RAWSEEDS

Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets

WorkPackage 4

Deliverable D4.1
Benchmark Problems

Project no. 045144

Instrument: Specific Support Action

Thematic Priority: IST-2005-2.6.1 Advanced Robotics

date due: end of month 27 (January 31st, 2009)

authors: Domenico G. Sorrenti

contributors:

W. Burgard, G. Grisetti, M. Ruhnke, C. Stachniss, ALUFR

G. Fontana, M. Matteucci, POLIMI

D. Marzorati, UNIMIB

J. D. Tardós, UNIZAR

Date of completion of this document: Monday, March 16, 2009

Table of Contents

Executive Summary

This document is the description of the result of the activity performed in WP4, which aims at the definition of the Benchmark Problems.

A Benchmark Problem, in the RAWSEEDS parlance, is the union of the description of a problem, with the sensor streams collected during the activity of WP2, and the measures that we propose to use for the performance evaluation. The datasets have been collected within WP2 using the infrastructure defined in WP1, and have been validated during the activity of WP3.

Under the point of view of the data, in WP4 we had more a re-working than something fully new; quite differently from that, has been to deal with the issue of the rating methodologies. This part of the work, although more theoretic, proved difficult because of the different interpretations and objectives that the different partners gave to the performance measures. While this is typical also in other fields, when it comes to performance evaluation, we were expecting a smoother advancement of the work. We concluded our work with a rich set of performance measures, which will be subject to real life usage, in order to appreciate their acceptance in the community.

The measures for the performance evaluation that we devised aim at the evaluation of SLAM algorithms. While some of these measures aim at the evaluation of the intrinsic quality of the result, others aim at an evaluation that is based on the quality of the robot performance in some mobile robotic task, by using the result, and not by looking at the result per sé. We believe this last kind of performance measure to be a necessary complement to former and to be the definitive way of benchmarking robotic activities.

In the end, in WP4 we define 7 Benchmark Problems; these are then turned into the much more numerous Benchmark Problem instances, by means of the union with the collected datasets.

In the rest of the document "Benchmark Problem" is shortened in BP, "Benchmark Solution" in BS, and "Ground Truth" in "GT".

Introduction to the RAWSEEDS Benchmark Problems

As a short remind about the structure of the project work, we recall that providing the sensor datasets is the task of WP1, WP2 and WP3: in particular WP1 is about setting up the physical infrastructure devoted to the experimental activity, WP2 is about the physical collection of the datasets, while WP3 is about their validation. Providing methodologies for the quantitative evaluation of the performance of algorithms is part of the task of WP4. On the other hand, providing proven algorithms, having already demonstrated successful performances, to be used for comparison with newer solutions from the community, is the task of WP5. The setup of the website, which acts as the repository of the collected datasets and documentations, is the task of WP6.

> *A Benchmark Problem, or BP, is defined as the union of:*
>
> 1. *the detailed and unambiguous description of a task;*
> 2. *an extensive, detailed and validated collection of multisensorial data, gathered through experimental activity, to be used as the input for the execution of the task;*
> 3. *a rating methodology for the evaluation of the results of the task execution.*
>
> *The application of the given rating methodology to the output of an algorithm or piece of software designed to solve a Benchmark Problem produces a set of scores that can be used to assess the performance of the algorithm or compare it with other algorithms.*
>
> *[From RAWSEEDS' Description Of Work (Annex I to the Contract)]*

A result of the activity in WP4 is that we considered more appropriate to introduce a relatively small number of BPs, namely Laser SLAM, Monocular SLAM, Stereo SLAM, Trinocular SLAM, Omnidirectional vision SLAM, Sonar SLAM, Multi sensor SLAM. Each such BP can therefore be associated to a specific data collection session, so making a BP instance. While the BPs are not a large number, when applied to a number of dataset, they result in a large number of BP instances. In all the BPs, a few sensors streams, namely the IMU and the odometry, are currently considered always available. In the future other BPs, e.g., a pure Monocular SLAM BP, might be defined, where even wheel encoders could be saved.

BP \ dataset	1	...	$dataset_i$	...	n
⋮					
Laser SLAM			BP instance		
Monocular SLAM					
⋮					

The performance evaluation of a given algorithm is obtained applying the rating methodology proposed by RAWSEEDS to the output obtained from the execution of the algorithm on the relevant RAWSEEDS dataset. This approach, in the maximum generality, allows to evaluate the applicability and the performance of the algorithm to the widest set of conditions. Of course, not all algorithms will be able to deal with all the BP instances, i.e., not all the algorithms will be able to process all datasets. An example of such situation might be represented by the application of an algorithm, developed for indoor conditions, to outdoor scenarios, where the algorithm is likely to fail. In such cases the "score table" might simply not have the entry corresponding to the application

of this algorithm to the outdoor BP instances, if the author prefers not to give evidence to the cases of failure. Of course, the absence of the rating will play some role, in the overall evaluation of that algorithm that will be performed by the readers of the "score table".

Please notice that it is not the task of the RAWSEEDS project to compile such a table; on the other hand, the project objectives are to make it possible to have such a table compiled in an uniform way, so that different algorithms can be compared; e.g., the project will make available web forms for guiding the submission of BSs.

It might be convenient to summarize here the appearance that we expect the final score table to have, so to speak about the effect of the performance evaluation measures. We expect such table to have a row for each BS, i.e., the application of an algorithm to a BP instance. In the columns of the table there will be the values of the different performance measures that are relevant for that BP. The performance of the same algorithm, applied to the different datasets, relates to different BSs, and will be therefore shown in different rows.

In order to allow a realistic usage of the table, we expect to be able to allow the reader of the table to group the different BSs according to a few criteria; e.g, the datasets to which the algorithms have been applied, the specific sensor streams used by the BS, the BP, etc.

Generally speaking, our work has the objective of requiring each BS to submit enough information so to allow the replication of its results, by other groups. This is in agreement with the recent attention in the community to good experimental methodologies, which resulted also in a SIG of the EURON2 NoE.

Available Ground Truth

Ground Truth Systems for Pose (poseGT)

The obtainment of the Ground Truth for the robot pose has been the most difficult part of the whole project so far. As documented in the past project documents (D1.1, D1.2, D8.2, AD2.3), we were expecting to base on a commercial system, for indoor conditions, but we then discovered that this approach was not viable because of the many not explicitly mentioned hypotheses that were going to affect both the absolute accuracy and the cost of each single installation. We then designed, implemented and also validated a system based on a network of cameras, called shortly GTvision, see also AD2.3. This system was validated with respect to hand measurements and also cross-validated with respect to the accuracy attainable from the onboard sensors. This last option (to adopt as poseGT the best known algorithm based on the onboard sensors) was, since the original "DoW", our default solution, in case external systems would fail. Though we had such option, we were not fully happy with it because of the implication of being based on the onboard sensors, which would imply statistical dependence between the GT and the outcome produced by the BSs basing of those sensor streams. Moreover, beside the poseGT system based on the camera network, we also designed and implemented another poseGT system; this has been done after the delivery of AD2.3, and because we wanted to have an alternative in case of failures of the other system. This last poseGT system is based on a network of external and fixed laser scanner; the system provides what is shortly called GTlaser poseGT.

