference point.

5.6 Future Work

For future work on this product we would like to employ a different fusion method. As discussed in 5.3 the fusion method we would like to implement is a Kalman filter. Both to avoid the discontinuities in our fused trajectory - as discussed in section 5.4 - and get a more accurate result.

Since the camera trajectory was directed using the angles given by the INS, the camera trajectory is subject to an additional drift due to the angles drifting in time. A better choice for directing the camera trajectory would be implementing and using the angles from a digital compass - which would not drift in time.

Another improvement that could be done with the camera trajectory is to use subpixel refinement in the visual odometry, which will most likely improve the results slightly.

Lastly, an obvious addition in the future would be to add a laser scanner. It would be of interest to test our current fusion method with a laser scanner to see

how well the system works. But, most likely, a different fusion method needs to be implemented in order for the data from the laser scanner to give satisfactory results - especially near the discontinuities.

6 References

[1] Geiger, A., Ziegler, J. and Stiller, C., 'Stereoscan: Dense 3D Reconstruction in Real-time', 2011.

[2] NovAtel, *Inertial Explorer User Manual*. [Online]. Available: http://www.novatel.com/assets/Documents/Waypoint/Downloads/Inertial-Explorer-User-Manual-870.pdf.

[3] Lantmäteriet, *SWEPOS RINEX-data*, visited on 2017-01-13. [Online]. Available: https://swepos.lantmateriet.se/tjanster/efterberakning/rinexdata/rinexdata.aspx.

[4] PointGrey, *Getting Started Bumblebee XB3*. [Online]. Available: https://www.ptgrey.com/support/downloads/10133.

[5] PointGrey, *FlyCapture SDK*, visited on 2017-01-13. [Online]. Available: https://www.ptgrey.com/flycapture-sdk.

[6] NovAtel, *SPAN-IGM-S1*. [Online]. Available: http://www.novatel.com/assets/Documents/Papers/SPAN-IGM-S1-PS_2.pdf.

[7] NovAtel, *OEM615*. [Online]. Available: http://www.novatel.com/assets/Documents/Papers/OEM615.pdf.

[8] NovAtel, *GPS-702-GG*. [Online]. Available: http://www.novatel.com/assets/Documents/Manuals/om-20000095.pdf.

[9] Sensonor, *Sensonor AS - High precision MEMS sensors - STIM300*, visited on 2017-01-05. [Online]. Available: http://www.sensonor.com/gyro-products/inertial-measurement-units/stim300.aspx.

[10] Woodman, O. J., *An introduction to inertial navigation*, 2007. [Online]. Available: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf.

[11] Kornhauser, A. L., *Global Navigation Satellite System (GNSS)*, 2006. [Online]. Available: https://www.princeton.edu/~alaink/Orf467F07/GNSS.pdf.

[12] Gabor, M., *GPS Overview*, visited on 2017-01-05. [Online]. Available: http://www.csr.utexas.edu/texas_pwv/midterm/gabor/gps.html.

[13] National Coordination Office for Space-Based Positioning, N. and Timing, *GPS.gov: Control Segment*, visited on 2017-01-05. [Online]. Available: http://www.gps.gov/systems/gps/control/.

[14] Skogsstyrelsen, *Skogskartan*, visited on 2017-01-19. [Online]. Available: https://skogskartan.skogsstyrelsen.se/skogskartan/Default.aspx?startapp=skogligagrunddata.

[15] Sharifi, M., Chen, X. and Pretty, C. G., 'Experimental study on using visual odometry for navigation in outdoor GPS-denied environments', in *Mechatronic and Embedded Systems and Applications (MESA), 2016 12th IEEE/ASME International Conference on*, IEEE, 2016, pp. 1–5.

[16] Kelly, A., 'A 3D state space formulation of a navigation Kalman filter for autonomous vehicles', DTIC Document, Tech. Rep., 1994.

[17] Helmick, D. M., Cheng, Y., Clouse, D. S., Matthies, L. H. and Roumeliotis, S. I., 'Path following using visual odometry for a mars rover in high-slip environments', in *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, IEEE, vol. 2, 2004, pp. 772–789.

[18] Rezaei, S. and Sengupta, R., 'Kalman filter-based integration of DGPS and vehicle sensors for localization', *IEEE Transactions on Control Systems Technology*, vol. 15, no. 6, pp. 1080–1088, 2007.

