An image–to–map loop closing method for monocular SLAM

Brian Williams[†], Mark Cummins[†], José Neira[⋆], Paul Newman[†], Ian Reid[†] and Juan Tardós[⋆]
[⋆]Universidad de Zaragoza, Spain
[†]University of Oxford, UK

***Abstract*— In this paper we present a loop closure method for a handheld single–camera SLAM system based on our previous work on relocalisation. By finding correspondences between the current image and the map, our system is able to reliably detect loop closures. We compare our algorithm to existing techniques for loop closure in single–camera SLAM based on both image–to–image and map–to–map correspondences and discuss both the reliability and suitability of each algorithm in the context of monocular SLAM.**

I. Introduction

Single–camera SLAM systems (monocular SLAM) have the potential to turn inexpensive cameras into powerful pose sensors for applications such as robotics and augmented reality. However, before they become practical, there are still problems to address. In this paper we focus on the problem of loop closure and the related problem of independent map joining. These issues have of course been discussed previously in relation to mobile robots but the use of only a single camera presents new challenges.

The main difference with hand-held monocular SLAM compared to traditional robotic SLAM systems is that of scale ambiguity. With a lack of odometry and the bearing only measurements provided by a camera, there is an inherent ambiguity in the scale of the map being created. Two maps made of the same region of the world will be at different scales, and so a loop closing algorithm for monocular SLAM must be able to cope with this scale change.

The second challenge associated with hand-held systems is the frequent sudden motion. With a simple rotation, the user can quickly point the camera at an entirely different part of the world. In our previous work [14], we showed how a system could be made more robust to these sudden motions by quickly relocalising the camera relative to the map. However, if the camera is left facing an unmapped region of the world, then it will have to build a new independent map which can later be joined with the original map when an overlap is detected. This process is commonly used in multi-robot mapping [9] when the robots build a map from different starting positions. However, for a hand-held system the process is even more important due to its frequent occurrence. As with loop closure, the overlap detection algorithm must also be able to cope with the relative scale difference between the two maps.

Methods for detecting overlap and loop closure in monocular SLAM can be divided into three categories:

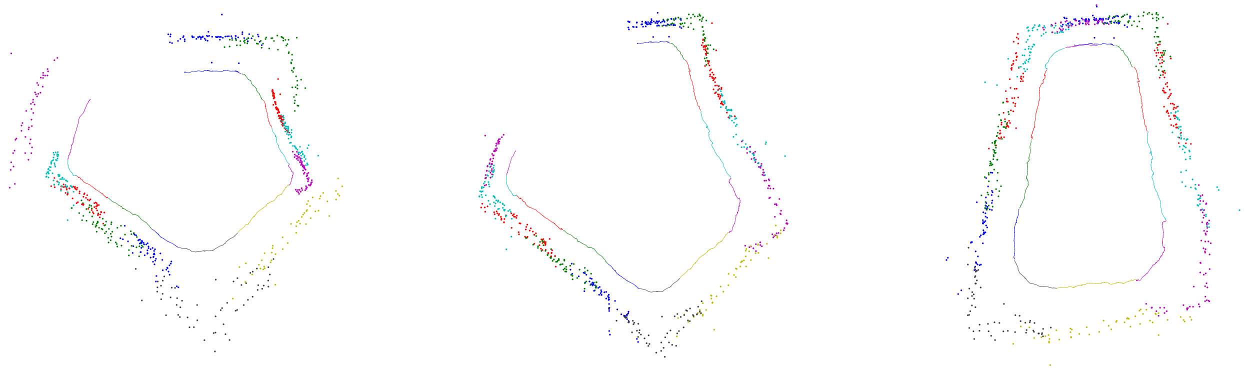
- **Map–to–map** – Correspondences are sought between features in the two maps. Clemente *et al.* [2], applied the variable scale geometric compatibility branch and bound (GCBB) algorithm to loop closing in monocular SLAM. The method looks for the largest compatible set of features common to both maps, taking into account both the appearance of the features and their relative geometric location.
- **Image–to–image** – Correspondences are sought between the latest image from the camera and the previously seen images. Cummins *et al.* [3] have presented an algorithm that uses the occurrences of image features from a standard library to detect that two images are of the same part of the world. Careful consideration is given to the distinctiveness of the features – identical but indistinctive observations receive a low probability of having come from the same place. This minimises false loop closures.
- **Image–to–map** – Correspondences are sought between the latest frame from the camera and the features in the map. This is the method we present in this paper, making use of the relocalisation module we developed previously.

