# Examination of Methods for Error-Tolerant Data Association between Independent Relative Pose Graphs during Multi-Agent SLAM

Julian Schmiemann[1], Hannes Harms[1], Jan Schattenberg[1], Ludger Frerichs[1]
[1]*Institute of Mobile Machines and Commercial Vehicles, TU Braunschweig, Germany*
*j.schmiemann@tu-braunschweig.de*

Keywords: 3D-SLAM, cooperative mapping, unknown initial relative poses

Abstract: In this paper we introduce work done within the joint development project "**ANK**omm**E**n". We want to present techniques, which we use to enable cooperative 3D-SLAM with ground based vehicles. The presented methods are – in this special case – designed for, but not limited to, Velodyne VLP-16 sensors. Due to the obtained trajectories, sensor readings from other environmental sensors carried along can easily be mapped, too.

## 1 INTRODUCTION

Highly automated machines - or even clusters of machines - designed for exploration tasks are capable of contributing to enhance efficiency for a broad variety of applications - for example during search and rescue missions. Due to this variety of different applications and environmental conditions, defining one single exploration task, without losing generality, is kind of complex. Hence designing a system, which fulfills all requirements is even more complex. More or less different exploration tasks need different types of environmental sensors to capture relevant information. For such a purpose, a distributed system approach seems reasonable. An approach which includes a cluster of dissimilar machines even can enhance functionality. Aerial vehicles are suitable to carry out tasks like spotting interesting areas in spacious terrain. Once such spots are located, ground based vehicles may be the best choice to examine those particular spots of limited spatial expansion in detail. The Institute of Mobile Machines and Commercial Vehicles, the Institute of Flight Guidance – both Technische Universität Braunschweig – and AirRobot GmbH & Co. KG, utilized within the joint development project "**ANK**omm**E**n" (german acronym for: "Automated Navigation and Communication for Exploration") - funded by the German Federal Ministry of Economic Affairs and Energy administrated by the Space Administration of the DLR - to develop such a system consisting of three UAVs and two UGVs equipped with environmental sensors like LiDAR, stereo vision, high-resolution RGB- and IR-cameras.

Knowledge of the environment is a crucial factor not only for rescue forces whereas catastrophic scenarios, but for highly automated machines to operate as well. Due to the absence of an external human operator, the machine itself has to be capable of classifying traversable and non-traversable areas for example or detecting obstacles in its operating field. Therefore autonomous mobile machines operating in a dynamic environment are equipped with environmental sensors like RADAR, LIDAR and mono / stereo-vision. In unknown environments information gathered by previous sensor readings may be useful, too. Therefore collecting consecutive readings from those types of sensors and merging them into a single common frame of reference – hereafter called mapping – can be useful. The machine can use those information for path planning algorithms when revisiting a place for example. In addition a system capable of collecting readings from environmental sensors in such manner can be used by human to run autonomous exploration tasks as mentioned above. The reliable and accurate self-localization is the elementary pre-requisite for building such maps. State-of-the-art localization solutions mostly are based on GNSS-receivers. Those are performing convincingly in the open country, but failing when the receiver isn't in direct line of sight to a sufficient amount of satellites. By taking environmental sensor readings into account, the robustness and accuracy of self-localization can be improved in built-up urban environments for example

or even can be enabled in indoor applications. Concurrent self-localizing while simultaneously mapping the environment is often referred to the SLAM problem. Due to the last decades the interest and effort in this field of research grows steady.

Cooperative mapping of the environment always deals with the problem of finding correspondences between sensor readings from different agents at potential different times. In dynamic environments and with mobile machines - without the loss of generality - this will lead to the problem of associating data between independent robot's trajectories. This is the fundamental basis to find a distinct transformation which allows to convict all measurements to a single common frame of reference.

To give the reader an overview about the past work made on the field of multi-agent pose graph SLAM, the following section will briefly introduce the most important concepts and techniques which are related to the work made in this paper. Afterwards methods developed within the context of "**ANK**omm**E**n" are presented. Therefore section 3 will deal with methods for constructing relative pose graphs, before section 4 will show up how data association between independent trajectories can be archived in an error-tolerant fashion. Afterwards experiments are shown to validate the described methods.

## 2 RELATED WORK

Due to the presence of massive noise while measuring with environmental sensors, a probabilistic approach for modelling the SLAM problem is standard in literature. The robot's trajectory consists of a sequence of random variables $x_{1:T} = \{x_1, \dots, x_T\}$ where $x_i$ represents the robot's pose at a given time $i$. While the robot is driving around it is collecting control inputs $u_{1:T} = \{u_1, \dots, u_T\}$, which for example can be acquired using wheel encoders or other odometry measuring devices. Assuming the markov property holds, one can model those actions with $P(x_i|x_{i-1}, u_i)$. It describes the probability that the robot is at exact the spot $x_i$, because it is affected by a control input $u_i$, starting from pose $x_{i-1}$ and typically is modeled with Gaussians as presented in eq. 1.

$$x_i = f_i(x_{i-1}, u_i) + w_i \tag{1}$$

Where $f_i$ represents the process model and $w_i$ is normally distributed zero-mean process noise with covariance $\Lambda_i$. Observations of the environment on the other hand are collected into the measurement vector $z_{1:T} = \{z_1, \dots, z_T\}$ and are modeled with the measurement model $P(z_k|x_{i_k}, l_{j_k})$ consisting of Gaussians as well (eq. 2).

$$z_k = h_k(x_{i_k}, l_{j_k}) + v_k \tag{2}$$

Here $h_k$ defines the measurement function and $v_k$ is the normal distributed zero-mean measurement noise with covariance $\sum_k$.

