Semesterthesis – Final Presentation

Welcome everyone to the Final Presentation of my Semesterthesis about Visual-Inertial Odometry. I will show up what is out there and demonstrate where we are towards a Visual-Inertial Localization which is efficient, accurate, robust and lightweight.

-

In the future Tasks like windmill inspection and maintenance will be done by autonomous mobile robots. One key challenge towards this goal is Single Robot Localization. Subsequent tasks like coordinated work, dense reconstruction of a surface or real manipulation rely on an accurate localization – globally, the robot has to fly to the windmill, and locally for the inspection itself.

Visual-Inertial Odometry is one framework takling Localization. By combining the rich structure information of a camera with the short-time accuracy of an Inertial Measurement Unit an accurate Robot Localization can be achieved.

Advantages against other Localization Approaches are the reliance on lightweight and cheap sensors. The use of passive sensors with a small power consumption; and the reliance on solely onboard sensors averting the need of an external localization system.

-

I would like to present 2 main topics today. In the first part I will compare two Visual-Inertial Odometry Implementations by demonstrating their working principles, showing differences and demonstrating their accuracy.
In the second part I will show an analysis regarding the question if Visual-Inertial Odometries can work with non-timesynchronized Hardware.

-

Visual-Inertial Odometry is part of the larger field of Visual Odometry. The Visual-Inertial Odometries can be divided into 2 principally different approaches – Filtering-based and Keyframe-based.

Most Filtering-based Algorithms are based on an extended kalman filter and estimate the robot pose in 2 steps: Firstly they propagate the robot pose between frames based on the IMU measurements. Secondly, as soon as a new image is available, the estimate is updated based on the observation. The camera-observation of a past image is transferred into the current one with the propagation, so via the prior belief of the filter. But a past estimate will never be corrected based on the current observation.

On the other hand the keyframe-based approach estimates the robot pose by performing a nonlinear optimization based on a large number of landmark observations taken from a set of past keyframes and the current frame. Like that poses of the past are corrected based on the current observation as well.

Because of these inhearent differences the Keyframe-based approach often shows better accuracy as it tends to drift less over time.
On the other hand the filtering-based approach is in general less computational expensive while performing, especially locally, surprisingly accurate.

For this work I focused on Rovio and Okvis – two promising Implementations of the two approaches. Both have been developed here at ETH.

-

Rovio stands for Robust Visual-Inertial Odometry and it is using a direct Extended Kalman Filter Approach. What you can see on the slide is a Visualization of Rovio during operation. On the left side we see the current image Rovio is working with. Rovio is using a small number of features – in this moment around 20 – which are reprojected into the image.

Rovio detects new features with a fast corner detector and is working with Multilevel Patch Features. It includes new features into the filter state under certain conditions – they must not lie near an already tracked feature, their Intensity patches has to show reasonable intensity gradients and there needs to be free space in the filter state.

Rovio estimates the Robots Position, Velocity, Attittude, the IMU Biases, the Extrinsics between IMU and Camera and the features themselves. The features are represented in the filter state by their bearing vector wrt the camera frame and their distance to the camera.

On the right side we see a Visualization of Rovios estimates – In the center we see an inertial coordinate frame and the body frame (which is not yet well visible). The white dots represent the filters belief of the Features and the green line denote the 2 sigma bound of the feature uncertainty.

At the initialization all features are initialized with a high uncertainty

PLAY

As we can see, as soon as the camera is moved the uncertainty associated with the features is decreased.
One can observe, that Rovio often is working with edge features, which are not well suited for tracking as they can move along a line.

-

Okvis stands for Optimal Keyframe-based Visual-Inertial SLAM. In the Visualization we can see on top the latest keyframe and on the bottom the current frame.

Okvis detects new features with a multiscale Harris-Corner detector and describes them as Brisk features. In the image, the extracted features are are shown as circles, the green lines highlight matching pairs.

Okvis is estimating the robot pose by performing a nonlinear optimization minimizing an error function containing reprojection error terms and IMU error terms. Like that it fullfills an optimization of all keyframe poses, the newest frame poses, speed and imu-bias terms and all landmark positions.

In order not to grow unbounded, Okvis performs a Marginalization of old keyframes, their landmarks and Speed/Bias Terms.