Lastly, after having obtained the two independent estimates of the poseGT, we also computed an integrated poseGT estimate, based on both the GTvision and the GTlaser. This integration has been performed by means of a Kalman Smoother, so to make full use of all the available data. As the accuracy of the two poseGT systems is not the same, and the manual validation in the data collection area is less accurate than the pose GT systems (differently from what was reported in AD2.3, where the smaller dimensions of the environment allowed hand measurements to be very accurate), we used the onboard laser scanners to obtain an estimate of the accuracy of the two poseGT systems, to be used in the Kalman Smoother. A a side effect, we also obtained an accurate map of the area. Of course, algorithms not making use of laser scanners can obtain the poseGT from the laser scans onboard the robot, for a much larger area than the one covered by the poseGT systems deployed indoor. A very rough comparison of pros and cons of the different indoor poseGT systems can be found in the table below.

	cost	setup	pose	frequen-cy	position accuracy	orientation accuracy	range	gaps in the working range
GTvision	low-cost	complex	6DoF	5Hz	less	average	smaller	a few
GTlaser	expensive	simple	3DoF	75Hz	higher	average	larger	absent
laser onboard, for BS not basing on them	expensive	very simple	3DoF	75Hz	higher	average	very large	absent

For what concerns the outdoor conditions, we carefully collected datasets in conditions with maximum exposure to GPS satellites, so to have a large area with the so-called RTK-GPS, in the maximum accuracy version ("RTK Fixed Integers"). Of course, such accuracy could not be obtained everywhere, because of clouds, building, trees, etc. Nevertheless, the GPS system provides an accurate estimate of the accuracy of the provided position, in terms of standard deviation, so that the reliability of the poseGT could be appreciated when rating SLAM algorithms.

A complete description of the poseGT systems can be found in AD2.3, D2.1, D2.2.

Ground Truth for Mapping (mappingGT)

In the project it was decided, after some verifications, to base on the executive drawings of the space explored by the robot, for the reference values of the maps. Such taken-for-perfect values are usually mentioned as Ground Truth in other domains and also in robotics benchmarking. The accuracy required by such kind of information is of course "the highest possible". Nevertheless, we consider that a realistic accuracy value can be set at about 0.1m.

We checked whether the executive drawings were accurate enough. This simplified validation, where the term "simplified" is used w.r.t. to the extremely large burden required for the validation of the poseGT, has been performed during the indoor data-acquisitions. The actual procedure was based on the verification of some distances in the executive drawings w.r.t. their actual values. The actual values have been determined by manual measurement, which means that the measure was obtained using the range measuring device mentioned in AD2.3, i.e., a laser-based ranging device typical of civil engineering. A total of 30 measurements have been performed in the area where the indoor poseGT is available, see Figure 1, and the quality of the mappingGT has been confirmed. Statistics of the errors are (in mm): mean = 0.55 σ = 90.34, confidence interval 95% = [-31.78, 32.88]; for further details, see D2.1. We can conclude that the executive drawings, at least in the area covered by the indoor poseGT systems, are accurate enough for being used as mappingGT.

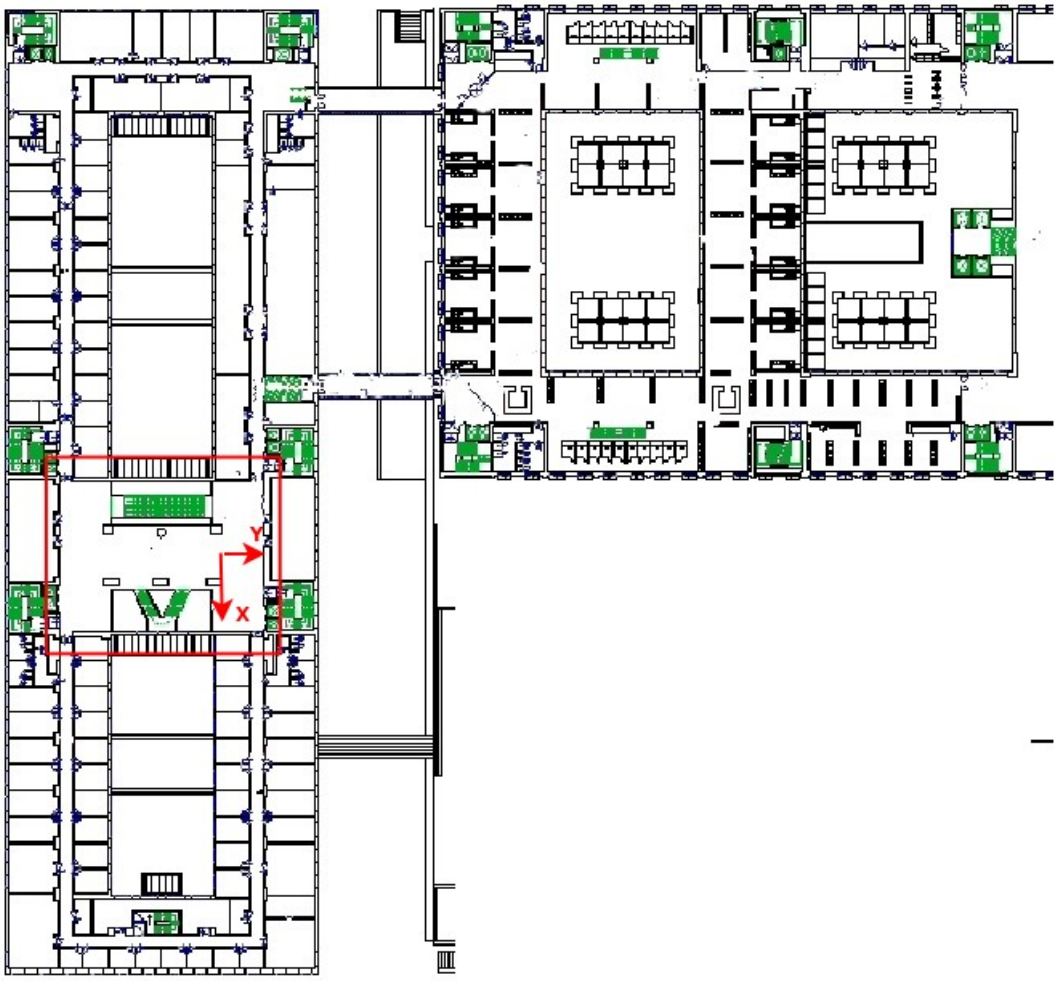

Figure 1: The area in the rectangle in red is approximatively the poseGT area.

Definition of features

The performance evaluation for SLAM and mapping BPs is based also on the verification of the accuracy of the reconstruction performed by the algorithm under evaluation. The other part deals with the accuracy of the trajectory, and we will deal with it later on.

The reconstruction accuracy can be evaluated for some salient features of the environment, and, of course, not for all points of the world. Therefore we need to deal with salient features in our work. On one hand, we have relevant elements in the mappingGT, i.e., the ones on which we base the performance evaluation. On the other hand, we have the features reconstructed by the mapping and SLAM systems.

For what concerns the mappingGT, it has been decided that the relevant elements are the corners, i.e., the intersections between two non-parallel segments in the birds-eye view of the environment; this is also independent of the specific format used for representing such segments in the 2D drawings. Each mappingGT corner $\boldsymbol{x}^{GT}$ is therefore represented by an "*id*", and by the 2D coordinates of the corner ($\boldsymbol{x}^{GT}_i = [\ i\ x_i\ y_i\ ]$).

For what concerns the features reconstructed by the algorithm, it has to be noticed that, in general, each BS give out a specific set of features. In other words, in the reconstructed map, a BS will provide the type of features that are more appropriate for its inner functioning; notice also that typically these features will not be the corners between the walls of the environment that we have in the mappingGT. Typical examples are salient points on the walls, grid maps, etc. We need to cover this gap between GT and reconstructed maps.

For what concerns the GT features, our proposal is as follows: RAWSEEDS will publish the lists of corners, extracted from the executive drawings with some algorithm (the executive drawings are sketched in Figure 2, and 3). As the extraction of the GT features is also error prone, we propose such lists as the recommended set of features, to be used for the evaluation of BSs, but we consider that it might be amended, in case some mistake is found by the community of the users. Of course, to change the list of GT features will require an adequate explanation of the error that has been found. As it will be mentioned later on, a change in the list of GT features will not prevent or make extremely cumbersome the re-computation of the performance evaluation measures, for the previously submitted BSs. If this would be cumbersome, then keeping the set of results up-to-date w.r.t. troubles in the list of GT features could become unfeasible (it would require to contact the BS author(s) and ask for a re-submission of the BS, a potentially unfeasible event). We are actually asking more to the BS authors, at the time of the submission, in order to gain such robustness.

Figure 2: Executive drawings of the Bicocca location, from this drawing the corners will be extracted and will constitute the list of mappingGT features.