In our previous work on relocalisation [15], we suggested that a possible use of the module would be for map alignment and loop closure. Here we give a description of that process and compare its performance to the other loop closure techniques discussed above.

First we will briefly describe the underlying monocular SLAM system we use. Then we will describe in more detail, the different methods for loop closure. Finally, results will be given on the performance of our algorithm at closing a loop and its performance will be compared to the previously existing methods.

II. The Monocular SLAM System

The monocular SLAM system we use is derived from Davison's original system [4], but with a few improvements to bring it up to date. The underlying system is essentially the same as the system described in [2] but with our own relocalisation module to recover from situations where the system becomes lost. We have also added a system to prevent premature loop closure and added the ability to perform independent map merging. Here we give a brief description of the system, so details of the loop closing system can be better understood.



(a) Local maps obtained with pure monocular SLAM

(b) Local maps auto-scaled

(c) After loop closing

Fig. 1. Map made of a university courtyard. Twelve submaps with a total of 848 features were made during the 70m trajectory. The loop closure was detected using our image–to–map method. Note that the trapezium rather than rectangular shape is consistent with the real courtyard.

A. Map Building

The monocular SLAM system tracks the pose of a hand-held camera while simultaneously building a map of point features in 3D using the EKF. The points are initialised using the inverse depth parameterisation [11], and they are recognised in subsequent frames via normalised cross correlation. An image patch is stored when the feature is initialised, but is warped to correspond with the current camera pose estimate. To speed up the observation of features, the image is only searched in an ellipse given by the uncertainty in the camera and feature estimate in a process called active search. By gating the search in this way the chances of incorrect data association are reduced. This is further helped by the use of the joint compatibility branch and bound algorithm (JCBB) [12] which detects observations which are incompatible with the others and rejects them.

Despite the improvement given by active search and JCBB, there is still a chance of incorrect data association, particularly near loop closures when the system can believe that distant features are again visible and attempt to measure them. If the system is allowed to observe these features as usual, it will likely make incorrect data association due to the large uncertainty in the camera pose relative to these features. Our approach is to prevent the system from making these observations and delay the loop closure until a separate loop close module has detected it (section III). To determine which observations to attempt, we make use of the covisibility data from all the features in the map.

With every set of observations, a tally is updated indicating which features have been successfully observed together. Using this information, a simple graph is constructed where a vertex corresponds to each feature, and the edges indicate those that have been observed together. This graph provides an easy way of determining which features are in the local neighbourhood and which are not. Those which are distant in the graph are not eligible for observation since their relative position to the local features is very uncertain and attempting their observation would likely lead to incorrect data association. Readers should note that another way of determining feature covisibility in a stochastic map is to compute the inverse covariance, the information matrix. Features that have been covisibile at some point will have a high value of co-information.

B. Larger Maps

Due to the accumulation of linearisation errors in the EKF algorithm as well as the increase in update time, we limit our system to quite small local maps (around 70 features). To map larger regions, the Hierarchical SLAM [5] technique is used. This allows the system to map an environment by building a series of submaps, each of which is small enough to allow the system to be run in real-time as well as reducing linearisation errors. This method was already applied to monocular SLAM in [2] but we give a brief summary here.

As each new submap is created, the transformation between its base reference and the previous map is stored in a global state vector. However, for monocular SLAM, this transformation must also include the scale difference which is determined as follows. Each new submap is created with new features initialised at the location of some of the features in the previous map. The geometry of these common features in each submap are used to determine the relative scale. Since the features were newly initialised, information is not shared between the submaps and they remain independent. This scale correction can be seen in Fig. 1(a) and (b).

When loop closure is detected, the global hierarchical map can be updated by adjusting the transformations between submaps in a non-linear constrained optimisation. The result of the optimisation for the loop closure detected by our algorithm is shown in Fig. 1(c).

III. Detecting Loop Closure

In order to close loops in a map, the system must recognise when it has returned to a previously mapped region of the world. Essentially, at this point two regions in the map are found to be the same region in the world even though their position is incompatible given the uncertainty estimate in the map – the classic loop closure problem. The system must then be able to calculate the transformation needed to align these two regions to ‘close the loop’. Finding correspondences between either features or the pose of the camera is usually used to do this. As an extra challenge for monocular SLAM systems, it must also determine the relative scale change between the two mapped regions. Only then can the regions be aligned and the global map adjusted using non-linear constrained optimisation, thus closing the loop and correcting the map.

