Realizing, Reversing, Recovering : Incremental Robust Loop Closing over time using the iRRR algorithm

Yasir Latif, César Cadena, and José Neira

Abstract— **The ability to detect failures and reconsider information over time is crucial for long term robust autonomous robot applications. This applies to loop closure decisions in localization and mapping systems. This paper describes a method to analyze all available information up to date in order to robustly remove past incorrect loop closures from the optimization process. The main novelties of our algorithm are: 1. incrementally reconsidering loop closures and 2. handling multi-session, spatially related or unrelated experiments. We validate our proposal in real multi-session experiments showing better results than those obtained by state of the art methods.**

I. Introduction

A mobile robot can use a place recognition system to find loop closure constraints that help improve the estimation of its pose in the environment and the precision of the model of the environment being built. Any failure in the loop closure process will surely corrupt the state estimation. The robot is expected to continue exploring the environment, even with the corrupt estimation, gathering more information that allows it to realize any failures and to recover the accurate state estimation. No matter how robust a place recognition system might be, there always exists the possibility of getting just one false positive loop closing constraint. That possibility increases in long term operations and in persistent mapping tasks, where failures may happen over time. But with the arrival of new evidence, the estimation can be corrected. Only by collecting more evidence over time can we detect and correct these mistakes in loop closing.

In this work we propose an incremental algorithm to **r**ealize that the place recognition system has generated wrong constraints, **r**emove them if necessary, and **r**ecompute the state estimation. The proposal is based on evaluating the consensus among constraints with two main novelties: (1), making the best possible decision according to the information provided up to date by the place recognition and odometry systems; and (2), the ability to decide over loop closures between different session without requiring connectivity through weak links.

Our method works with the pose graph formulation. It is a part of the back-end of a SLAM system and is therefore independent of the type of sensor used for odometry or loop closing. All our method needs is a system that is able to generate a pose graph for the sequential pose constraints and a place recognition system for the non-consecutive loop closure constraints.

In the next section we discuss relevant work related to this subject, and highlight the need for a robust loop closing over time method. In section III we detail our proposal and carry out real experiments in section IV. Finally, in section V discussion and conclusions about the work are presented along with possible future work.

II. Related Work

Several approaches to multi-session SLAM have been presented in robotics literature in the recent past. Konolige and Bowman [1] presented a stereo visual odometry system that works together with a bag of words place recognition system towards building multiple representations of dynamic environments over time. Multiple map stitching, recovery for odometry failures, loop closing, and global localization rely on the robustness of the place recognition system. "Weak links" are removed when place recognition is able to close loops, making it prone to errors when the place recognition closes loops wrongly. Similarly, McDonald et al. [2] presented a multi-session 6 DOF Visual SLAM system using "anchor nodes". In their approach, place recognition is assumed to be perfect and its output is trusted every time. Sibley et al. [3] presented a relative bundle adjustment approach for large scale topological mapping. They show an example of mapping from London to Oxford, over a trajectory of about 121-km. They also use appearance based place recognition and are therefore also in danger of making wrong decisions in case of incorrect place recognition. All of these approaches to large scale and persistent mapping rely on a place recognition system with zero false positives. This is usually achieved at the cost of a high false negative rate, and thus loss of precision. But even the most robust place recognition systems [4], [5] cannot guarantee 100% accuracy.

Some approaches delay decision making and maintain multiple topologies of the map with an associated belief for each one [6], [7]. Ranganathan and Dellaert [6] follow this approach using a Rao-Blackwellized particle filter. Their method is also unique in the sense that it uses the estimation process itself to reason about possible loop closures. Tully et al. [7] also maintain multiple hypotheses with a forest expansion considering all possible loop closures. However, those approaches do not explicitly show how their system is affected by and recovers from wrong loop closures. Tully et al. favour the smaller graph in cases of perceptual aliasing inducing to collapse it and not recover it from the failure.

Yasir Latif, César Cadena, and José Neira are with the Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Zaragoza 50018, Spain. {ylatif, ccadena, jneira}@unizar.es.

This research has been funded by the Dirección General de Investigación of Spain under projects DPI2009-13710 and DPI2009-07130.