On the right side of the image you can see a Visualization of the landmarks. All landmarks currently used in the optimization are highlighted in green.

PLAY

One remark regarding Okvis: We have not been able to run it on our machines until today. The OKVIS results I present have been generated some time ago by Stefan Leutenegger and Simon Lynen.

-

I want to compare the two presented algorithms on a dataset Simon Lynen collected for evaluating Okvis. The data was captured with the VI-Sensor which is equipped with two global-shutter monochrome cameras with 120 degree wide lenses and a high-quality MEMS-IMU. I will compare the algorithms running on a single camera and the IMU. The data contains a long trajectory captured indoors with Vicon ground truth. Therefore approximately 100 handhald loops have been captured.

The 2D overview shows the ground truth trajectory.

Start and Endpoint are lying on the down left corner, the camera was facing in direction of movement during the whole set. Walking was quite fast with 2 m/s and rather rotational rates up to 3 rad/s.

Here you can see a visual impression of ROVIO's tracking performance – in the beginning it is tacking the circle quite well but over time we can observe it drifts.

If we look at this overview plot of Okvis we see that it is also drifting but it stays nearer to the ground truth. For all the upcoming slides I will show ROVIO in RED and OKVIS in blue.

-

To measure performance it is useful to differentiate between Global and Local Accuracy. The most common metric to describe the Global Accuracy is the absolute translation error. It describes simply the norm between the ground truth and estimated position after a certain travelled distance.

I like to show errors with boxplots. After 1000 meter we can see that Okvis has a median absolute translation error of below 1 meter, while Rovio shows a median absolute translation error of 2.5 meter.
If we look at the absolute orientation error which denotes the absolute angle between ground truth and estimated attitude after a certain travelled distance we see that Okvis shows with 4 degree error 10 times better accuracy than Rovio with 45 degrees.

-

To understand what is going on it makes sense to look once into the global errors on the separated axes. For roll and pitch, the rotations around the optical axis and the pitching we see for both algorithms bounded errors with Okvis being 2 to 3 times more accurate. The reason why roll and pitch errors are bounded lies in the fact that the gravity vector is observable with the IMU and therefore errors axes in the horizontal plane do not drift.

The large difference shows up in the yaw error – the error around the world-z axis. Here we can see where the major part of the global orientation error from the previous part is coming from.

The Errors in world-x and world-y direction are mainly driven by the yaw error and would look completely different for another dataset – if the camera would be moved in a straight line for example a yaw error of 45 degrees would sum up to huge world-x and world-y errors.

The world-z error instead gives again valid information and we can see that world-z drift of Rovio after 1000 meter is 3 times larger than of Okvis.

-

The most important information regarding Global Accuracy are therefore that, for this dataset, Rovio shows a yaw drift which is 10 times higher over 1000 meter and a world-z drift, which is 3 times higher over 1000 meter.

-

Beside of the Global Accuracy it is of great interest how well tracking is working on short distances. The relative translation error compares the Travelled Distance between two frames. After a travelled ground truth distance of 1 meter, the difference of the travelled distance by the visual inertial Odometry approach will be a bit larger or smaller.

We can see that the relative translation error is a bit higher for Rovio, but in the same order of magnitude for both algorithms. For both algorithms we see an error of approximately 2 cm in average after 1 meter travelled distance.

The second metric to describe the local accuracy is the relative orientation error. It compares the change in attitude given by the ground truth with the change of attitude given by the visual inertial Odometry. Here we see similar performance for both algorithms. Both algorithms show a mean error of 0.6 degrees after 1 meter travelled distance.

-

The following table summarizes the results of comparing the two algorithms on this dataset. Regarding Global Accuracy we see a clear gap between Rovio and Okvis. The absolute yaw error is 10 times higher. The absolute world-z error is 3 times higher.
Locally we saw that both algorithms show similar results.

Regarding Computational Complexity I analysed Rovio on my machine. The processing time per frame is dependent on the maximum number of features you use in the state, for the setting with 25 features which I used for all results, it is about 10 ms. For Okvis we do not have a value to compare but we know that it is computationally more demanding but able to run in real time with a framerate of 20 Hz.