Goal of SLAM algorithms now shall be to estimate the posterior probability of the robot's trajectory $x_{1:T}$ and the map $m$.

$$p(x_{1:T}, m \mid z_{1:t}, u_{1:T}, x_0) \tag{3}$$

Firstly introduced formulations of the SLAM problem were based on filtering techniques, while newer methods mostly are based on smoothing approaches. The filtering formulation only contains the current robot pose and map. Therefore filtering methods are often referred to online state estimation techniques. Famous filtering techniques for example are Kalman Filters (Smith 1990), Particle Filters (Hähnel 2003, Grisetti 2007) and Information Filters (Eustice 2005, Thrun 2004).

Smoothing approaches on the other hand are formulated to estimate the whole robot trajectory from the full set of measurements in the mean of least squares error minimization. In general there are different ways to model the Smoothing approach of the SLAM problem. One can interpret equation 3 according to a Bayesian Belief Net. The upper section of Figure 1 illustrates such a Belief Net. While the robot travels along a trajectory consisting of 3 poses,

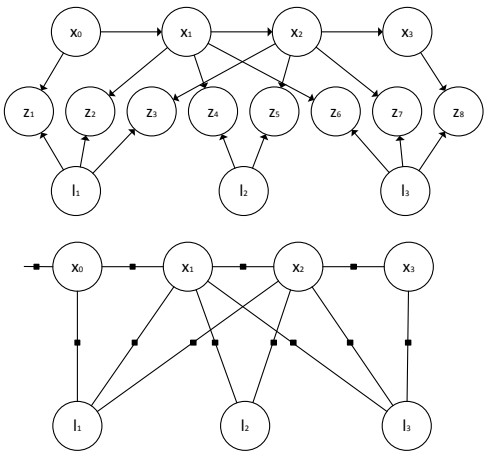


*Figure 1: Different represntations for the SLAM problem. Top : According to Bayesian Belief Net. Bottom : According to Factor Graph*

it takes for instance 8 measurements of 3 different landmarks. The lower section is reflecting the same situation as a Factor Graph representation, which will be discussed later on.

In thoughts of the former representation and with the markov property in mind – which implies that the robot's state depends only on the previous (but not before the previous) state – one can postulate the joint probability as described in eq. 4. The initial state $P(x_0)$ therefore can be chosen arbitrary.

$$P(X, L, Z) = P(x_0) \prod_{i=1}^{M} P(x_i|x_{i-1}, u_i) \prod_{k=1}^{K} P(z_k|x_{i_k}, l_{j_k}) \quad (4)$$

The most probable solution for the set of unknowns given all measurements can then be obtained by least mean squares error minimization. Therefore the maximum a posteriori estimate (eq. 5) for the whole trajectory $X \triangleq \{x_i\}$ and the map $L \triangleq \{l_j\}$ given the measurements $Z \triangleq \{z_k\}$ and control inputs $U \triangleq \{u_i\}$ is formulated.

$$\begin{aligned} \Theta^* \triangleq argmax_\Theta P(X, L|Z) \\ = argmax_\Theta P(X, L, Z) \\ = argmin_\Theta \\ -\log P(X, L, Z) \end{aligned} \quad (5)$$

Solving this leads to a non-linear least-squares problem. Suitable for solving such problems are for example Gauß-Newton or Levenberg-Marquardt algorithms.

Lu et al. (1997) introduced a novel graph notation for the smoothing formulation of the SLAM problem. Those approaches have been significantly improved over the time (Olson 2006). Due to their nature of incorporating each and every measurement this group of approaches is often referred to the full SLAM formulation. A superb introduction to graph based SLAM is given by Grisetti et al. (2010).

Dellaert (2005) introduced a novel approach to the SLAM domain, which had already proven successfully that it is capable of handling large problems with lots of unknowns in the domain of photogrammetry – there this technique is related to bundle adjustment - and in the domain of computer vision - where it is referred to the structure from motion problem. This novel approach to solve the full SLAM problem is called Square Root Smoothing and Mapping (Square Root SAM). It is based on factorization of the information matrix. A detailed description is given in (Dellaert, 2006). But for a better understanding of this paper, the fundamental ideas of this techniques shall be discussed briefly.

The representation with Factor Graphs has some major practical advantages, because it represents the underlying optimization process in more detail. The evidence that the measures $z_k$ are known in fact shall be represented. Eliminating them as variables and introducing them as parameters of the joint probability factors over the actual unknowns yields to the factor graph representation (cf. Figure 1).