A similar estimation-based reasoning approach using pose graph formulation was presented by Sunderhauf and Protzel [8] which is a robust SLAM back end using "switch factors". The central idea is to penalize those loop closure links during graph optimization that deviate from the constraints they suggest between two nodes. Similar to our approach, they change the topological structure of the graph based on identification and rejection of wrong loop closures. In contrast, however, they assess the validity of every loop closure on its own, without forming a general consensus using all the available information. In cases where there are a number of hypotheses that suggest the same but wrong loop closings (for instance due to perceptual aliasing in long corridors), overall consensus helps us in rejection of such outliers. Their method suggests a continuous function governing the state of "switch factors" which does not make sense in most of the cases like traversal paths. We show comparisons against their method in the experimental section.

III. Our proposal

We propose a robust consistency-based loop closure verification method using the pose graph formulation. It is based on the observation that correct loop closure in conjunction with odometry can help in the detection of wrong loop closures. Our method follows the line of work in which the estimation process itself is used in making the distinction between correct and false loop closures.

The graph based formulation for SLAM, the so-called "graph-SLAM" models robot poses as nodes in a graph where relative transformations from odometry or loop closures form edges or "constraints". Let $\mathbf{x} = (x_1 \dots x_n)^T$ be a vector of parameters that describe the configuration of the nodes. Let ω_{ij} and Ω_{ij} be the mean and the information of the observation of node j from node i. Given the state $\mathbf{x}$, let the function $f_{ij}(\mathbf{x})$ be a function that calculates the perfect observation according to the current state. The residual r_{ij} can then be calculated as

$$r_{ij}(\mathbf{x}) = \omega_{ij} - f_{ij}(\mathbf{x}) \tag{1}$$

Constraints can either be introduced by odometry which are sequential constraint ($j = i + 1$), or from a place recognition system which are non-sequential. The amount of error introduced by each constraint weighed by its information can be calculated as

$$d_{ij}(\mathbf{x})^2 = r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) \tag{2}$$

and therefore the overall error, assuming all the constraints to be independent, will be:

$$D^2(\mathbf{x}) = \sum d_{ij}(\mathbf{x})^2 = \sum r_{ij}(\mathbf{x})^T \Omega_{ij} r_{ij}(\mathbf{x}) \tag{3}$$

The solution to graph-slam problem is to find a state $\mathbf{x}^*$ that minimizes the overall error. Iterative approaches such as Gauss-Newton or Levenberg-Marquadt can be used to compute the optimal state estimate [9].

We can divide the constraints into two sets; S containing sequential links and R containing loop closure links. Since all constraints are independent of each other, the error in 3 can be written as

$$D^2(\mathbf{x}) = \sum_{(i,j)\in S} d_{ij}(\mathbf{x})^2 + \sum_{(i,j)\in R} d_{ij}(\mathbf{x})^2 \tag{4}$$

We can further divide the set R into n disjoint subsets R_k, where each subsets only contains topologically related constraints (links that relate similar portions of the robot trajectory) such that $R = \cup_{k=1}^{n} R_k$ and $\forall (i \neq j) R_i \cap R_j = \emptyset$. We term each of theses subsets as "clusters".

Then the error for set R can be written as

$$\sum_{(i,j)\in R} d_{ij}(\mathbf{x})^2 = \sum_{c=1}^{n} \sum_{(i,j)\in R_c} d_{ij}(\mathbf{x})^2 = \sum_{c=1}^{n} d_{R_c}(\mathbf{x})^2 \tag{5}$$

where $d_{R_c}(\mathbf{x})^2$ is the error contributed by the cth subset. This simply means that the overall error introduced due to loop closure constraints is the sum of the individual errors of each cluster.

Assuming that initially we do not have any outliers in odometry (if there are errors in odometry the front-end can detect them and split the experiment in two sessions), the error in (3) is caused practically only by the loop closing links. Once we iterate to find the optimal state, the error in the odometry is no longer zero. This increase in odometry error gives us a measure of the change in the topology that must take place in order for the graph to conform to loop closure constraints. This error will be smaller when the corresponding loop closures are correct because of the comparatively smaller change needed in topology of the graph as opposed to the change needed when loop closures are wrong. Moreover, clusters that suggest the same change in topology would cause smaller errors among them as compared to clusters that suggest different changes in topology. By measuring how much errors the clusters introduce, we can detect which clusters agree with each other. A point to note here is that even though odometry drifts with time, it is still a useful measure of the underlying topology of the graph.