In the following sections, we describe three methods (including the new method) for achieving these goals based on three quite different approaches. We will later test the performance of all three algorithms.

A. *Map–to–Map Matching: GCBB*

Clemente *et al.* [2] presented a method to close loops in monocular SLAM maps based on finding correspondences between common features in different submaps. The algorithm used is a variable scale version of the original geometric compatibility branch and bound algorithm (GCBB). The system uses both similarity of patch appearance (unary constraints) as well as relative distances between features (binary constraints) to find the largest compatible set of common features between two submaps. Once a consistent set has been found, the relative scale, rotation, and translation needed to align the two submaps can easily be determined.

The system was shown to work in [2] where it found a set of five common features between the first and last submaps in a large loop.

B. *Image–to–Image Matching: Cummins* et al.

Cummins *et al.* [3] have developed a method to detect loop closures based on finding correspondences between the most recent image and previous images seen by the camera. The matching is performed by detecting in each image the presence or absence of features from a visual vocabulary [13] based on SURF features [1], which is learned off line from training data. The system takes into account the probabilities of features appearing together and is able to work out the probability that two images show the same region of the world. This method does not depend on a metric map being created since it only compares images directly. However, it could be used with a metric map if the camera pose relative to such a map can be found for each image as well as the relative pose between two images for the loop closure. Much work has been done on this problem in the field of computer vision [7].

C. *Image–to–Map Matching: New Method*

The new method we have developed for performing loop closure in monocular SLAM is based on our relocalisation module [14]. This module determines the pose of the camera relative to a map of point features by finding correspondences between the image and the features in the map. The pose is then determined from the correspondences using RANSAC and the three–point–pose algorithm [6].

The relocalisation module is able to run faster than framerate through the use of a fast matching algorithm based on the randomised fern classifier [10] where patches are classified through a series of fast pairwise pixel intensity comparisons. While the features are being tracked, each successful observation is used to train the classifier. This classifier is fast but it has a high false positive rate. Incorrect classifications are handled using RANSAC. Details of the randomised fern classifier can be found in [14].

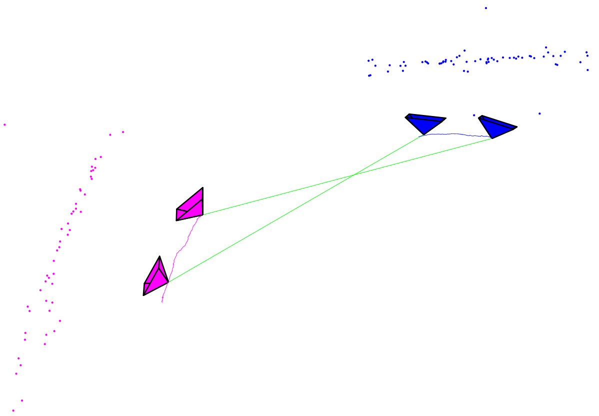


Fig. 2. While tracking in the left map, the system relocalises in the right map using our image–to–map algorithm. The two independent maps are merged by first aligning the common trajectories, and then enforcing the constraint that the two sets of corresponding camera poses (linked by green lines) are equal.

To detect loop closures, the system uses the module to attempt relocalisation in distant regions of the map according to the covisibility graph described in section II-A. When a relocalisation is successful, it gives a correspondence between the current pose being tracked, and the pose given by the relocalisation elsewhere in the map. This gives the translation and rotation needed to align the two regions, but a single pose is not enough to determine the scale difference. To achieve this, the camera is tracked for some time in both regions (while freezing one of the maps so information is not counted twice), and this common trajectory can be used to find the transformation between the two regions including the relative scale difference [8].

When performing independent map merging, it is not sufficient to simply find the transformation between the two maps. We wish instead to combine them into a single correlated map. After transforming map A into the coordinate frame of map B by aligning the trajectories, we are still left with two uncorrelated maps and two estimates of the camera pose ($\hat{\mathbf{x}}^{A}_{cam}$ and $\hat{\mathbf{x}}^{B}_{cam}$).

$$\hat{\mathbf{x}}_{aligned} = \begin{pmatrix} \hat{\mathbf{x}}^{A}_{cam} \\ \hat{\mathbf{x}}^{A}_{map} \\ \hat{\mathbf{x}}^{B}_{cam} \\ \hat{\mathbf{x}}^{B}_{map} \end{pmatrix} \tag{1}$$

$$P_{aligned} = \begin{pmatrix} P_A & 0 \\ 0 & P_B \end{pmatrix} \tag{2}$$

To correlate the maps, we enforce the constraint that the two camera poses are equal. However, since one pose is not sufficient to constrain the scale, we also use the pose at the start of the trajectory, $\hat{\mathbf{x}}^{A,B}_{cam^*}$, which is left in the state vector of each map (Fig. 2). The constraint is enforced through an