Apart from classifying the underlying mathematical concept, one can subdivide multi agent SLAM methods with respect to processed data. Most effort in research has been made in the subject of landmark based methods. Fenwick et al. (2002) firstly extended single agent approaches to the multi agent domain. Most methods developed need initialization with known relative poses (Howard, 2006) (Anderson, 2008). A different often used technique is to gather direct relative pose measurements between robots (Howard, 2004). Therefore a common technique is to tag robots individually (Zhou, 2006). Franchi et al. (2013) introduced a novel method that is capable of handling direct relative pose measurements without the need of tagging them uniquely. Another often used technique to find transformations between multiple agents is to estimate the relative pose indirectly by comparing sets of landmarks. This is mostly done using any kind of RANSAC fashion algorithm (Montijano, 2011) (Cunningham, 2012). The major advantage of indirect methods is the fact that no physical rendezvous are required. Unlike to direct relative pose measurements, indirect methods are capable of inferring positional information from revisiting a spot another agent visited before – a time independent generalization of physical rendezvous – hereafter called encounters.

The subject of multi agent pose graph SLAM has not been investigated in similar depth in the past, but recently few very interesting approaches have been published. Kim et al. (2010) introduced a method which handles multi agent pose graph SLAM in novel fashion. They are keeping individual trajectories for every agent and introducing the relative poses between those trajectories as unknowns to the optimization process.

Furthermore it is important to notice, that the majority of presented methods in literature are treating data association as given. In reality this assumption is the first to fall. Erroneous data association in graph based SLAM usually leads to false positive constraints – i.e. observations from physically unequal objects will be constrained to be

equal. Introducing false positive constraints to the SLAM graph may lead to instant divergence of both - the trajectory and map. To avoid wrong positive entries, one is well-advised to introduce constraints in a strictly conservative fashion. In our work we are combining the strength of both methods. The robustness of a pose graph approach is combined with landmark based methods for place recognition purposes. Those in fact are useful to enable loop closing (or finding encounters in general) functionality in the total absence of any external position information (like GNSS). Our flexible hybrid approach is capable of associating data on the fly, when needed. Therefore the risk of introducing false positives due to errors in data association is minimized.

# 3 CONSTRUCTING RELATIVE POSE GRAPHS

A relative pose graph is constructed by poses and constraints connecting them. To increase accuracy – especially when building maps over large trajectories including lot of poses – one can introduce constrains not only between consecutive poses, but between arbitrary poses. This is mostly done using loop closing techniques.

But factors do not necessarily need two poses. It is even possible to introduce factors, which influence only a single unknown – therefore called Single Factors.

## 3.1 Factors

Factors in the context of graph SLAM represent measurements (cf. eq. 4). One can distinguish factors by the number of unknowns affected by precisely that factor. Although it is conceivable (and useful in some cases as to be presented later on) the maximum amount of affected unknowns by a single factor within a single trajectory is limited to two. Therefore in the SLAM domain there are Single Factors – affecting only a single unknown – and Pairwise Factors – affecting a pair of unknowns.

Single Factors can be obtained by sensor systems which measure "absolute" position. Absolute in this context means in reference to a global common reference frame (like WGS84). In the mobile machines community GNSS is the most common way to acquire such absolute position measurements. Pairwise factors on the other hand can be generated in various ways.

Therefore the next section will give a brief introduction to the topic of generating Pairwise Factors.

## 3.2 Pairwise Factors

Introducing a Pairwise Factor in this context means to introduce a constraint between two poses. Such constraints can be obtained using odometry sensors for example. But Pairwise Factors can be generated with respect to readings from environmental sensor, too. Therefore consecutive sensor readings are matched. The resulting transformation obtained by suitable matching algorithms – like the famous Iterative Closest Point (ICP) algorithm – implies the movement of the acquiring sensor and therefore can be used to connect consecutive poses.

In the next sections a few commonly used techniques to generate Pairwise Factors are briefly presented.

### 3.2.1 Motion Models

A motion model can predict motion in a pure mathematical way. This method is only useful, if in fact no sensor to measure motion is available or for guessing initial poses for more complex procedures (like guessing initial poses for ICP to avoid convergence to local minima). Such a model can be interpreted as a "continuous" source for Pairwise Factors. Experiments have shown that introducing to many factors to the factor graph does not lead to a higher precision in localization nor mapping. Because of this fact, an estimate from motion models should only be introduced – if used - when new sensor readings from low frequency environmental sensors arrive.

Different motion models are suitable for different situations. The most popular motion models are the zero motion – which implies the absence of any motion - and the constant motion model.

### 3.2.2 Odometry

Most odometry sensors can be interpreted as "continuous" sources for factors as well, because the output frequency of such sensors in general is many times – up to few magnitudes - larger than frequencies of most environmental sensors. Therefore, accumulating readings from odometry sensors until a new reading from an environmental sensor arrives, seems reasonable, in this case, too.

### 3.2.3 ICP Pose Factors

As mentioned earlier, ICP algorithms are perfect suitable for generating pairwise factors. In the past decades various papers about new ICP variants had been published. Therefore finding a suitable variant is not trivial. A superb introduction to this topic is given by Pomerleau et al. (2013). Figure 2 shows a simple ICP pipeline.