Figure 3: Executive drawings of the Bovisa location, from this drawing the corners will be extracted and will constitute the list of mappingGT features.

We need also to cover the issue of the reconstructed features, i.e., the ones generated by the algorithms in their internal representation. On one hand, we will have a mappingGT, which will essentially be a point-based 2D map of the world. On the other hand, we will have the reconstructed maps that, depending on the specific BS, might be built basing on 2D or 3D features, segments or points, etc. We have therefore to compare the two representations, one built with the corners of the mappingGT, the other built from the map reconstructed by the BS. Our idea is that we can map reconstructed features on the mappingGT corners, provided we solve the mismatch of geometric primitives. The 2D-3D issue can be solved with a suitable "projection", while the geometric primitive mismatch could be solved with ad-hoc interpolations.

We considered two options for the projection: one is to project all reconstructed features on the mappingGT floor (in RAWSEEDS' datasets, floors are mostly horizontal); the other option is to select only the reconstructed features that belong to a certain vertical interval about the mappingGT floor. We decided for the first one, for its greater simplicity.

For the other aspect, i.e., the matching of different geometric primitives, our proposal is that each BS author has to derive "reconstructed corners" from the geometric primitives produced by her algorithm, in the cases of the algorithm output not being a directly corner-based map. A short list of steps for handling mismatching geometric primitive features follows; notice that very simple software helpers will largely speed up this work.

- 3D segments maps, like those produced by 3D segment based SLAM approaches:
  1. project the 3D extrema of the segments on the floor,
  2. reconstruct the corners by intersection of adjacent segments;
- 3D points maps, like those produced by monocular SLAM approaches:
  1. project the 3D points on the floor,
  2. cluster (human intervention) the 2D points into 2D segments,
  3. verify (human intervention) that the 2D segments (i.e., the reconstructed walls) are reliable,
  4. compute the support line of the segments,
  5. reconstruct the corners by intersection of adjacent segments;
- grid-based 2D maps like those produced by most approaches based on sonars and/or laser scanners:
  1. cluster (human intervention) the grid elements into segments,
  2. verify (human intervention) that the 2D segments (i.e., the reconstructed walls) are reliable,
  3. compute the support line of the segments, taking into account the PDF on the grid,
  4. reconstruct the corners by intersection of adjacent segments;

Performance evaluation measures

In this part of the document we propose the performance evaluation measures. This set of measures is independent of the BPs. While some of these measures aim at the evaluation of the absolute quality of the outcome of BSs, others aim at an evaluation that is based on the quality of the robot performance, in some mobile robotic task, but performed by using the result of the algorithm under evaluation. We call these measures "usage-based".

Moreover, the proposed performance evaluation measures can be of one out of two types: one is *recommended*, i.e., a performance evaluation measure that RAWSEEDS will be recommending the BS contributors to provide; the other is *mandatory*, i.e., BS failing to provide this measure will not be admissible for publication as a RAWSEEDS BS.

Also another distinction can be observed in our proposed performance measures: some of them are based on mapping performance, while other base on the trajectory accuracy. Basing on trajectories, we can gain two important properties. First, we can naturally compare the result of algorithms that generate different types of metric maps, such as feature-maps or occupancy grid maps. Second, the method is invariant to the sensor setup of the robot. For example, the result of a graph-based SLAM approach working on laser range data can be compared with the result of vision-based FastSLAM. The only property that is required is that the SLAM algorithm has to estimate the trajectory of the robot in terms of a set of poses. The performance evaluation will be performed on this set.

Of course, as typical in the performance evaluation in many fields, also in our case it is unlikely that anybody could define a definitive performance measure, i.e., one on which all involved actors will agree. This is regarded as work in progress, and this is also in agreement with the fact that the aim of RAWSEEDS is to facilitate the performance evaluation, and also to stimulate the spreading of a performance evaluation culture. Whether our proposed performance evaluation measures will perfectly fit their expected usage or not, can only be a matter of ex-post analysis.

ME (Mapping Error)

The Mapping Error is a measure that is intended to capture the accuracy of the reconstructed map. As a large number of geometric primitives are usually present in realistic maps, we need to provide an accuracy measure that is based on the distribution of the error, so to have it averaged on a statistically significant number of samples. Such error distribution usually won't be following accurately a normal distribution, but we believe that the relevant aspects that we want to capture for performance evaluation can be well represented by the usual two parameters of a normal (mean and standard deviation). Confidence interval will also be considered, to take into account the cardinality of the sample.

The computation of the value of the Mapping Error (ME), for a given BS applied to a given BP instance, requires to follow these steps:

1. given the set of features in the mappingGT $\{ \boldsymbol{x}^{GT}_i \}$, the user submitting the BS has to determine, in the output of the BS, i.e., in the set of reconstructed features $\{ \boldsymbol{x}_l \}$, the features corresponding to the ones in the mappingGT; this task concludes with the list of associations between mappingGT and reconstructed map $\{ < \boldsymbol{x}^{GT}_i , \boldsymbol{x}_l > \}$, which has to be provided; the set of reconstructed features, $\{ \boldsymbol{x}_l \}$, is also to be provided, in order to allow for a re-computation of the performance in case of any change in the mappingGT;
2. BSs that detect features that are not easy to associate with features in the mappingGT, need to follow a different path, for what concerns the determination of the set $\{ \boldsymbol{x}_l \}$. Examples of such BS are those working with grid-based representations, those based on sparse image patches, etc. Examples of the ahe approaches to be followed for such conversions are mentioned in the previous section. The requirement imposed to such BSs is to provide a

complete description of the procedure followed to determine, from the originally reconstructed features, the ones used in the performance evaluation;

3 the author of the BS has then to compute the geometric distance between all k pairs of mappingGT features $\{D^{GT}_k = \| \boldsymbol{x}^{GT}_i - \boldsymbol{x}^{GT}_j \| \}$, and the distance between the corresponding k pairs of reconstructed features $\{D_k = \| \boldsymbol{x}_l - \boldsymbol{x}_m \| \}$, where $\boldsymbol{x}_l$ is the correspondent of $\boldsymbol{x}^{GT}_i$, and $\boldsymbol{x}_m$ the correspondent of $\boldsymbol{x}^{GT}_j$;

4 for BSs that cannot determine the absolute scale of the environment, e.g., algorithms based solely on monocular vision, the mappingGT data can be used to define the scale of the reconstructed map;

5 for each of the distances above, the author has to compute the normalized difference $N_r = (D_r - D^{GT}_r) / D^{GT}_r$, where D_r is the r-th distance measured on the map produced by the BS, and D^{GT}_r is the corresponding distance measured in the mappingGT;

6 ME =

6.1 mean of the set of normalized differences $\{ N_r \}$;

6.2 standard deviation of the set of normalized differences $\{ N_r \}$;

6.3 confidence interval (3σ) of the set of normalized differences $\{ N_r \}$;

7 ME = $[\quad \bar{N}_r \quad , \sigma_{Nr} , conf.interval.endpoint1_{3\sigma}, conf.interval.endpoint2_{3\sigma}]^T$;

Notice that the ME measure does not require the author to determine the pose of the map produced by her BS w.r.t. the mappingGT, i.e., to align the reconstructed map to the mappingGT; this property is a consequence of the measure being based only on relative distances.

ME is a *recommended* measure.

ATE (Absolute Trajectory Error)

The absolute trajectory error is a useful performance measure that captures at the same time both the accuracy in mapping and in localization. It is a compact, although indirect, representation of the accumulation of errors due to data associations, biases in the resulting map and robot pose estimates. It can be reduced by loop closures, which are informative events that happen during the execution of SLAM algorithms.

An instantaneous measure is provided at the poseGT frequency (50Hz), i.e., a robot pose estimate is provided for each poseGT value available, and these instantaneous measures are then integrated. The instantaneous measures are provided as part of the BS, so that anybody could eventually compute other statistics on these values, and/or recompute the performance measure in case of changes to the poseGT.