A Appendix - Program Usage

A.1 The backpack

A.2 Installation

To be able to use this program the right way it is necessary to make the installation of the different softwares in the correct order, otherwise you might get some problem.

1. Install Microsoft Visual studio 2010.
2. Install Triclops 3.4.0.8, which will prompt you to install the correct version of FlyCapture.
3. Install MATLAB (confirmed working on version 2016b).
4. Add the MinGW C++ compiler to MATLAB via add-ons.

A.3 Pre-setup

A.4 Use the program

Synchronize the clock with the computer

- Open the program Novatel Connect.
- A dialog pop-up is showing, close the dialog window.
- Click on the Novatel symbol, open the workspace 'ws'.
- Click on device and syncronize the PC clock. This dialog window will pop up twice but it is just to close them both.
- Close the connection.
- Exit Novatel Connect (under the Novatel symbol).

Collect data

- Make a new folder where you want the pictures to be saved.
- Make sure the python script and the program CaptureTimestamped_bmp both exist in the newly created folder.
- Open the cmd.

- Write: "cd" + the pathway to the folder.
- Run the python file by writing: "python GatherData.py". A command window will open and as long as new dots are printed data is being logged. To end the data logging press ctrl+c.

Process the camera data

- Open MATLAB.
- Change the parameters in the file DEMO.m. Input parameters is: the camera pair that is going to be used, path to the folder where the images exist and the name of the new .txt-file created by the script.
- Run DEMO.m. The command window in MATLAB will display the current image being processed as well as the number of matches and inliers. Note that the number of matches and inliers is between consecutive images in time and not between image pairs.

A.4.1 Logging data from SPAN-IGM-S1

The orientation of the INS as well as the offsets to its antenna is set up as:

SETIMUORIENTATION 6
SETIMUTOANTOFFSETS 0.1 0 0.1 0.005 0 0.005
t The logs that are needed to use Inertial Explorer are the following:

log GLOEPHEMERISB onchanged
log IMUTOANTOFFSETSB onchanged
log RANGECMPB ontime 0.25
log RAWEPHEMB onchanged
log RAWIMUSXB onnew
log VEHICLEBODYROTATIONB onchanged

A.4.2 Base Station Data

The base station data can be received from SWEPOS.
On SWEPOS ftp-server ftp://ftp-sweposdata.lm.se/ in the folders $Rinex-data/Rinex2$ you then chose whether you want daily data in the folder *se_swepos_daily* or hourly from the folder *se_swepos_hourly*. Then you choose the folder for which year you want data from, and then the folder for which day.

The four data files from the base station in Holmsund, which is the closest base station to Umeå, starts with “0hos”. Then there are four characters where

the three first represent the day of the year. The fourth is either a letter and then represents an hour from the day, where a is 00.00 to 01.00, and so on. Or the fourth character is a 0 and then the file is for the whole day. After that there is a dot, two numbers showing which year the data is from and lastly a letter. The last letter can be d, g, n or s.

d: observation file, hatanaka packed
n: navigation file GPS
g: navigation file GLONASS
s: quality information for observation file

A.4.3 Inertial Explorer

When starting a new project in Inertial Explorer use the Project Wizard. First it asks you to name the project under Project Info. Then you get to Remote (Rover) Data and add the .gps file from the INS. The IMU data will also be detected from the file and added to the project. Under Remote (Rover) Antenna profile select the $NOV702GG\_1.03$ antenna profile. After that you want to add base station data from file. Here the file you want to add is the observation file. Inertial Explorer will then say that it is missing a NAV file, and you add the navigation file GPS. Under the Base (Master) Station set datum to ETRS89 and then finish the project.

Under Process you select Process TC (Tightly Coupled) and let Inertial Explorer do its job. This can take a few minutes. When it has finished processing the data, chose the Export Wizard and chose the file name. The profile Tomten has been made to contain all the data that is needed for the Kalman filter. Let datum be the processing datum WGS84, and select UTM to be the grid to use for transformation. Under Geoid Correction, you need to give the program a geoid file. This can be received from NovAtel Support, from the following link: http://www.novatel.com/support/waypoint-support/waypoint-geoids/. After that you come to IMU Epoch Settings, where you choose Export Interval Options. Here you set Binary trajectory Interval to 0.0080 s, and Time Interval to 0.0100s. Then click finish, and you have a ASCII data file.