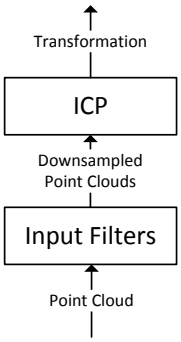


*Figure 2: The ICP pipeline*

First the incoming point cloud is sampled down. Due to the fact that computation effort raises exponential to the amount of data points within the cloud, down sampling is a common technique to accelerate computation. The most common filtering techniques use Voxel Grid Filters. This technique subdivides the three dimensional space in equidistant grid cells, i.e. the voxels (volumetric pixels). All data points within a voxel are then accumulated. Another – and way easier to handle and implement - often used technique is random sampling.

After filtering, the point cloud is given to the ICP algorithm. If there had been a previous cloud the first step of each iteration will be to find corresponding data points in both clouds. Once correspondences have been established the incoming cloud is transformed with respect to minimize the sum of a squared error metric. After transforming the iteration step is done and if no abort criterion is satisfied, a next step is triggered. Commonly used abort criterions include minimal translation and / or rotation per step, maximum amount of iterations or a maximum summed squared distance from source to target cloud. After convergence, the transformation is introduced to the factor graph as a factor between consecutive (or in the mean of loop closure between arbitrary) poses at which the measurements have been acquired. Commonly used error-metrics are point-to-point, point-to-plane and plane-to-plane.

Since probabilities are the decisive aspect in modelling the SLAM problem in a factor graph fashion as proposed in this paper, the factors have to incorporate uncertainties. Errors and the related uncertainties in scan matching arise from different type of sources. Censi (2007) discusses the most important errors and presents a method for effective estimation of covariance matrices in ICP. Those source for errors in estimating transformations with ICP are: wrong convergence, under-constrained situations – those appear when the environment of the robot does not dispose enough information to estimate the robot's pose completely - and sensor noise.

### 3.2.4 ICP-Sequence Pose Factors

One major drawback using scan to scan ICP matching algorithms to introduce factors in a Factor Graph is the unbound error which arises from drift while matching the current scan only against the previous. Errors accumulate over time and the uncertainty in estimating the current pose growth unbound.

To address this shortcoming, one can match the current scan with a map (Pomerleau, 2011). Figure 3 shows a simple ICP-Sequence pipeline.

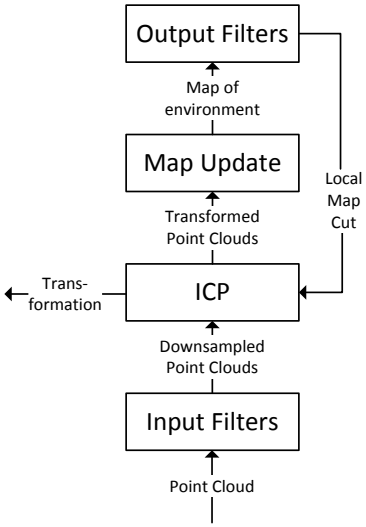


*Figure 3: The ICP-Sequence pipeline*

The difference to the plain ICP algorithm is, that the reference, i.e. the point cloud the reading will be matched against, does not only contain the previous scan but a whole map of the local environment. A crucial step is the map update step. One must not simply add the current transformed point cloud to the map, instead one has to search for the nearest

neighbors of all points from the reading point cloud in the reference point cloud. Once the neighbors are determined a new point is added only if the distance to its nearest neighbor in the map point cloud exceeds a certain threshold. In our experiments we used 0.1 meters for this threshold. This technique avoids unbound growth of the map and reduces errors from erroneous matching results of the ICP algorithm.

### 3.3 Anchor Factors

Based on the work of Kim et al. (2010) one can formulate the multi robot mapping problem in a single common frame of reference, while keeping single trajectories. Therefore every trajectory is based on an anchor, which explicitly represents the transformation to the common multi robot frame of reference. To be able to constraint independent relative pose graphs they are introducing factors affecting four unknowns. Besides both poses the encounter has been observed by, the relative transformation (the anchor) between both trajectories is affected, too.

This means the relative pose between independent trajectories is in fact part of the underlying optimization procedure and therefore it may vary over time, when new constraints are introduced to the graph.

## 4 DATA ASSOCIATION BETWEEN INDEPENDENT POSE GRAPHS

The hardest problem in the domain of multi agent SLAM may be the problem of correct data association between independent trajectories. In the total absence of any external "absolute" position information, i.e. in the absence of GNSS signals, there is no trivial method for searching correspondences in independent trajectories. Running SLAM with three dimensional environmental sensors and browsing whole long-term trajectories for correspondences may easily thrash out any processing unit. The idea of our approach is to reduce the complexity in comparing independent trajectories by using general purpose features. Key points should be selected wisely, to compensate the loss of information which gets together with dropping most of the data points. Extracting features from 2D images is a widely spread technique in computer vision for example. Nowadays few techniques have been adopted for 3D point clouds, but they haven't been developed to such an extent yet (Rusu, 2011) (Alexandre, 2012). Therefore in this paper we are using methods from two dimensional computer vision to improve performance.

Those features are detected in real time. Therefore, while acquiring sensor readings, a higher amount of processing power is consumed, but once computed they can be used to compare trajectories in a small fraction of time compared to handling the full set of data points.

### 4.1 General purpose features

Li et al. (2010) introduced a novel approach to extract general-purpose features from LIDAR data. They have shown that their approach produces features, which are highly recognizable from a wide range of varying viewpoints. A slightly adopted version of that algorithm is used in this paper to generate features from point clouds acquired by LIDAR.