EKF update where the ‘observation’ is the difference in these poses.

$$z = \begin{pmatrix} \hat{\mathbf{x}}^{A}_{cam} - \hat{\mathbf{x}}^{B}_{cam} \\ \hat{\mathbf{x}}^{A}_{cam^*} - \hat{\mathbf{x}}^{B}_{cam^*} \end{pmatrix} \quad (3)$$

Afterwards, the maps are correlated and the extra camera poses can be removed from the state.

IV. Results

We have used the monocular SLAM system to build a map of a university courtyard. Due to the size of the environment, the system built twelve submaps as the camera was moved around the 70m trajectory facing the wall. Each new submap was begun by initialising new features in the same image locations as those just observed as the last submap finished. These common features can then be used to fix the relative scale between submaps as shown in Fig. 1.

Even after the scale between submaps has been corrected, the map still exhibits a common problem, that although it has returned to the same region in the world, this is not reflected in the map. A loop closure detection system is needed to recognise that the system has traversed a loop so the map can be corrected accordingly.

We have used all three algorithms to try to detect the loop closure in this sequence. We have also evaluated the performance of the algorithms further by checking their susceptibility to false positives. For a loop closure detection system to be useful, it should be able to be tuned to detect a significant number of true positives while getting very few (ideally zero) false positives. It should also run in a reasonable time for the detection to be relevant.

A. *Map–to–Map Matching: GCBB*

When enough common features exist between two maps, the GCBB algorithm is able to find the correspondences between the maps. This is clear when the algorithm is run between consecutive submaps. It is able to detect the common features that were automatically added. When the system comes to close a loop, it is able to find the common features between the two maps as shown in Fig. 4(a). Unfortunately, during the loop closure, there is no guarantee that the system will have initialised features in the exact same place in two different maps. In fact, in our experiments to date, we have found submaps with sufficient common features to detect the loop closure to be rare. Fig. 3 shows an example of the same frame being tracked in two different maps. Despite the large number of features visible, only two features are common to both maps.

Even getting a corresponding set of features does not guarantee a true correspondence between the two submaps. Fig. 4(b) shows that the GCBB algorithm also found sets of five “common” features between eight other pairs of submaps. All other pairs of submaps were found to have sets of four or three “common” features. We were unable to find a threshold able to distinguish between true positives and false positives for the maps created by our SLAM system. There were simply too few true common features.

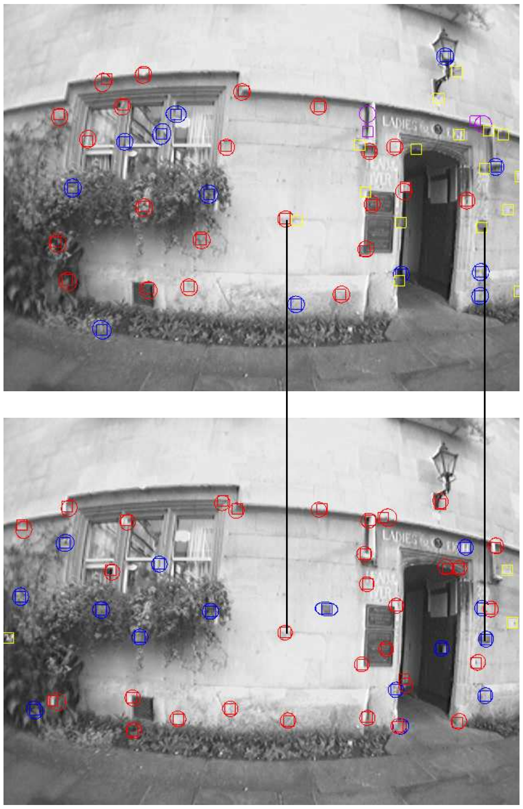


Fig. 3. During the overlap in the sequence, the system tracks the camera in two submaps. The colours indicate if an observation was successful (red), unsuccessful (blue), rejected by JCBB (purple), or not attempted, (yellow). Only two of the visible features are actually common to both submaps.

During our tests, the variable scale GCBB algorithm took around 100ms[1] to compare two maps. When the SLAM system finishes one submap, there is easily time to compare this submap to all previous submaps before the next one is completed.