The robot pose estimate has to be referred to the same reference frame as the poseGT, in order to be operated on. In order to reach this situation please refer to "*Alignment of mappingGT and reconstructed maps*" hereafter.

The computation of the value of the ATE requires to follow these steps:

1 for each instant, provide the robot pose estimate;

2 put all reconstructed robot pose in a file, to be provided as part of the BS; the file is a list of lines, one for each pose; for each pose the format is $< timestamp, [x_j, y_j \theta_j] >$;

3 for each instant where the poseGT is available, compute the distance, in terms of translation, between the poseGT and the reconstructed robot pose; $d_j = \| trans(\boldsymbol{x}_j) - trans(\boldsymbol{x}^{GT}_j) \|$, the orientation has been considered implicitly taken into account by the high sampling rate of the position;

4 put all error distances in a file, to be provided as part of the BS; the file is a list of lines, one for each pose; for each pose the format is <*timestamp*, d_j >;

5 ATE =

5.1 mean of the translation error { d_j };

5.2 standard deviation of the translation error { d_j };

5.3 confidence interval of the translation error { d_j };

6 ATE = [$\bar{d}_j$ σ_{d_j} *conf.int.d1*$_{3\sigma}$ *conf.int.d2*$_{3\sigma}$]^T ;

The ATE measure is *mandatory.*

REC (Rough Estimate of Complexity)

This measure aims at allowing the appreciation, though in a rough way, of the complexity of the algorithm under evaluation. It is based on the running time of the algorithm. In order to avoid any bias induced by the specific machine on which the algorithm is executed, only the overall shape is considered, and only in a qualitative way. The BS author is required to provide the running time as obtained by her system during the execution, with a frequency of 1 second to process all data streams involved up to that instant. The file to be provided is a list of lines, one for each second; the format is < *timestamp*, *running time* >. The time origin is the start of the dataset.

The REC measure is *mandatory.*

SLE (Self-Localization Error)

The Self-Localization Error is an interesting measure, in our view, as it is intended to evaluate the effectiveness of a given reconstructed map, in accordance with the RAWSEEDS idea of rating the *real-life usefulness* of the outcome of the BSs.

Once the algorithm have built a map with SLAM on a given BP instance (*slamdataset)*, the idea is to use different datasets (*localizationdatasets*), from the same location, and a localization algorithm that optimally suites the map representation of the algorithm, to localize the robot in the map. 10 different time-stamps are defined to start from, as if the robot were kidnapped. The robot pose in the GT area has to be reconstructed, basing on the map built before. This measure requires BS providers to determine (i.e., to develop and/or to evaluate already available ones) a localization algorithm that fits nicely with the built map.

The operative definition of the value of the Self-Localization Error (SLE) for a given BS requires to follow these steps:

1 select the self-localization algorithm of your best choice, for optimal matching performance between the SLAM outcome and the self-localization algorithms, and document the choice;

2 the datasets on which the SLAM output is going to be used are, by default, all the datasets from the same location; these are called the *localizationdatasets*;

3 run the algorithm under evaluation on the *slamdataset*, and get the produced data (map);

4 for each starting timestamp, feed the localization algorithm with the sensor stream(s) from a *localizationdataset*, beginning from that time-stamp, and reconstruct the robot poses by running the localization algorithm. This reconstruction has to be performed for all the timestamps belonging to the time intervals subsequent to the starting time-stamp, where the poseGT is available. Repeat for all the starting time-stamps, and for all the *localizationdatasets*;

5 if the BS and/or the localization algorithm, cannot provide the reconstructed robot pose at

each required timestamp, then the BS author has to provide an interpolation as well as a detailed explanation of how the interpolation has been computed; the level of detail of the description must allow replication of the results;

6 for each such timestamp, compute the distance of the reconstructed pose from the poseGT; this pose has to be referred to the same reference frame of the poseGT, to reach this situation please refer to "*Alignment of mappingGT and reconstructed maps*" hereafter;

7 SLE = mean, standard deviation and confidence interval of the translation errors $d_j = \|trans(\boldsymbol{x}_j) - trans(\boldsymbol{x}_j^{GT})\|$;

8 SLE = [$\bar{d}_j$ σ_{d_j} *conf.int.d1*$_{3\sigma}$ *conf.int.d2*$_{3\sigma}$ errtraslerrtraslerrorienerrorien]^T ;

This measure is a *recommended* measure.

RPE (Relative Pose Error)

In this section, we propose the Relative Pose Error metric for measuring the performance of a SLAM algorithm; this measure considers the poses of the robot during data acquisition, as the ATE and the SLE measures. Let $x_{1:T}$ be the poses of the robot estimated by a SLAM algorithm from time step *1* to *T*. Let $x^{GT}_{1:T}$ be the real poses of the robot, i.e., the poseGT locations. A straightforward error metric could be defined as:

$$\epsilon(x_{1:T}) = \Sigma_{t=1}^{T} (x_t \ominus x_t^{GT})^2 \quad \text{(Eq. 1)}$$

where $\oplus$ is the standard motion composition operator and $\ominus$ its inverse. Let $\boldsymbol{x}_{i,j} = \boldsymbol{x}_j \ominus \boldsymbol{x}_i$ be the relative transformation that moves the node $\boldsymbol{x}_i$ onto $\boldsymbol{x}_j$ and $\boldsymbol{x}^{GT}_{i,j}$ accordingly. Eq. 1 can be rewritten as

$$\epsilon(x_{1:T}) = \Sigma_{t=1}^{T-1} ((x_1 \oplus x_{1,2} \oplus \ldots \oplus x_{t-1,t}) \ominus (x_1^{GT} \oplus x_{1,2}^{GT} \oplus \ldots \oplus x_{t-1,t}^{GT}))^2 \quad \text{(Eq. 2)}$$

We claim that this metric is suboptimal for comparing the result of a SLAM algorithm. To illustrate this, consider the following 1D example in which a robot travels along a straight line. Let the robot make a translational error of e during the first motion, $\boldsymbol{x}_{1,2} = \boldsymbol{x}^{GT}_{1,2} + e$, and perfect estimates at all other points in time $\boldsymbol{x}_{t,\,t+1} = \boldsymbol{x}^{GT}_{t,\,t+1}$ for $t > 1$. Thus, the error according to Eq. 2, will be $T \cdot e$, since $\boldsymbol{x}_{1,2}$ is contained in every pose estimate for $t > 1$. If we, however, estimate the trajectory backwards starting from $\boldsymbol{x}_T$ to $\boldsymbol{x}_1$ or alternatively by shifting the whole map by e, we obtain an error of e only. This indicates that such an error estimate is suboptimal for comparing the results of a SLAM algorithm. See also Figure 4 for an illustration.

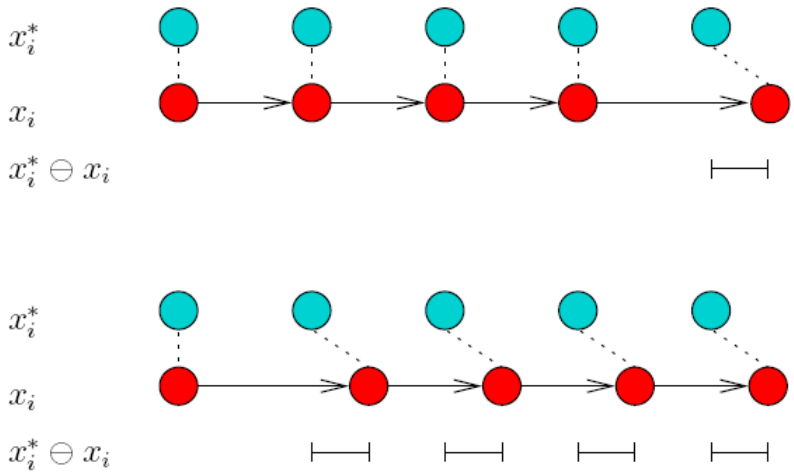

Figure 4: This figure illustrates a simple example where the metric in Eq. 1 fails. The blue circles show the GT positions of the robot $\{x^*_i\}$ (x^*_i is here used instead of x^{GT}_i) while the red circles show the actual positions of the robot $\{x_i\}$. The correspondence between the measured locations and the GT is shown with dashed lines, and the direction of motion of the robot is highlighted with arrows. In the situation shown in the upper part, the robot makes a small mistake at the end of the path. This results in a small error. Conversely, in the situation illustrated on the bottom part of the figure the robot makes a small error, of the same entity, but at the beginning of the travel, thus resulting in a much bigger global error.