A. Method

1) Clustering: Our method starts by collecting topologically related loop closing hypotheses into clusters. Clusters are sets of loop closure links that relate similar portions of the trajectory. We use a simple incremental way to group them related in time and in space, given that each pose has an associated timestamp. We proceed as follows: with the first loop closure that arrives, we initialize the first cluster R_1. Then, we decide according to (6) if the next loop closure arriving belongs to same cluster or a new cluster needs to be initialized:

$$\begin{aligned} &\omega_{i,j} \in R_k \iff \\ &\quad \exists \omega_{p,q} \in R_k \;\mid\; \|t_i - t_p\| \leq t_g \wedge \|t_j - t_q\| \leq t_g \end{aligned} \tag{6}$$

where t_i means the timestamp related to the node i, and t_g can be selected according to the rate at which place recognition system runs. This threshold defines the cluster neighbourhood. In our experiments. We consider loop closing hypotheses that are less than 10 second apart to be part of the same cluster. A cluster is considered closed if after t_g time no more links are added to it. The completion of a cluster triggers our algorithm.

2) Cluster-wise individual compatibility: After clustering hypotheses together, the next step is to compute the individual compatibility for each cluster. This involves optimizing the pose graph with respect to just this single cluster and checking which links satisfy the corresponding χ^2 bound. The links inside the cluster that do not pass this test are removed from the cluster and are no longer considered in the optimization. Algorithm 1 describes the cluster-wise individual compatibility. This procedure is carried out for each cluster as soon as it is closed.

Algorithm 1 Cluster_IC

Input: $poses$, $slinks$, $cluster$ of $rlinks$
Output: $cluster$
  add $poses, slinks$ to PoseGraph
  PoseGraphIC $\leftarrow$ PoseGraph
  add $cluster$ to PoseGraphIC
  optimize PoseGraphIC
  if $D_G^2 < \chi^2_{\alpha,d_G}$ **then**
    for each $rlink_l \in cluster$ **do**
      if $D_l^2 < \chi^2_{\alpha,d_l}$ **then**
        Accept $rlink_l$
      else
        Reject $rlink_i$
      end if
    end for
  else
    Reject $cluster$
  end if

3) The RRR algorithm: Having established individual compatibilities, we now look for clusters that are mutually consistent. We initially assume that all the clusters present in the optimization process are consistent and carry out optimization by including all of them in the optimization motor. Once optimized, we check for any links whose residual error satisfies the χ^2 test. The clusters to which these links belong are then selected and added to the candidate set to be evaluated for joint compatibility. Joint compatibility in this case implies that the Mahalanobis distance contributed by the clusters and the overall Mahalanobis distance of the optimized graph should be less than their corresponding χ^2. This is shown in algorithm 2.

We accept the clusters that are jointly compatible and term them as the *good set*. At this point, we remove the good set from the optimization and try to re-optimize with the remaining clusters. The idea behind doing so is that, in the absence of the good clusters, other correct clusters will be able to pass the χ^2 tests. As long as we keep finding clusters that are jointly compatible with the good set, the good set will grow.

Algorithm 2 JC

Input: $goodSet$, $candidateSet$, PoseGraph
Output: $goodSet$, $rejectSet$
  PoseGraphJC $\leftarrow$ PoseGraph
  add $(goodSet, candidateSet)$ to PoseGraphJC
  $rejectSet \leftarrow \{\}$
  optimize PoseGraphJC
  if $D^2 < \chi^2_{\alpha,d} \wedge D^2_{all} < \chi^2_{\alpha,d_{all}}$ **then**
    $goodSet \leftarrow \{goodSet, candidateSet\}$
  else
    find the $cluster_i \in candidateSet$ with largest CI
    remove $cluster_i$ from $candidateSet$
    $rejectSet \leftarrow cluster_i$
    if $\neg$isempty $candidateSet$ **then**
      $(goodSet, rSet) \leftarrow JC(goodSet, candidateSet)$
      $rejectSet \leftarrow \{rejectSet, rSet\}$
    end if
  end if

Algorithm 3 RRR

Input: $poses$, $slinks$, $\mathcal{R}$ set of clusters containing $rlinks$
Output: $goodSet$ of $rlinks$
1: add $poses, slinks$ to PoseGraph
2: $goodSet \leftarrow \{\}$
3: $rejectSet \leftarrow \{\}$
4: **loop**
5: PoseGraphPR $\leftarrow$ PoseGraph
6: $currentSet \leftarrow R \backslash \{goodSet \cup rejectSet\}$
7: $candidateSet \leftarrow \{\}$
8: add $currentSet$ to PoseGraphPR
9: optimize PoseGraphPR
10: **for** each $cluster_i \in currentSet$ **do**
11: **if** $\exists D_l^2 < \chi^2_{\alpha,d_l} \mid rlink_j \in cluster_i$ **then**
12: $candidateSet \leftarrow \{candidateSet, cluster_i\}$
13: **end if**
14: **end for**
15: **if** isempty$(candidateSet)$ **then**
16: STOP
17: **else**
18: $s = goodSet.size$
19: $(goodSet, rSet) \leftarrow JC(goodSet, candidateSet)$
20: **if** $goodSet.size > s$ **then**
21: $rejectSet \leftarrow \{\}$
22: **else**
23: $rejectSet \leftarrow \{rejectSet, rSet\}$
24: **end if**
25: **end if**
26: **end loop**