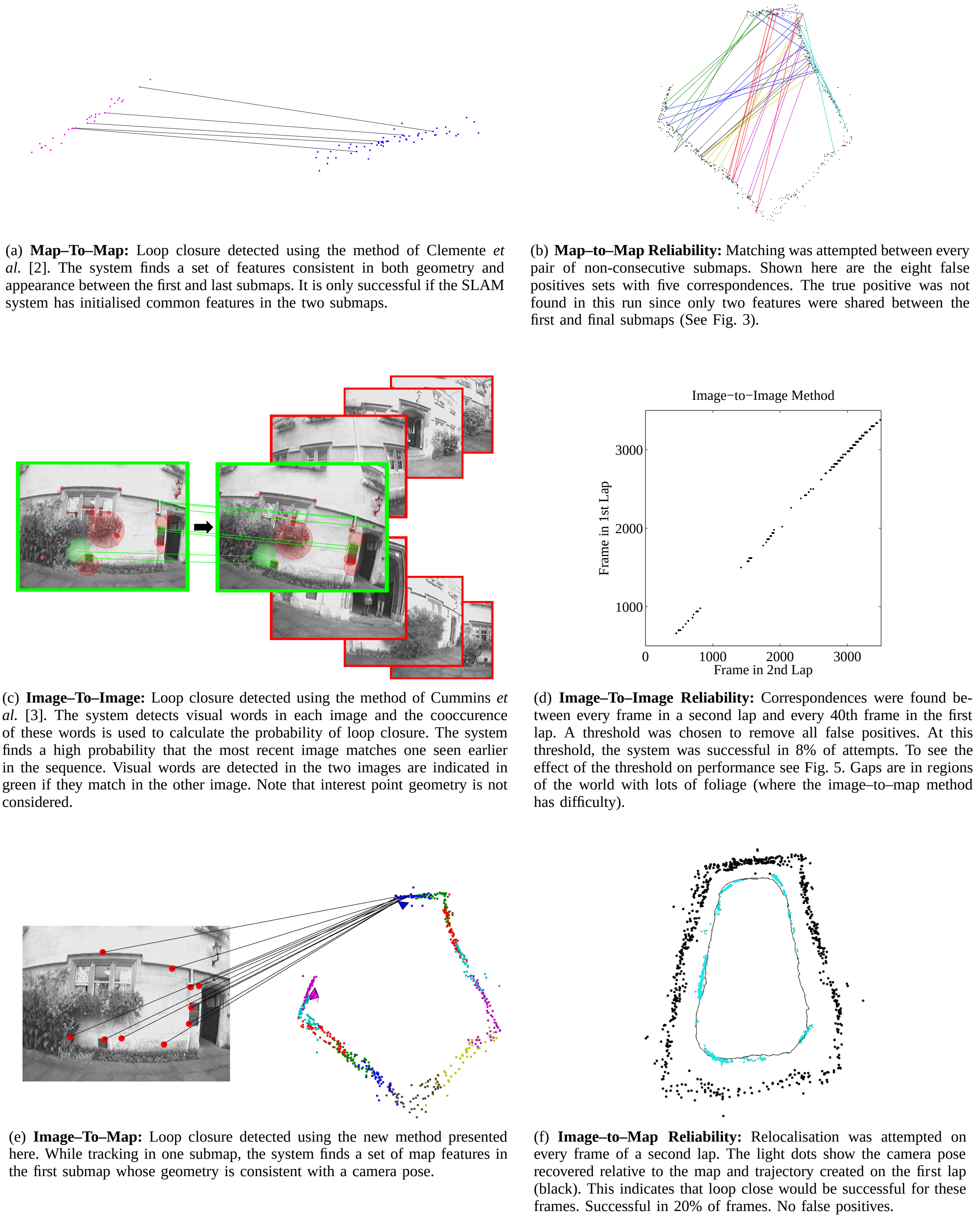
B. *Image–to–Image Matching: Cummins* et al.

The image–to–image matching method of Cummins *et al.*, is designed to work with non-overlapping key frames. When run on a robot, the odometry is used to trigger key frame capture. Without odometry, we simply used every 40th frame of the video to test the system. Ideally though, automatic key frame selection based on appearance should be used.

The loop closure detection system determines for each of these input images if it is a new place or a loop closure. The algorithm correctly gave high probability that each image was a new place until the camera had traversed the loop and returned to the start of the loop. At this point, the system gave high probability (99.9%) that the most recent image corresponded to an image at the start of the sequence (Fig. 4(c)).