In the past, the so-called NEES measure proposed in [1] as

$$\epsilon(x_{1:T}) = \Sigma_{t=1}^{T} (x_t - x_t^{\mathrm{GT}})^T \Omega_t (x_t - x_t^{\mathrm{GT}}) \quad \text{(Eq. 3)}$$

has often been used to evaluate the results of a SLAM approach. Here Ω_t represents the information matrix of the pose x_t . The NEES measure, however, suffers from a similar problem than Eq. 1 since it uses absolute poses to compute ε. In addition to that, not all SLAM algorithms provide an estimate of the information matrix and thus cannot be compared based on Eq. 3. Furthermore, an algorithm can improve its score by simply underestimating Ω.

Based on this experience, we propose to use a measure based on the relative displacement between poses to perform comparisons. Instead of comparing x to x^{GT} (in the global reference frame), we do the operation based on $\boldsymbol{x}_{i,j}$ and $\boldsymbol{x}^{GT}_{i,j}$ as

$$\epsilon(.) = \Sigma_{ij} (x_{ij} \ominus x_{ij}^{\mathrm{GT}})^2 \quad \text{(Eq. 4)}$$

In this case, the error in the above-mentioned example will be consistently estimated as e, no matter where the map is located in the space or in which order the data is processed.

This measure can be interpreted as the deformation energy that is needed to change the estimated trajectory into the ground truth. This can be done – similarly to the ideas of the graph mapping introduced by Lu and Milios [2] – by considering the nodes as masses and connections between them as springs. Eq. 4 can be rewritten as:

$$\epsilon(.) = \frac{1}{N} \Sigma_{i,j} trans(x_{ij} \ominus x_{ij}^{\mathrm{GT}})^2 + rot(x_{ij} \ominus x_{ij}^{\mathrm{GT}})^2 \quad \text{(Eq. 5)}$$

where N is the number of relative relations and $trans(\cdot)$ and $rot(\cdot)$ are used to separate the translational and rotation components. We suggest to provide both quantities individually. In addition to Eq. 5, one can define the metric according to the absolute error rather than the energy.

Then the metric can be specified accordingly as:

$$\epsilon(.) = \frac{1}{N} \Sigma_{i,j} trans \| x_{ij} \ominus x_{ij}^{\mathrm{GT}} \| + rot \| x_{ij} \ominus x_{ij}^{\mathrm{GT}} \| \quad \text{(Eq. 6)}$$

The squared error measures the average energy per constraint, required to deform the current estimate in the ground truth. The absolute error can be interpreted as the average metric displacement between the estimated and the true relative transformations.

The mathematical definition of this metric, however, leaves open which relative displacements $\boldsymbol{x}_{ji}$ are included in the summation in Eq. 1. We propose that the relative displacements have to be provided in the benchmarking problem, which according to the Annex I, provides the log file and the GT information. Anyway, the RPE metric can be tailored about which relative displacements $\boldsymbol{x}_{ji}$ to include in the summation in Eq. 4. Evaluating two approaches based on a different set of relative pose displacements will obviously result in two different scores. As we will show in the remainder of this section, the set $\boldsymbol{x}_{(...)}$, and thus $\boldsymbol{x}^{GT}{}_{(...)}$, can be defined to highlight certain properties of an algorithm. Nevertheless, in the context of RAWSEEDS, we will base on the available set of poseGT, i.e., we will use all the relative pairs of consecutive poses from the poseGT.

Selecting Relative Displacements for Evaluation

Benchmarks are designed to compare different algorithms. In the case of SLAM systems, however, the task the robot finally has to solve should define the required accuracy and this information should be considered in the measure.

For example, a robot generating drawings of buildings should build a map that reflects the geometry of a building as accurately as possible. In contrast to that, a robot performing navigation tasks only requires a map that can be used to robustly localize itself and to compute valid trajectories to a goal locations. To carry out this task, it is in most cases sufficient that the map is topologically consistent and that its observations can be locally matched to the map. A map having these properties is often referred to as locally consistent. Figure 5 illustrates the concept of locally consistent maps which are suited for a robot to carry out navigation tasks.

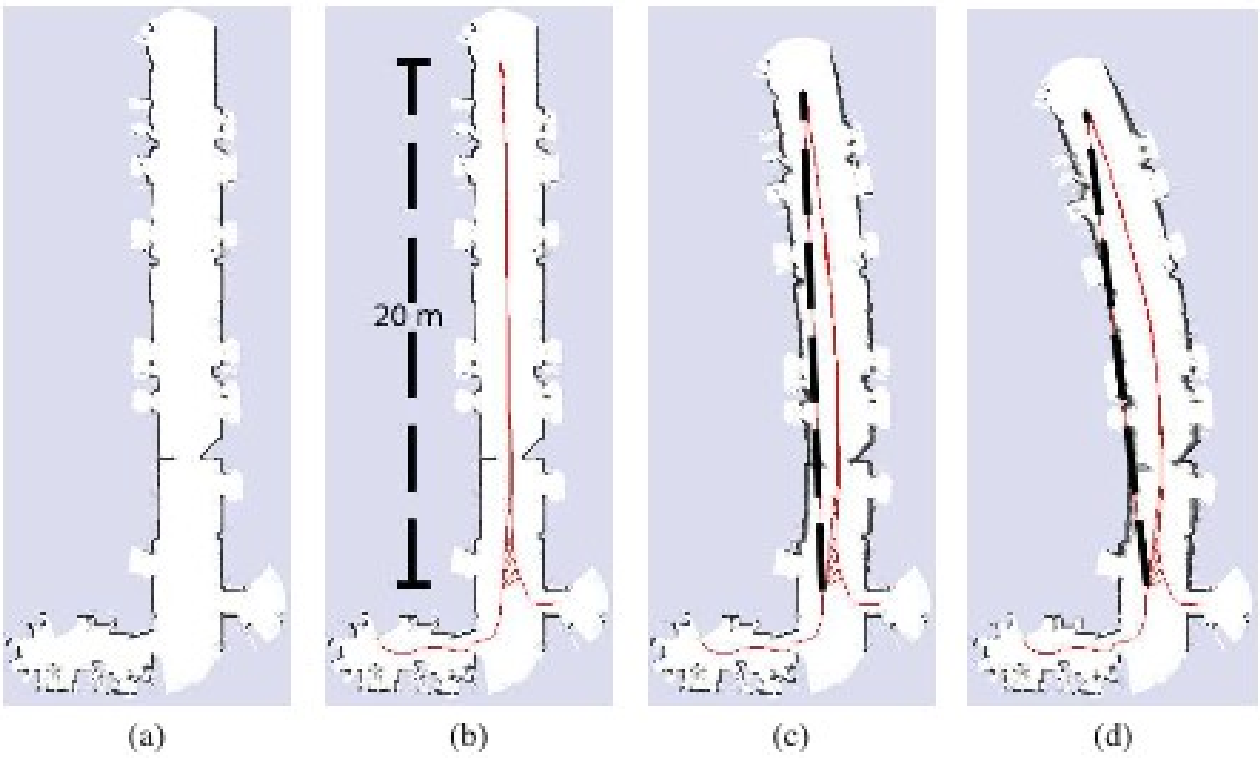

Figure 5: Example of the RPE performance measure on maps generated using different sensor setups. The relation between close-by positions are determined by a human assisted scan-alignment procedure performed on scans acquired at close-by locations. The long dashed line represents a relation added by manually measuring the relative distance at two locations of the robot: (a) the mappingGT, as obtained from the relative measurements, (b) the mappingGT with superimposed the network of relative measurements, (c) a map obtained by scan matching using a 4 meters range sensor, with the superimposed relation (this map is still usable for navigating a robot), (d) a map obtained by cropping the range of the sensor to 3 meters. Whereas the quality of the rightmost map is visibly decreased, also this map is adequate for robot navigation since it preserves a correct topology of the environment (all doorways are still visible) and it correctly reflects the local metric structure of the corridor.