An important point to mention here is the use of the *reject set*. The reject set contains all the clusters that we checked in the last iteration but found them to be incompatible with the good set. We omit them from the optimization process until something is added to the good set in the next iteration. The idea behind doing so is the following: during the previous iteration of the algorithm, we found some clusters to satisfy the threshold test, and therefore we evaluated their compatibility with the good set. We found some of them to be incompatible and they were added to the reject set. If no changes were made to the good set and we include the clusters from the reject set in the optimization again, the algorithm would suggest the same clusters for

evaluation but we already know them to be inconsistent. The reject set therefore acts as an accumulator of clusters that are inconsistent with the present good set. The reject set is therefore maintained until something is added to the good set and is cleared once something is appended to the good set. The main algorithm is given in the algorithm 3.

The algorithm terminates when we are no longer able to find any clusters to add to the candidate set.

B. Incremental Implementation

The method proposed can be carried out in an incremental fashion, it being triggered every time we close a cluster. We calculate the individual compatibility for this cluster. If it is not individually compatible, it is discarded and nothing further is done, otherwise we execute an incremental version of algorithm 3.

The incremental version of algorithm 3 works this way: for the first cluster that passes IC, we begin with an empty goodSet and rejectSet (line 2-3), otherwise values evaluated at the previous step are used. It should be mentioned that the set R contains only the clusters that have passed the IC test. The loop (line 4-26) is then carried out with the following changes: in JC (line 19), rather than considering only the members of candidateSet for elimination, we consider the member of goodSet as well. This is to ensure that if further evidence become available against member of goodSet, they can be eliminated. Since we are building a consensus gradually over time, anything that is rejected by JC is inconsistent. Therefore we do not clear the rejectSet in the incremental implementation (line 20-24). Finally, once we have calculated the goodSet, we optimize the graph with it, giving us the best estimate at the current point in time.

C. Multi-Session

Our method is able to work with multiple sessions without needing "weak links" or "anchor nodes". We argue that if there is a correct loop closure between two sessions, this information alone is enough to align the two sessions correctly after optimization. But this information may or may not be available or may arrive in the system after a long time. Therefore, we need to deal with multiple session in a unified manner, so that if the loop closing information between different sessions becomes available, we can align the sessions, otherwise we maintain up-to-date estimate of all the sessions independently.

In order to achieve this, we make the observation that for n connected sessions, the $goodSet$ denoted by G^t at time t can be partitioned into n subsets such that $G^t = \cup_{i=1}^{n} G_i^t$ and $\forall i \neq j : G_i^t \cap G_j^t = \emptyset$. Since there are no clusters that link these sets, they are independent of each other, and we can calculate/expand the G_i for each subset independently. Similar partitioning can be done for the $candidateSet$. Once we compute/update the required G_i^{t+1} we combine all the subsets to form the new G^{t+1}. Mathematically,

$$G^{t+1} \leftarrow iRRR(G^t, C^t) = \cup_{i=1}^{n} iRRR(G_i^t, C_i^t) \quad (7)$$

where C^t is the corresponding $candidateSet$, at time t.

This formulation helps us in detecting loop closing clusters that connect two different sessions. As soon as such a cluster forms a part of the $goodSet$, we have the up-to-date information on how the two sessions, linked by this cluster, are related to each other. This formulation also ensures that only the session to which new loop closing clusters are being added are updated, because for the other sessions the C_i^t would be empty, indicating that we can not reason about them given this new information.