Beside of the presented “hard” numbers there is an additional argument for Rovio: It is working robustly for many people in here – live with the visensor, in simulation and on other different datasets I was working on.

Regarding Global Accuracy Okvis the clear winner.

I would now like to get over to the second topic, where I want to show you an analysis I have done on the way towards non-timesynchronized hardware. The results I presented so far have been based on data collected with the VI Sensor. The Vi Sensor is hardwarewise time-synchronized which means that the IMU measurements are triggering the camera image capture. The image capture is even exposure compensated which means that, depending on the exposure time the triggering is shifted in time.

An important question when we want to apply Visual-Inertial Odometry generically is, if an algorithm is also able to work with a sensor setup that does not perform such a hardware time-synchronization.

-

As a first step towards this question I did an analysis, where I took data collected with the VI-Sensor and added an artificial bias on the IMU-Timestamp.

You can see in the plot the Translation and Orientation Errors averaged over the whole timesequence as a function of the artificially added IMU timestamp bias. What we observed with this analysis was, that, for the given data, Rovio is able to do quite a good job until 50 ms, and eventually starts diverging at 100 ms.

A second result out of this analysis was, that we can get satisfying results at even higher Biases by fixing the Camera-IMU extrinsics. This means that normally Rovio is constantly estimating the camera-IMU extrinsics over time. If we initialize Rovio with a good set of extrinsics and reduce this degree of freedom in the filter state, higher artificial timestamp biases can also be tracked.

-

This analysis encouraged us to collect a real dataset to compare Rovio once working with the VI sensor data and once with a non time-synchronized setup.

I attached a Bluefox camera with the same characteristics as the VI-Sensor camera to the VI-Sensor, making sure that they capture the same field of view.

With that I collected two datasets within the Vicon room and analysed Rovio performing once with the VI sensor data and once with the data from the Bluefox camera and the IMU data from the VI-Sensor. For both sets I collected a set over a distance of 60 meters which was 3 loops in the vicon room.

-

Here you can see a short impression on the “slow dataset”

PLAY

I collected it in the LEO vicon room, where a lot of landmarks are in the field of view. The camera stream of the VI sensor camera and of the Bluefox camera look very similar. Running now Rovio once with the visensor data and once with the non-timesynchronized setup got the following result.

The most important result of the analysis is, that the non-timesynchronized setup does not break Rovio and it still does a good job in estimation.

The second and not too surprising result was, that the tracking performance of Rovio working with the VI-Sensor was more accurate.

-

In the following I tried to go towards the dynamic limits and collected the following shaky dataset.

PLAY

Running now Rovio on the VI sensor data still resulted in non-diverging tracking, while Rovio running on the non-timesynchronized data was diverging. We reached a limit for Rovio working with non-timesynchronized hardware.

The last step I tried now was to fix the camera-imu extrinsics. By reducing this degree of freedom in the filter state Rovio is not diverging anymore. It is an important result that fixing the extrinsics can improve performance when applying Rovio on non-timesynchronized hardware.

-
DURCHATMEN

Summerizing the presented evaluations we showed, that

1) Okvis showed better global accuracy
2) Okvis and Rovio showed similar local accuracy
   and
3) Rovio is able to run on a non-hardware-timesynchronized Sensorsystem

This brings me to the outlook after this Semesterthesis.

To tackle the global drift of Rovio it has to be combined with a “Backend”, which is a secondary module taking the output of Rovio and performing global optimization. This backend could than feedback its results to Rovio like a GPS input.

One improvement that was lately already included into Rovio is the ability to include additional sensorinformation like GPS or Magnetometer signals. I belief, that by only including the information of a magnetometer yaw drift could be drastically reduced.

A third upcoming analysis will be to run Rovio on an embedded system with lower-quality IMU and non-timesynchronized hardware. This will take place in the upcoming week.

And as a last point, the presented results encourage even more to run Okvis and evaluate it’s performance on complementary datasets and on the data I collected for my analysis towards non-timesynchronized Hardware.

-

Coming back to the Task of Windmill inspection and maintenance I would like to highlight again, that both analyses Visual-Inertial Odometries show very promising results. I strongly belief, that Visual-Inertial Odometry is the key towards efficient, accurate, robust and lightweight Localization.