A pipeline to extract such features from sensor readings is shown in Figure 4. The incoming point cloud is projected to the ground plane with respect to

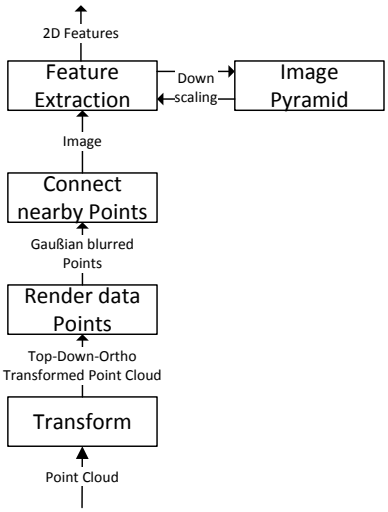


*Figure 4: Extracting general purpose features*

the robot's roll and pitch. Therefore it is required that the robot is able to measure those unknowns with sufficient accuracy. Those measurements can arise from extern sensors like gyroscopes, but also can be determined indirectly by methods like ICP (cf. section 3.2.3 and 3.2.4). Each point is rendered to the ground plane by a two dimensional Gaussian kernel with varying kernel width to represent noise in sensor readings. Therefore the variance $\sigma^2$ which in fact reflects the kernel width, i.e. the affected amount of pixels, is obtained by the measuring uncertainty in distance $\sigma_s^2$, the distance $r$ itself and the horizontal angular resolution of the LIDAR $\Delta\Theta$ in a fashion like eq. 6 represents.

$$\sigma^2 \approx \sigma_s^2 + (r \sin(\Delta\theta))^2 \quad (6)$$

Nearby points – which are considered to belong to the same physical object - are then connected with a line, which is blurred similar to eq. 6. The gray value of each pixel then depends on the height measured in exact that cell. Therefore cells containing only a single measurement point aren't rendered at all. This procedure produces images which contain in particular physical objects which are extended in height and therefore are suitable to match against.

A Kanade-Tomasi corner detection algorithm (Tomasi, 1991) is then applied to the picture. Afterwards the image is scaled down by a power-of-2 image pyramid. This technique is widely spread in computer vision and is used to detect features at different scales. Unlike readings from two dimensional mono-vision cameras – those typically used in computer vision – readings from LIDAR are scale invariant, therefore using an image pyramid in this fashion means to detect features at different physical scales. Therefore this method is capable of extracting features from both – physically large and small objects. In our experiments the pyramid level is set to 4. At lowest resolution a picture has only 1 / 256 of original pixel size. This procedure is quite comparable to extracting commonly known features like SIFT or SURF. A typical output can be found in Figure 5. The scene is rendered from corner of a building.

*Figure 5: Typical output of general feature detection algorithm. Feature responses are drawn via covariance ellipses.*

Note that the radii of the covariance ellipses are scaled with a constant factor to make them visible in the picture. To generate more stable and robust features, one could reject features with respect to uncertainty in position defined by the covariance for example.

## 4.2 Finding Encounters

As described earlier finding encounters with the lack of initial position information can be a tough problem, but with those general purpose features described in section 4.1 one can use brute force methods to simply compare each pose of each trajectory with every pose on all other trajectories. The number of features generated per pose is about of 3 magnitudes lower than the amount of data points. A typical point cloud rendered with approximately 30.000 points contains less than 30 features. Nevertheless a direct comparison of feature sets with a RANSAC approach for example, would require to check each feature with any feature from the other set.

To compute the similarity of sets of features we use an approach similar to that introduced by Cunningham et al. (2010). They are not matching sets of features in a direct fashion, but they are using the Delaunay triangulation to generate unique and invariant features. The Delaunay triangulation is a well-studied technique and is commonly used in computer graphics for example. A set of triangles $T(P)$ in a plane are estimated so that none of the two-dimensional points $p_i$ is inside the circumcircle of any triangle $t_j(P)$. For each triangle with edge length $a_j, b_j, c_j$ they compute the pair of features $\{f_n^j\}$. Former is defined as the circumference (eq. 7) of the triangle.

$$f_1^j = a_j + b_j + c_j \quad (7)$$

And second is the area (eq. 8) of that triangle.

$$f_2^j = \sqrt{s(s-a_j)(s-b_j)(s-c_j)}, s = \frac{1}{2} f_1^j \quad (8)$$

A set of correspondences $C(T_i, T_j)$ can then be estimated. Therefore the sets of triangles are browsed for pairs of triangles, which have an error metric $S$ less than a certain threshold $\tau$.

$$C(T_i, T_j) = \{(t_m, t_n) \mid t_i \in T_i, t_n \in T_j, S(t_m, t_n) < \tau\} \quad (9)$$

The error metric is presented in eq. 10.

$$S(t_m, t_n) = e^{\left((f_m^1 - f_n^1)^2\right)} e^{\left((f_m^2 - f_n^2)^2\right)} \quad (10)$$

To estimate outliers in the set of correspondences $C$, which occur due to ambiguities in this geometric approach, a simple RANSAC

algorithm is used. Therefore a transformation model – consisting of a two dimensional rotation and translation - is built by randomly sampled two correspondences. Afterwards other randomly sampled correspondences are checked against exact that transformation model. Therefore points from the source frame are transformed to the target frame with respect to the obtained transformation model. If the distance of the transformed source point to the target point is within certain bounds the correspondence in question is supposed to be an inlier. The likelihood for an encounter can then be postulated to the ratio of inliers to the total amount of potential correspondences.