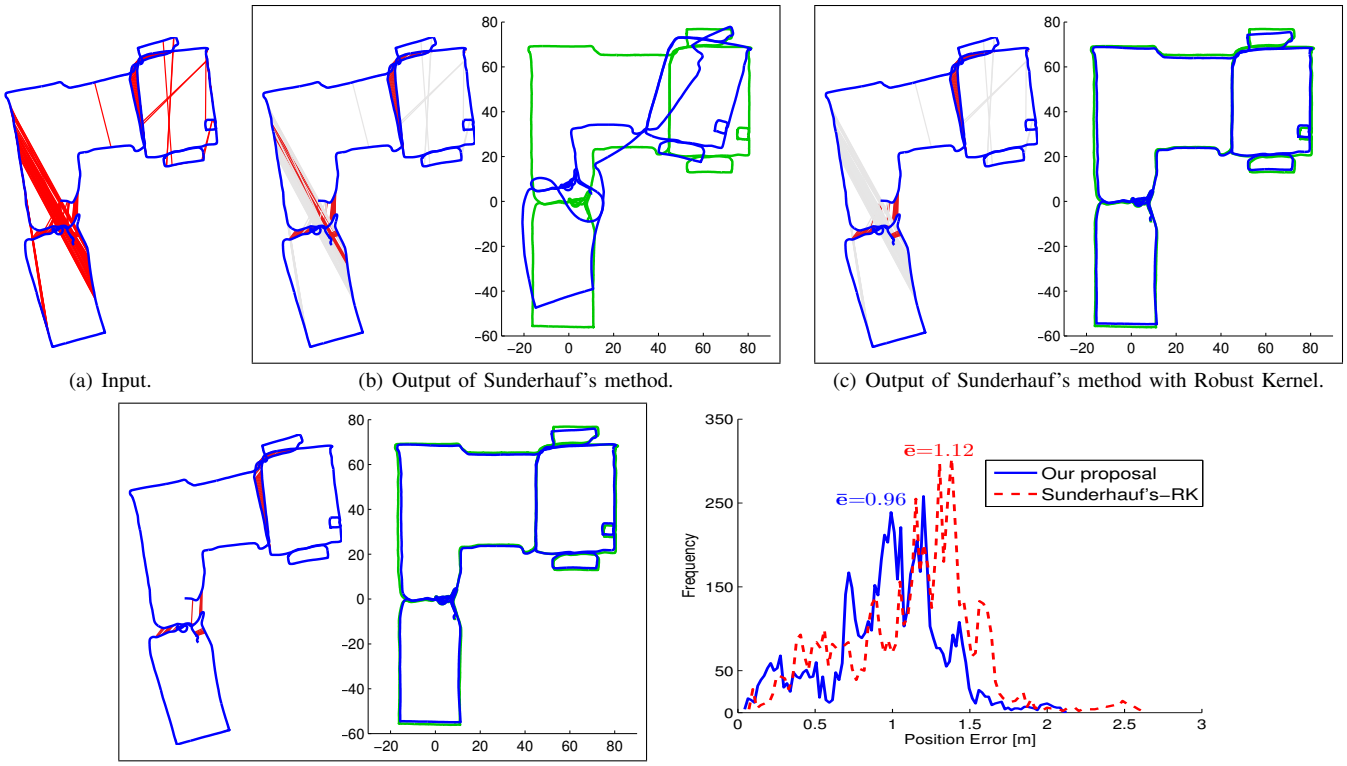
IV. Experiments

In this section, we present the result of experiments for our proposal using multiple and diverse datasets. We use the popular Intel dataset ($\sim$420m) collected at the Intel Research Laboratory (Seattle, WA), which is available with the downloaded version of g2o. Also, we use the New College dataset ($\sim$2.2km), an outdoors environment from [10], and a multi-session dataset ($\sim$4km) in an indoor environment (Bicocca campus) from the RAWSEEDS project [11]. Our method can use any graph optimizer (e.g. iSAM [12] or g2o [9]), for this work we use g2o configured with Gauss-Newton method and four iterations.

A. Comparisons

We first compare in batch mode our method (i.e. RRR algorithm) against our own implementation of Sunderhauf's approach to robust back-end [8] using the parameters and functions for switch links as given in their paper. The optimizer used was g2o as well, with two configurations: Levenberg-Marquadt and Levenberg-Marquadt with Huber Robust Kernel (RK) width=0.1. We evaluate both configurations against our method in a batch mode over one indoor session (Bicocca Feb 25b) with a laser scan matcher as odometry and a bag-of-words plus stereo geometrical checking as place recognition system. The results are shown in Fig. 1. Sunderhauf's method accepts false positive loop closures in the zones where there are several contiguous links because they all weigh in the same direction in the optimization and for this reason the switch links are not able to reduce their effect on the graph. The same method with RK works much better, but given that their method never completely rejects the wrong loop closures, the accuracy of the estimation is affected. In Table I we show he Absolute Trajectory Errors (ATE) computed using the *Rawseeds Metrics Computation Toolkit* provided by the RAWSEEDS Project. Also, we show the computational time required in each case.