By selecting the relative displacements $\boldsymbol{x}_{ji}$ used in Eq. 4 for a given dataset, the user can highlight certain properties and thus design a measure for evaluating an approach given the application in mind.

For example, by adding only known relative displacements between *nearby poses based on visibility*, a local consistency is highlighted. In contrast to that, by adding known relative displacements of far away poses, for example, provided by an accurate external measurement or by background knowledge, the accuracy of the overall geometry of the mapped environment enforced. In this way, one can incorporate the knowledge into the benchmark that, for example, a corridor has a certain length and is straight. This is a nice property of the metric and in the remainder of this section, we discuss how to obtain the displacements.

Obtaining Reference Relations in Indoor Environments

In practice, the key question regarding Eq. 4 is how to determine the true relative displacements between poses. Obviously, the true values are not available. However, we can determine close-to-true values by using the information recorded by the mobile robot and the background knowledge

of the human recording the datasets. This, of course, involves manual work, but is from our perspective the best method for obtaining such relations. Please, note that the metric presented above is independent of the actual sensor used. In the remainder of this deliverable, however, we will concentrate on laser range finders, which are popular sensors in robotics at the moment, and that are the most accurate sensor on the RAWSEEDS robot. To evaluate an approach operating on a different sensor modality, one has two possibilities. Either one temporarily mounts a laser range finder on the robot (if this is possible) or has to provide a method for accurately determining the relative displacement between two poses from which an observation has been taken that observes the same part of the space. Here, we propose the following strategy. First, one seeks for an initial guess about the relative displacement between poses. Based on the knowledge of the human, a wrong initial guess can be easily discarded since the human "knows" the structure of the environment. In a second step, a refinement is proposed based on manual interaction.

In most cases, researchers in robotics will have SLAM algorithms at hand that can be used to compute an initial guess about the poses of the robot. In the recent years, several accurate methods have been proposed to serve as such a guess. By manually inspecting the estimates of the algorithm, a human can accept or discard a match. It is important to note that the output is not more than an initial guess and it is used to estimate the visibility constraints which will be used in the next step.

Based on the initial guess about the position of the robot for a given time step, it is possible to determine which observations in the dataset should have covered the same part of the space or the same objects. For a laser range finder, this can easily be achieved. Between each pair of poses that are visible, one adds a relative displacement into a candidate set. In the next step, a human processes the candidate set to eliminate wrong hypotheses by visualizing the observation in a common reference frame. This requires manual interaction but allows for eliminating wrong matches and outliers with high precision. Since we aim to find the best possible relative displacement, we perform pair-wise registration procedure to refine the estimates of the observation registration method. It furthermore allows the user to manually adjust the relative offset between poses so that the pairs of observations fit perfectly. Alternatively, the pair can be discarded. This approach might sound work-intensive but with an appropriate user interface, this task can be carried out without a large waste of resources. For example, for a standard dataset with 1700 relations, it took an unexperienced user approximately four hours to extract the relative translations that then served as the input to the error calculation. Figure 7 shows a screen-shot of the user interface used for evaluation.

In addition to the relative transformations added upon visibility and matching of observations, one can directly incorporate additional relations resulting from other sources of information, for example, given the knowledge about the length of a corridor in an environment. By adding a relation between two poses — each at one side of the corridor — one can easily incorporate knowledge about the global geometry of an environment if this is available. This fact is, for example, illustrated by the black dashed line in Figure 5 that implies a known distance between two not neighboring poses in a corridor. Figure 6 plots a corresponding error introduced by the relations.

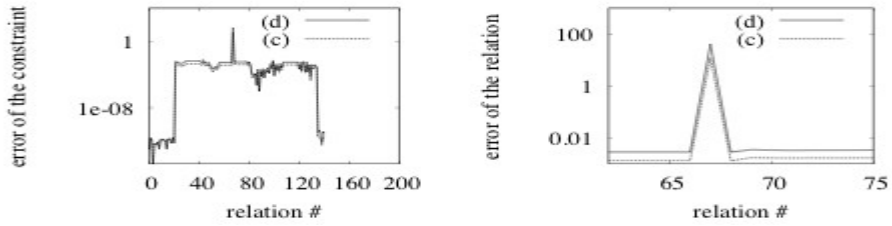

Figure 6: This figure shows the behavior of the RPE error metric for the maps (c) and (d) in the previous Figure. On the left we plot the error introduced by the individual relations. The right plot is a magnification of the left one in the region corresponding to the manually introduced relations marked on the images with the dashed line. This results in a significant increase of the global ε of SLAM results under comparison.

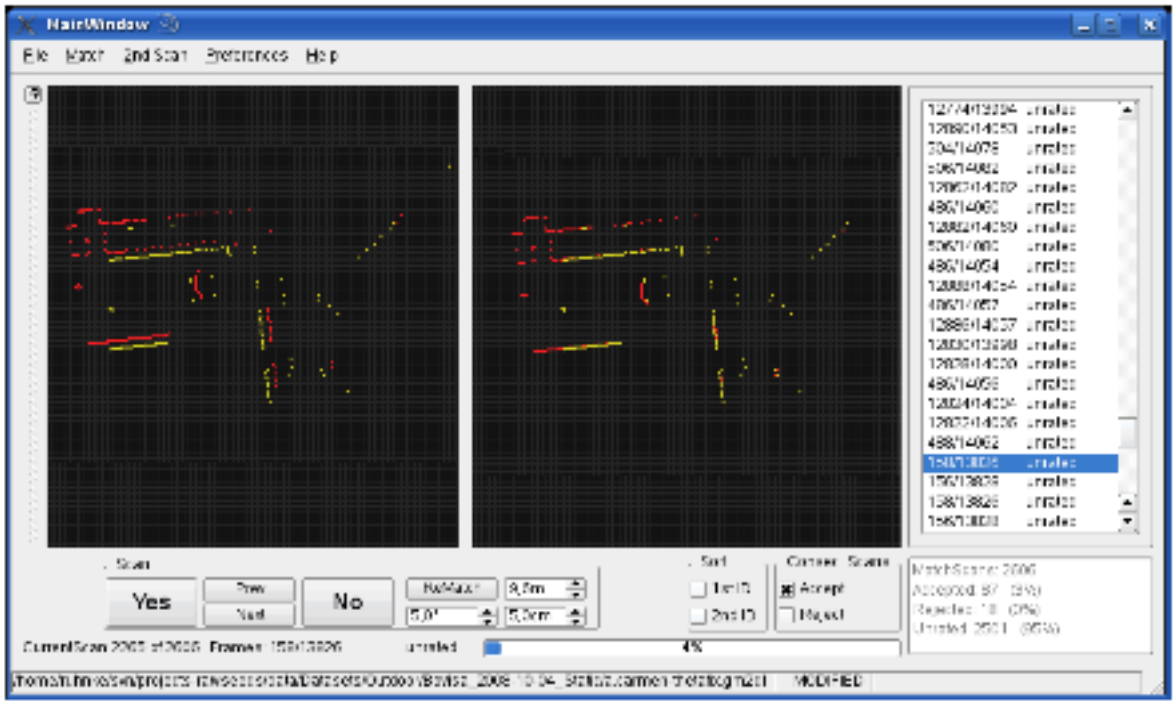

Figure 7: User interface for matching, accepting, and discarding pairs of observations.

The RPE measure is a *recommended* measure.

The SLAM Benchmark Problems

Description of the SLAM problem

The basic ability that a mobile robot must necessarily possess to autonomously operate in an unstructured (or partially structured) environment is moving safely and without collisions.

This requires, in particular, the robot to be able to localize itself, i.e., to estimate its own position in the environment. Moreover, the robot has generally to be able to construct some form of internal representation of the environment, i.e., a map, in order to determine its position on the map.