The processing pipeline to find encounters between independent relative pose graphs to finally estimate the relative pose between them is presented in Figure 6.

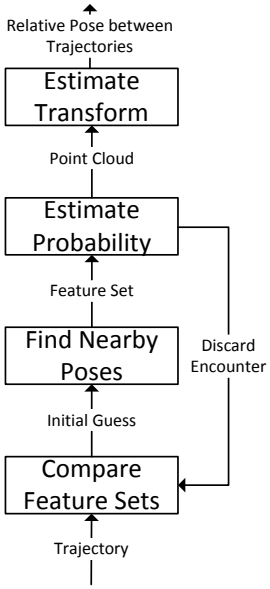


*Figure 6: Pipeline to estimate relative pose between Independent relative pose graphs.*

Once a promising initial guess for an encounter is made, we are estimating the probability for that guess to hold. Therefore we are searching for further encounters between both trajectories. This is done using loop closing techniques. A kD-Tree is set up to find nearby poses. Once nearby poses are determined the probability for that specific encounter is extended by the probability that those both new poses describing the same physical spot as well. This is also done using the two dimensional features, but with taking the centroids of both corresponding point clouds into account. If the centroids are that far away or even closer, it is highly probable, that both point clouds are representing the same physical scene. Afterwards a transformation between those poses can be estimated as well. A joint probability for the whole encounter can then be derived from the consistency of all estimated relative transforms between the trajectories in question. It is more likely that a probable transform was found, when all transforms are similar.

If the joint probability for an encounter exceeds a certain threshold, the transforms between all nearby poses from different trajectories are then constrained by anchor factors.

## 5 RESULTS

Finally we want to present some experiments we did to validate the presented techniques. Therefore a tiny setup with two independent trajectories was chosen. It may be noted that the aim of this experiment was only to show the basic functionality of this techniques. Further analysis and long-term multi robot exploration tasks will be investigated in future and lay way beyond the scope of this paper.

We ran this experiments with a SUMMIT XL robot equipped with a Velodyne VLP-16. To simulate two robots, a trajectory was cut in two pieces. While acquiring the sensor readings, no other positional information has been merged. The localization depends only on the results of the ICP Sequence method described in section 3.2.4. The relative pose estimation was performed as a batch operation in post processing. An online estimation is under development at the moment.

Figure 7 illustrates a trajectory of a robot, which just left a factory building. To make the trajectory and

*Figure 7: Robot trajectory (green) and map of the environment rendered to the ground plane.*

map interpretable, it was rendered to the ground plane. The trajectory (green) consists of 60 poses with one reading from the Velodyne sensor per pose. Therefore we have a total amount of approximately 2 million data points within that trajectory.

Figure 8 shows a magnified section of the same image. There the trajectory of the robot is drawn in green, too. It is to mention, that each pose is connected to its previous pose with a Pairwise Factor generated via ICP-Sequence Pose Measurements (cf. section 3.2.4). Those cannot be seen, because the distance of poses of the trajectory is too small with respect to the size of the drawn poses.

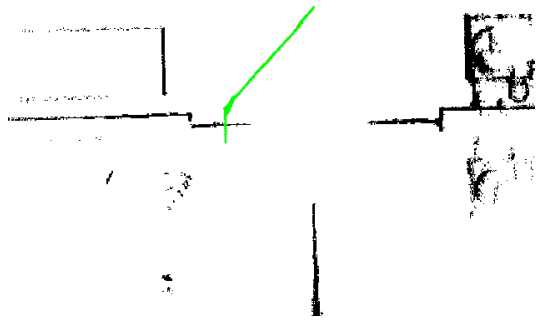

*Figure 8: Magnified section containing the trajectory (green) and the local area around the robot*

Figure 9 shows another robot's trajectory. This trajectory was build outside that factory hall in an area surrounded by multiple buildings and it contains 60 poses as well. Due to the higher velocity of the robot while acquiring the sensor readings, the trajectory covers a broader space. The higher velocity may be the reason for some inaccuracies, which become apparent in the upper left corner of Figure 9.

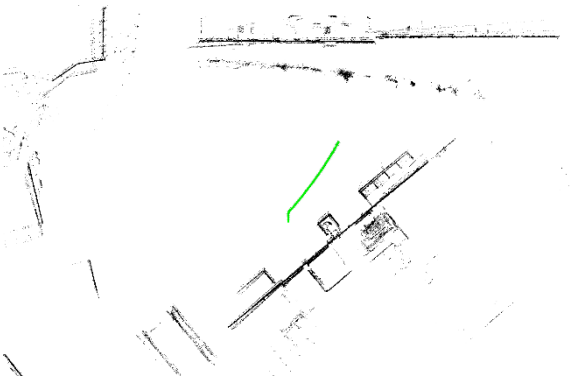

*Figure 9: Robot trajectory (green) and map of the environment rendered to the ground plane.*

The same trajectory is shown in Figure 10. To give a detailed overview of the scene, the image is magnified.