Our proposal efficiently keeps only the loop closures that form the majority consensus among them and with the odometry constraints. It takes half of the time compared to Sunderhauf's method and the accuracy is better than 1m, 0.12% of 774m of travel in this session. The ATE distribution is shown in Fig. 1(e) for our method and for Sunderhauf's with RK activated.



(a) Input. (b) Output of Sunderhauf's method. (c) Output of Sunderhauf's method with Robust Kernel.

(d) Output of our proposal. (e) Absolute trajectory errors distributions.

Fig. 1. **(a):** One of the sessions from Bicocca campus shown in Fig. 2(a)(bottom) with laser odometry and the constraints from the place recognition system. **(b):** The result of Sunderhauf et. al [8], **(c)** with Robust Kernel activated, *left:* the loop closures, light to dark red proportional to the value taken by the switch factors, *right:* optimized graph (blue) and ground truth (green). **(d):** The result of our proposal, *left:* the final loop closures in the good set, *right:* optimized graph (blue) and ground truth (green). **(e)** the distribution of the Absolute Trajectory Errors for the final pose graphs against the ground truth (c,right) and (d,right).

	Mean (m)	Std. Dev (m).	Time (sec)
Sunderhauf's	14.72	14.00	20.9
Sunderhauf's RK	1.120	0.430	18.7
Our method	**0.964**	**0.367**	**10.1**

TABLE I

SUMMARY OF RESULTS FOR FIG. 1.

B. Persistent Mapping

The task of persistent mapping involves robustly maintaining and when information becomes available joining multiple sessions into a single map.

For the Intel dataset, g2o gives us an optimized graph with sequential constraints and true loop closures. In order to use it as multi-session and to evaluate our proposal, we split the dataset in four sessions after each loop in completed deleting the corresponding odometry constraint. The first pose in the resultant session is assumed to be at the origin and we transform each session according to that. Finally, we corrupt the dataset with 600 wrong loop closures, grouped in 200 randomly generated clusters of 3 links each. In Fig. 2(a)(top) we show the resultant sessions with the original and generated loop closures.

The New college dataset provides laser scan and images. We use a precomputed visual odometry[1] and again, we split this odometry in three session deleting one of the sequential constraints and changing the reference frame for each session. We obtain the loop closure constraints with a bag-of-words place recognition system, as described in [5], plus a stereo geometrical checking (BoW+gc). In Fig. 2(a)(middle) we show the resultant sessions with the detected loop closures.