The presence of such capabilities are, of course, not sufficient to ensure that the robot is able to reach the goal; but they can be thought of as a necessary condition for a robot to be capable of autonomous navigation.

There are special cases in which the capability of autonomous building a world model is not required, e.g. when the map is already known with sufficient precision and it does not change with time. Real-world applications where these conditions hold are not frequent.

In the most common applications, the problems of mapping and self-localization have to be tackled and solved simultaneously by the robotic system, in order to register the information gained in the many poses passed during the exploration. This is not a trivial task, and it is usually identified as SLAM (Simultaneous Localization And Mapping).

The main difficulty comes from the fact that the data on which the robot can perform its elaborations come from sensors, and are thus affected by their imperfections, such as:

- limited spatial range and/or field-of-perception;
- noise;
- sensibility to spurious effects;
- low dynamic range;
- systematic errors or drift effects;
- failures.

These imperfections are very significant for any sensor, including costly state-of-the-art ones; but they become increasingly stringent as the cost of the sensors decrease. Very sophisticated algorithms are needed to process sensor output, in order to extract the information needed to solve the mapping and localization problems.

These algorithms become much more complex when multiple sensors are used (as it is frequently done, to partially compensate for the intrinsic limitations of each sensor), because they need to include a process of sensor fusion between the data coming from different sensors. Sensor fusion is most difficult when different kinds of sensors are employed (e.g., cameras and sonars), which is exactly what is generally done to explore different aspects of the environment and to exploit the capabilities of different sensor technologies.

Cheap sensors (such as the ones that present and future mass-market robotic applications are forced to employ for cost reasons) are characterized by quite low performance, and so they need the most sophisticated algorithms to be effective sensors. Of course, the ability to use cheap sensors and nonetheless build high-performance robotic products is necessary for the diffusion of mass-market robotic applications. However, the use of sophisticated algorithms does not necessarily have a significant impact on the final cost of a robotic product, as the main economic and conceptual effort is required by the development and test phases of the algorithms, while implementation can be usually made with inexpensive hardware.

The study, design, engineering and marketing of autonomous robotic systems and solutions relies on the fact that the actors involved (enterprises and research groups) possess or can easily acquire the tools to develop and test sophisticated SLAM algorithms. Given the relevance of the SLAM problem in the autonomous robotics domain, the RAWSEEDS toolkit is likely to become a significant tool.

The RAWSEEDS toolkit will include the following BPs, but others could be conceived, and perhaps added in the future. Laser SLAM, Monocular SLAM, stereo SLAM, Trinocular SLAM, Omnidirectional vision SLAM, Sonar SLAM, Multi sensor SLAM.

It has to be noticed that most of these BPs currently include also access to two other sensor streams, i.e., IMU and odometry.

- **Laser SLAM BP**

This BP tackles the SLAM problem basing on data from the laser scanners streams. This is a quite well-known sensor in the robotic community, mainly for its accuracy and range, though it is an expensive, bulky, and power-hungry device.

- **Monocular SLAM BP**

This BP tackles the SLAM problem basing on data from a single perspective camera. This is a device with a promising potential, because of the richness of the output, the light weight and cost. Many recent research is actually basing on this type of device. The instance of device that is onboard the RAWSEEDS robot is a low-cost device.

- **Stereo SLAM BP**

This BP tackles the SLAM problem basing on data from stereo vision. This is a device slightly more complex than a single camera, though potentially capable to provide 3D data. According to the specific processing performed in the logical sensor, one might obtain segments or points, 2D or 3D.

- **Trinocular SLAM BP**

This BP tackles the SLAM problem basing on data from trinocular stereo vision. This is a device more complex than a single camera, though capable to provide 3D data. Main difference w.r.t. stereo is the usage of the 3rd camera for simplifying the stereo matching. Commercial devices producing 3D point are available, while also devices based on 3D segments are common.

- **Omnidirectional vision SLAM BP**

This BP tackles the SLAM problem basing on data from an omnidirectional vision system. This is a 2D device, where the 3D data is generated in the SLAM filter, by means of the parallax produced by the observer motion, like for other vision-based SLAM BPs.

- **Multi sensor SLAM BP**

This BP aims at using more than one sensor stream at the same time, in order to appreciate the combined effect of complex sensor suites, on reliability, accuracy, etc.

Further BPs that can be devised basing on RAWSEEDS datasets

The RAWSEEDS datasets can be quite useful also for mobile robotics problems other than SLAM. These are all the problems where the offline collection of the sensor streams is realistic. Examples of such BP are the mapping BP, the self-localization BP, multi-robot SLAM, multi-robot mapping.

The main difference between mapping and SLAM is the absence of the requirement for online execution, for mapping. In such BP the algorithm can rely on the whole set of sensor data, to build a map of the environment. Notice that online, nowadays, does not imply real-time. A mapping BS trades being online with the accuracy. On one hand, being online, means handling new sensor data at each new activation, which typically implies a recursive formulation of the estimation problem, and also a more or less consistent exploitation of the Markov hypothesis during the exploration. On the other hand, the accuracy attainable cannot reach the levels of an offline algorithm, where all data can be used for smoothing away the noise as well as to invest time in order to reduce the effects of non-linearities.

Self-localization is devoted to localize the robot in the map of the working space, therefore making it a core component of the navigation system. There are cases where such map, in terms of executive drawings, is both available and accurate enough for being used in mobile robotics. Most of times, though, the appropriate way to approach this problem is through a map building process.

Multi-robot SLAM, and multi-robot mapping are cooperative approaches to the SLAM and mapping activities. More than one robot is moving in the work space and collecting its own sensor data streams. In RAWSEEDS, whose datsets are collected once at a time, this can be obtained by considering a few datasets, from the same location, like if they were collected starting a the same time. The datasets need to be shifted in time to make the multiple explorations appear simultaneous. Such datasets will be referred to as "component datasets", and their union will be termed a "multi-robot dataset". It is also important to note that, beside being from the same location, the component datasets need to satisfy some other condition, to be considered as simultaneous; e.g., two datasets taken in very different lighting conditions like at noon and at midnight, cannot be the component of the same multi-robot dataset. Given the degrees of freedom in shifting the timelines of the component datasets and the number of components, the issue arise of how to check whether a given combination appears as a real multi-robot dataset. We devised a condition that holds true when a multi-robot dataset of ours is not appearing as a real one: given a multi-robot dataset built by aggregating and possibly time-shifting multiple datasets, we claim that it cannot come from a real multi-robot system if all the following conditions are true:

1. a component datasets includes observations of a region of space R over a time interval $T = [t_1, t_2]$;
2. another component dataset includes observations, over the time interval $T' = [t'_1, t'_2]$, of another region R';
3. the intersection $RR' = R \cap R'$ is not empty;
4. one or more objects move within RR' during a non-empty time interval $TT' = [\ \max[t_1, t'_1],\quad \min[t_2, t'_2]\ ]$, i.e., the intersection of T and T'.

This condition, though not easy to check, is a mean for building consistent multi-robot datasets. Notice that the special case of the condition is *multi-robot datasets where a component dataset features the robot passing a region of space, in a given time interval T1, while the same region is featured in another component dataset in time interval T2, and the intersection T1 ∩ T2 is not null, can be detected as being not realistic*, and consequently cannot be used to build a multi-robot BP. This still leaves ample possibilities to define multi-robot BPs: in fact the amount of time shift that each component dataset is subject to can be tweaked to avoid triggering the condition.

Sensor data

The sensor datasets useful for these BPs are all datasets available: indoor, mixed, outdoor. They can be reached from the project website (www.rawseeds.org). For a detailed description of the areas covered during each dataset collection, the sensors used, etc., please refer to the other relevant project deliverables (D1.1, D1.2, D2.1, and D2.2).

Ground truth

For a more accurate description of the GT systems used in RAWSEEDS, i.e., poseGT and mappingGT, please refer to the other relevant project deliverables (AD2.3, D2.1 and D2.2), and to the previous section about this topic.