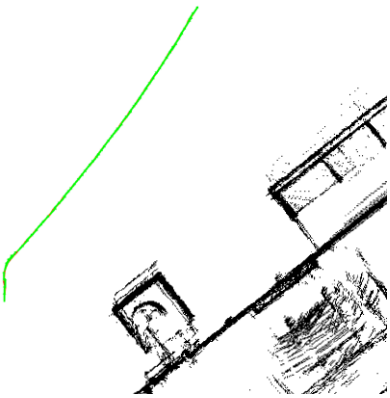

*Figure 10: Magnified section containing the trajectory (green) and the local area around the robot*

Figure 11 shows both trajectories with an established encounter. The poses, an initial guess was first introduced by, are connected with a blue line. Other factors established afterwards, to calculate the probability for that encounter to hold are left out for transparency.

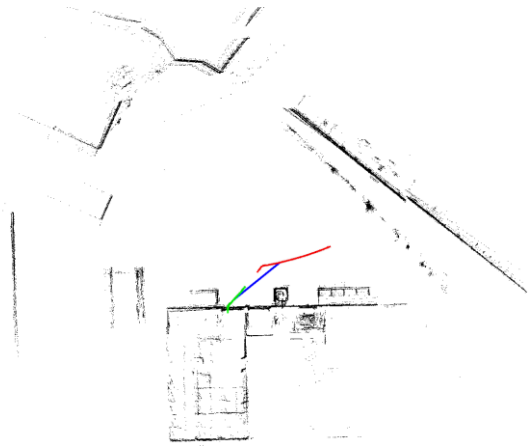

*Figure 11: Connected robot's trajectories. From trajectory 1 (green) to trajectory 2 (red). An encounter was established (blue).*

Figure 12 shows the same scene with the trajectories magnified. The Figure shows, that the algorithm is searching for the best set of corresponding features. Therefore it may occur, that a connection is not established between poses with the shortest distance, due to the fact that the distance

between poses from different trajectories is not known initially.

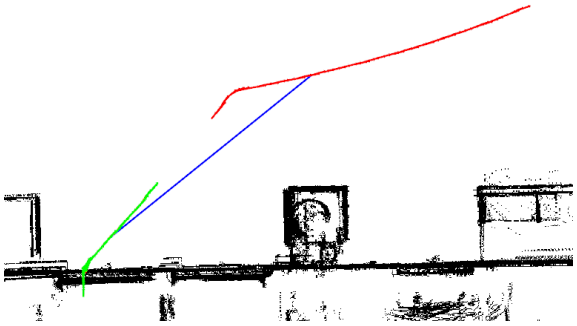

*Figure 12: Enlarged Part containing both trajectories.*

At least the same physical scene is presented as a three dimensional point cloud in Figure 13. The height is coded in colors. Furthermore the trajectories of both robots are drawn as well.

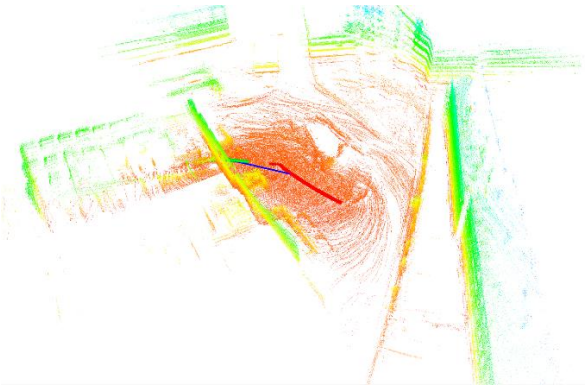

*Figure 13: 3D Point Cloud of the environment with both trajectories.*

## 5 CONCLUSIONS

It could be shown that the presented methods are capable of handling multiple independent robot's trajectories and build consistent maps from those trajectories. In future work long-term multi robot exploration tasks have to be investigated in much more detail. The approach has to be extended to aerial applications. While it is conceivable to handle point clouds from aerial based vehicles in a similar fashion like proposed in this paper, it has to be validated.

In future work we would like to investigate if it is possible and rewarding to use techniques proposed in this paper to incorporate prior knowledge in the map. It is for example imaginable to generate general features (cf. section 4.1) from floor planes of buildings – for indoor applications – or from street maps. If we are able to reproduce similar features with readings from environmental sensors we may be able to incorporate prior knowledge to our maps and / or we could obtain information about absolute pose of trajectories with respect to WGS84 for example.

## ACKNOWLEDGEMENTS

For implementation purposes lots of open source libraries around have been used. Therefore we'd like to mention the most commonly used ones. Those are GTSAM (Dellaert, 2006), iSAM (Kaess, 2008), Libpointmatcher (Pomerleau, 2011), PCL (Rusu, 2011). All dependencies will be published, when the source code will be published as well.

All presented work was done within the joint development project "**ANK**omm**E**n" funded by the German Federal Ministry of Economic Affairs and Energy administrated by the Space Administration of the DLR (funding code: 50NA1518).