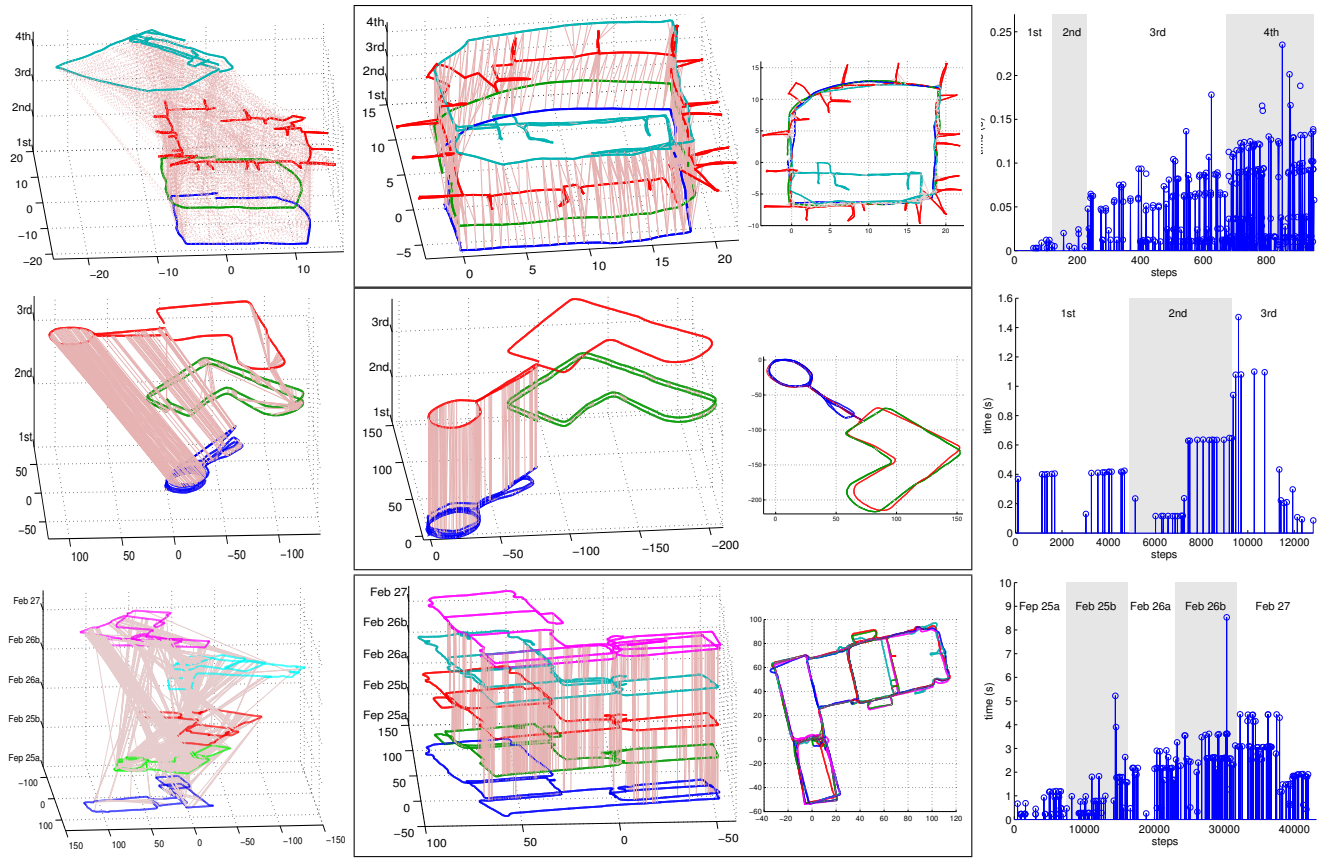
The rawseeds dataset contains five overlapping runs in indoor environment and provides wheel odometry, laser scans and images from a camera mounted on the robot. We use laser scans from the dataset to compute the laser odometry using a simple scan matcher and use the BoW+gc to compute the loop closure constraints. Each session starts with its own origin. In Fig. 2(a)(bottom) we show the laser odometry for the five sessions with the detected loop closures.

The results for the three datasets are shown in Fig. 2(b). It can be seen that we have recovered the correct loop closing as well as the relationships between the different sessions. In Fig. 2(c) we show the computational times required by the iRRR algorithm vs. the step when is triggered. For C clusters, our method needs to carry out C individual compatibility tests. More over, in the worst case if all the clusters are considered for joint compatibility after the first optimization and are not jointly compatible, we need C joint compatible checks. This makes our method linear $O(C)$ in the number of clusters which agrees with the linear behaviour shown in Fig. 2(c).

V. DISCUSSION

In this paper, we have presented a method of robustly solving the loop closing problem in a multi-session context.

[1]Available at `http://www.robots.ox.ac.uk/NewCollegeData/index.php?n=Main.Downloads`



(a) Inputs. (b) Output of our iRRR algorithm. (c) Computational time.

Fig. 2. Multi-session experiments. (a) We show the inputs, an odometry (each session is a different color and height) and loop closures (pink color). (b) The output of our proposal, each session on different height (left) and the floor-view (right). On top, the Intel dataset divided by us into four sessions and corrupted by 600 wrong loop closures. In the middle, the NewCollege dataset with using visual odometry divided into three sessions and with a BoW+gc place recognition system. Bottom, the Bicocca multi-session experiment from the RAWSEEDS project, odometry was computed by a laser scan matcher and the loop closures by a BoW+gc place recognition system. Each session is in its own frame of reference. (c) We show the computational time spends by our proposal when is triggered.

Our method is able to generate the best estimate of the possibly multiple maps with all the information available until the present moment simultaneously solving for the transformation relating the sessions.

We compare our method against the one proposed by [8] and we show that our method is able to deal with loop closing robustly. The aim of SLAM is not just to construct a good looking map, but the map should be usable for high level tasks such as path planning. In that context, loop closing links provide traversability information and the robot can take these path into account in order to calculate a path. "Switch factor" may provide a good estimate of the map but since they are governed by a continuous function they allow soft decision making with regards to loop closings decisions, which is in principle a boolean decision. In some cases, as has been shown in the comparison section, links are partially disabled, which is not a desirable effect.

Previously methods such as anchor nodes or weak links have been proposed to deal with the problem of connectivity in multiple sessions. Weak links while providing connectivity also introduce unnecessary uncertainty into the optimization process. Anchor nodes on the other hand provide linkage when there is a common observation but they do not show how to deal with errors in data association. We have shown that those two approaches are not required as our proposal can handle those unconnected session without sacrificing performance.