Robot logs

Logs of all robot motion and sensor data are part of the datasets, therefore, please refer to the relevant project deliverable (D2.1 and D2.2) as well as to the project website.

Specifications of data formats

Data formats for the sensor streams are described in detail with the online datasets; therefore, please refer to the project website as well as to the relevant project deliverable (D2.1 and D2.2) for what concerns this aspect. On the other hand, data formats for the BSs will be described in the related project deliverable, from WP5 (D5.2). We only mention here that it will be difficult to define, a priori, formats for the maps. One option is to ask the BS to include both a detailed description of the format of the output file as well as an email of the corresponding BS author, for assistance in decoding it. On the other hand, we can easily define formats for the trajectories, and also for the associations between mappingGT features and reconstructed features. While the first can be expressed in the usual 3DoF set of coordinates, the latter can follow what is mentioned in the description of the ME measure about the pairs of reconstructed and mappingGT features, i.e., just a list of pairs of id, for each feature.

Specification of the evaluation criteria

For the SLAM BPs the performance evaluation parameters that we are proposing are:

- ME (for the selected features in the mappingGT);
- ATE (for the reconstructed poses where we have the poseGT);
- RCE;
- SLE (for the reconstructed poses where we have the poseGT);
- RPE (for the reconstructed poses where we have the poseGT).

ME and ATE are performance measures that evaluate the absolute map quality and trajectory, which is good, but does not matches completely the RAWSEEDS approach to performance evaluation. On the other hand, RAWSEEDS aims to performance measures that are a direct consequence of the usage of a given BS output. In the case of SLAM, as a SLAM algorithm produces a map, we would like to measure the effectiveness of such map for typical mobile robotics tasks. The task we identified for the evaluation of the SLAM BP is self-localization. Other tasks might also be appropriate, but self-localization is interesting because essential for navigation tasks. For this reason we introduce, as a measure of performance for the SLAM BPs, a performance measure of self-localization (SLE). As we want to mimic the pattern a real robot would need to follow, i.e., using a map for navigating an environment, after having obtained a map of it. This map, in our situation

characterized by the usage of offline datasets, is the one that has been built by applying the SLAM algorithm to the dataset that is defining the BP instance (*slamdataset*). This map, i.e., the algorithm that produced it, is then evaluated by self-localizing the robot in it; this is obtained by using sensor streams from different datasets, though taken in the same location where the *slamdataset* was taken.

Alignment of mappingGT and reconstructed maps

Both the absolute trajectory error (ATE) and our usage-based measure SLE depend on the availability of the roto-translation of the first-pose of the dataset used for generating the map, with respect to the GTframe, which is the frame to which the poseGT is referred.

Notice that there is no dependence on the first pose of the dataset from which the sensor streams used for self-localization are taken, for SLE. Such dependency is not there because the self-localization algorithm will naturally provide a reconstructed pose that is expressed in the frame of the provided map (*slamdataset*), and not in the frame of the dataset from which the sensor stream(s) are taken.

We shortly review hereafter a few possible steps involved in the determination of this roto-translation.

- if the first pose of dataset is in the GT area, then an estimate of $RT_{1st_{pose}}^{GTframe}$ is available in the BP set of data (in the poseGT stream); as the accuracy of $RT_{1st_{pose}}^{GTframe}$ is affecting the performance measure, we suggest to take this value as just a first estimate;
- if the first pose is not in the GT area, and/or to refine the first estimate, then the author of the BS has to determine the $RT_{1st_{pose}}^{GTframe}$, basing on her/his preferred method, and use it for referring the BS map and poses to the GTframe; the author has also to provide an exhaustive explanation of how she determined the value; a suggestion on how to execute such a task is to perform an ICP-like alignment between the mappingGT and the reconstructed map, basing on the same features selected for computing the ME measure; notice also that such work might collapse into a simple "*usage of the roto-translation provided by the preceding xyz BS author*";
- for SLE, as the poses output by the self-localization algorithm are naturally referred to the first pose of the dataset used for building the map (*slamdataset*), and we know the $RT_{1st_{pose}}^{GTframe}$, we can move the poses output by the self-localization algorithm to the GTframe and compare with the poseGT;
- the performance measures (ATE and SLE) can then be computed for each such dataset.

Table summarizing the proposed BPs

	PROBLEM	SENSOR DATA	GROUND TRUTH	EVALUATION MEASURES
Laser SLAM	perform a map building activity with SLAM (online)	laser, IMU and odometry from a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE
Monocular SLAM	perform a map building activity with SLAM (online)	single camera, IMU and odometry from a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE
Stereo SLAM	perform a map building activity with SLAM (online)	stereo camera, IMU and odometry froma a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE
Trinocular SLAM	perform a map building activity with SLAM (online)	trinocular IMU and odometry from a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE
Omnidirectional vision SLAM	perform a map building activity with SLAM (online)	omnidirectional vision, IMU and odometry from a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE
Sonar SLAM	perform a map building activity with SLAM (online)	sonar sensors, IMU and odometry from a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE
Multisensor SLAM	perform a map building activity with SLAM (online)	streams from more than one sensor, for a dataset	mappingGT; poseGT	ME, ATE, RCE, SLE, RPE

There are currently 11 validated datasets, from indoor, mixed and outdoor locations (Bicocca and Bovisa); those are combined with the defined BPs, giving rise to a quite large number of BP instances, more than 60.

List of items constituting a BS

A BS is built of some pieces of information, which so far have been introduced and described in a dispersed way. We summarize hereafter the list of items that are required for the submission, for the benefit of the potential submitter.

1. title of the BS;
2. author(s) of the BS;
3. contact details (including email) of the corresponding author as well as of a faculty persone, in case the corresponding author is a temporary person (the idea behind this is to be able to reach the authors even years after the submission);
4. BP instance to which the BS applies;
5. document describing the algorithm, the level of detail is such that it must allow replication; thsi document have to describe also the settings used in the BSs;
6. (not mandatory) source code of the algorithm;
7. full output of the algorithm;
8. document describing the format of the full output; this document should allow anybody to read and interpret the submitted output; this is important for replicating the results;
9. reference to the mappingGT used for the ME measure;
10. map estimate (list of features) to be used for the ME performance evaluation; the format is the same as for the mappingGT features ($\boldsymbol{x}_i = [\ i\ x_i\ y_i\ ]$);
11. list of the associations between the mappingGT features and the reconstructed features, this is just a list of pairs of *id*s, an *id* for the mappingGT feature, an *id* for the reconstructed feature;
12. complete description of the procedure followed to determine, from the originally reconstructed features, the ones used in the performance evaluation, in case the BS output is not homogeneous to the mappingGT;
13. trajectory (list of poses) specified in the expected format (i.e., the usual 3DoF set of coordinates $<$ *time-stamp,* $[x_j, y_j\ \theta_j] >$, at each time-stamp; if the algorithm under evaluation is not able to provide a pose at such frequency, then an interpolation have to be provided;
14. document with description of the interpolation above, accurate enough to allow replication;
15. document with the explanation of how the $RT_{1stpose}^{GTframe}$ value has been determined;

Conclusions

We presented a few Benchmark Problems, i.e., class of problems and rating methodologies, which are then instantiated in their union with the RAWSEEDS datasets.

The performance measures that we developed concern the evaluation of SLAM algorithms. Some measure aim at rating the absolute quality of the output of the SLAM algorithms, while one follows what we call the "usage-based" idea, i.e., to measure the performance of the SLAM algorithm by running, on its output, some another algorithm, for solving another mobile robotics problem.

We then very shortly introduce the SLAM BPs as well as other potentially interesting BPs, for future developments. We finally describe how the performance measure have be applied to the output of an algorithm, in order to obtain data for the submission of a BS to the RAWSEEDS list on the web.

References

[1] Y Bar-Shalom, X. R. Li, and T. Kirubarajan, "Estimation with Application to Tracking and Navigation", Jonh Wiley and Sons, 2001

[2] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans", in IEEE Computer Vision and Pattern Recognition Conference (CVPR), pages 935–938, 1994