## REFERENCES

Alexandre L., A., 2012. *3D Descriptors for Object and Category Recognition: a Comparative Evaluation*, In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and System (IROS)

Anderson L., A., A., Nygards, J., 2008. *C-SAM: Multi-Robot SLAM using Square Root Information Smoothing*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)

Carlone, L., Ng, M., K., Du, J., Bona, B., Indri, M., 2010. *Rao-Blackwellized Particle Filters Multi Robot SLAM with Unknown Initial Correspondences and Limited Communication*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 243 – 249

Censi, A., 2007. *An accurate closed-form estimate of ICP's covariance*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 3167 – 3172

Cunningham A., Wurm, K., A., Burgard, W., Dellaert, F., 2012. *Fully Distributed Scalable Smoothing and Mapping with Robust Multi-robot Data Association*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)

Dellaert, F., 2005. *Square Root SAM*. In Robotics: Science and Systems (RSS), pages 177 – 184

Dellaert, F., Kaess, M., 2006. *Square Root SAM: Simultaneous location and mapping via square root information smoothing*. In: I. J. Robotics Res., Nr. 12, pages 1181 – 1203

Eustice, R., Singh, H., Leonard, J.J., 2005. *Exactly sparse delayed-state filters*. In Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 2428 – 2435, Barcelona, Spain

Fenwick, J. W., Newman, P. M., Leonard, J. J., 2002. *Cooperative Concurrent Mapping and Localization*. In:

Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 1810-1817

Franchi, A., Oriolo, G., Stegagno, P., 2013. *Mutual Localization in Multi-Robot Systems using Anonymous Relative Measurements*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 81 – 95

Grisetti, G., Kümmerle, R. Stachniss C., Burgard, W., 2010. *A Tutorial on Graph-Based SLAM*, In: IEEE Intell. Transport. Syst. Mag., Nr.4, S. 31-43

Grisetti, G., Stachniss, C., Burgard, W., 2007. *Improved techniques for grid mapping with rao-blackwellized particle filters*. IEEE Transactions on Robotics pages 34 – 46

Gutmann, J.-S., Konolige, K., 1999. *Incremental mapping of large cyclic environments*. In: IEEE Int. Symp. on Computational Intelligence in Robotics and Automation (CIRA), pages 318 – 325

Hähnel, D., Burgard, W., Fox, D., Thrun, S., 2003. *An Efficient FastSLAM Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements.* In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pages 206-211, Las Vegas, NV, USA

Howard, A., 2006. *Multi-robot simultaneous localization and mapping using particle filters*. In: Intl. J. of Robotics Research, pages 1243 – 1256

Howard, A., 2004. *Multi-Robot Mapping using Manifold Representations*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 1360 – 1369

Kaess, M., Ranganathan, A., Dellaert, F., 2008. *iSAM: Incremental Smoothing and Mapping*. In: Int. Trans. On Robotics, TRO, vol. 24, no. 6, pages 1365 – 1378

Kim, B., Kaess M., Fletcher, L., John J., Bachrach, A., Roy, N., Teller S. J., 2010. *Multiple relative pose graphs for robust cooperative mapping.* In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 3185 – 3192

Konologie, K., 2004. Large-scale map-making. In: Proc. 21. AAAI National Conference on AI

Li, Y., Olson, E. B., 2010. *Extracting general-purpose features from LIDAR data.* In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 1388 – 1393

Lu, F., Milios, E., 1997. Globally consistent range scan alignment for environment mapping, Autonomous Robots, pages 333 – 349

Montijano, E., Matrinez, S., Sagues, C, 2011. *Distributed Robust Data Fusion Based on Dynamic Voting. In: Proc.* of the IEEE Int. Conf. on Robotics & Automation (ICRA)

Olson, E., Leonard, J., Teller, S., 2006. *Fast Iterative optimization of pose graphs with poor initial estimates*. In: Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), pages 2262 – 2269

Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S., 2013. *Comparing ICP variants on real-world data sets – Open-source library and experimental protocol*. In: Auton. Robots, Nr. 3, pages 133 – 148

Pomerleau, F., Magnenat, S., Colas, F., Liu, M., Siegwart, R., 2011. *Tracking a Depth Camera: Parameter Exploration for Fast ICP*. In: Proc. Of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)

Rusu, R., B., Cousins, S., 2011. *3D is here: Point cloud Library (PCL)*. In: Proc. Of the IEEE Int. Conf. on Robotics & Automation (ICRA)

Smith, R., Self, M., Cheesman, P., Cox, I., Wilfong, G. T., 1990. *Estimating uncertain spatial relationships in robotics*, In: Autonomous Robot Vehicles, Nr.8, S. 167 – 193

Thrun, S., Liu, Y., Koller, D. Ng, A. Y., Gharamani Z., Durrent-Whyte, H., 2004. *Simultaneous localization and mapping with sparse extended information filters*. Int. Journal of Robotics Research, pages 693 – 716

Tomasi, C., Kanade, T., 1991. *Detection and Tracking of Point Features*. In: Technical Report CMU-CS-91-132, Carnegie Mellon University

Zhou, X., S., Roumeliotis, S., I., 2006. *Multi-robot SLAM with Unknown Initial Correspondence: The Robot Rendezvous Case.* In Proc. of the IEEE/RSJ Int. Conf.