Our method relies on consensus between the loop closing links and assumes that there is only one correct configuration of the graph on which most of the loop closing links agree. The links that do no agree, suggest random configurations. What this means is that we are able to deal with "random conspiracy" among loop closures. The second, more worrying kind is the "organized conspiracy" in which another set of loop closure agree on a different, wrong configuration of the graph. To this end, if everything else is the same, we can trust the odometry links to support the true configuration. Secondly, place recognition systems can be trusted to give either the correct place recognition or random false loop closings, this will prevent an organized conspiracy from forming. Even still, perceptual aliasing can give rise to such conspiracy, as is the case shown for Bicocca Feb 25b.

In order to ensure robustness to such configurations, the incremental implementation optimizes the graph after every

iteration. In the next iteration, this ensures that if the cluster is individually compatible, it will converge fast enough to end up in the candidateSet, but if the cluster is inconsistent with the current goodSet, the chances of it ending up in the candidate set will be less because other clusters will converge faster than it. Secondly, we maintain the rejectSet, which ensures that any cluster that does not agree with the current goodSet is not considered again. While we can not ensure the optimal configuration after each step in case of an organized conspiracy, we can ensure recovery if this becomes clear later on.

The method proposed in this work is applicable to the global metric pose graph formulation. The method is not restricted to any type of sensor for obtaining odometry or loop closures. SLAM algorithms that only generate local consistent maps such as [13] or [14] do not preserve the global geometric relationships between poses and therefore our method can not be applied to them.

We have demonstrated the performance of our algorithm in multiple real cases, in multi-session experiments and compared against the state of the art in robust back-end against false loop closures. Immediate future work consists in considering that odometry links can also include false links. In these cases, these links can be eliminated from the graph, breaking down a single session into a two multi-session case, that can be treated accordingly.

REFERENCES

[1] K. Konolige and J. Bowman, "Towards lifelong visual maps," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 1156–1163.

[2] J. McDonald, M. Kaess, C. Cadena, J. Neira, and J. Leonard, "6-DOF Multi-session Visual SLAM using Anchor Nodes," in *European Conference on Mobile Robotics, ECMR*, 2011.

[3] G. Sibley, C. Mei, I. Reid, and P. Newman, "Vast-scale outdoor navigation using adaptive relative bundle adjustment," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 958–980, 2010.

[4] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0," *The International Journal of Robotics Research*, 2010. [Online]. Available: http://ijr.sagepub.com/content/early/2010/11/11/0278364910385483.abstract

[5] C. Cadena, D. Gálvez-López, J. Tardós, and J. Neira, "Robust place recognition with stereo sequences," *IEEE Trans. Robotics*, 2012, to appear.

[6] A. Ranganathan and F. Dellaert, "Online probabilistic topological mapping," *The International Journal of Robotics Research*, vol. 30, no. 6, pp. 755–771, May 2011. [Online]. Available: http://ijr.sagepub.com/content/early/2011/01/23/0278364910393287.abstract

[7] S. Tully, G. Kantor, and H. Choset, "A unified bayesian framework for global localization and slam in hybrid metric/topological maps," *The International Journal of Robotics Research*, 2012. [Online]. Available: http://ijr.sagepub.com/content/early/2012/01/16/0278364911433617.abstract

[8] N. Sunderhauf and P. Protzel, "Towards a robust back-end for pose graph slam," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2012.

[9] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[10] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, May 2009. [Online]. Available: http://www.robots.ox.ac.uk/NewCollegeData/

[11] RAWSEEDS, "Robotics advancement through Webpublishing of sensorial and elaborated extensive data sets (project FP6-IST-045144)," 2009, http://www.rawseeds.org/rs/datasets.

[12] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental Smoothing and Mapping," *IEEE Trans. on Robotics, TRO*, vol. 24, no. 6, pp. 1365–1378, Dec 2008.

[13] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "Rslam: A system for large-scale mapping in constant-time using stereo," *International Journal of Computer Vision*, vol. 94, pp. 198–214, 2011, 10.1007/s11263-010-0361-7. [Online]. Available: http://dx.doi.org/10.1007/s11263-010-0361-7

[14] H. Strasdat, A. Davison, J. Montiel, , and K. Konolige, "Double window optimisation for constant time visual SLAM," in *IEEE International Conference on Computer Vision (ICCV)*, 2011.