To test the reliability of the loop closure detection, we computed loop closures for every frame from the second loop, against the set of images from the first loop. This simulates the ‘kidnapped robot situation’, a sudden transition

[1] Tests were done on a Dual Core 3GHz machine.



(a) **Map–To–Map:** Loop closure detected using the method of Clemente *et al.* [2]. The system finds a set of features consistent in both geometry and appearance between the first and last submaps. It is only successful if the SLAM system has initialised common features in the two submaps.

(b) **Map–to–Map Reliability:** Matching was attempted between every pair of non-consecutive submaps. Shown here are the eight false positives sets with five correspondences. The true positive was not found in this run since only two features were shared between the first and final submaps (See Fig. 3).

(c) **Image–To–Image:** Loop closure detected using the method of Cummins *et al.* [3]. The system detects visual words in each image and the cooccurence of these words is used to calculate the probability of loop closure. The system finds a high probability that the most recent image matches one seen earlier in the sequence. Visual words are detected in the two images are indicated in green if they match in the other image. Note that interest point geometry is not considered.

(d) **Image–To–Image Reliability:** Correspondences were found between every frame in a second lap and every 40th frame in the first lap. A threshold was chosen to remove all false positives. At this threshold, the system was successful in 8% of attempts. To see the effect of the threshold on performance see Fig. 5. Gaps are in regions of the world with lots of foliage (where the image–to–map method has difficulty).

(e) **Image–To–Map:** Loop closure detected using the new method presented here. While tracking in one submap, the system finds a set of map features in the first submap whose geometry is consistent with a camera pose.

(f) **Image–to–Map Reliability:** Relocalisation was attempted on every frame of a second lap. The light dots show the camera pose recovered relative to the map and trajectory created on the first lap (black). This indicates that loop close would be successful for these frames. Successful in 20% of frames. No false positives.

Fig. 4. The results of our experiments on all three loop closing methods. The left column shows a successful loop closure for each method. The right column shows tests on the reliability of each method.

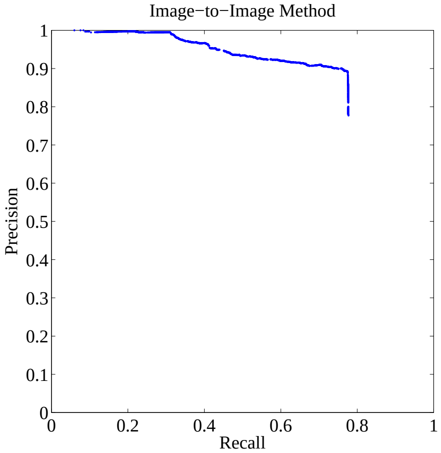


Fig. 5. This precision–recall curve for the image–to–image method [3] shows the algorithm performs well. Quite a high number of true loop closure are detected with few false positives.

from the end of the first loop to a random part of the courtyard. It is a way to test if the algorithm would be able to detect a loop closure at each position. As the method makes use of temporal coherence of detections, to test loop closure at frame i, the algorithm was also provided with frames $i-40$, $i-80$ and $i-120$.

The results are shown in Fig. 4(d) where frames that matched an image in the previous loop are marked. A threshold was chosen that removes all false positives to allow comparison with the image–to–map method. The system found matches that met this probability threshold in 8% of attempts indicating that the system would be able to close the loop at these positions. The precision–recall curve in Fig. 5 shows the effect of the probability threshold on the reliability of the system.

On each image, the algorithm takes on average 283ms to run. Much of this time (73ms) is taken up by SURF feature detection. This method relies on this descriptor which is richer yet slower than the randomised fern classifier. The overall speed is slower than the framerate, however, the loop closing algorithm does not need to be run on every frame.

C. Image–to–Map Matching: Our New Method

At every frame, there is usually enough remaining time after tracking to attempt relocalisation in one other submap. The system cycles through submaps until a relocalisation is successful, indicating a loop closure. For the university courtyard sequence, the system successfully detected the loop closure as the features in the original map came back into view (Fig. 4(e)). It then began tracking the camera in both submaps, and aligned the resultant common trajectory. Using the transformation to align trajectories, the map was optimised as shown in Fig. 1(c). Note that for this method, no common features are needed between submaps like for the GCBB method. Image–to–map matching is done using the features in each submap independently.

The reliability of our new loop closure method was tested using the same ‘kidnapped robot’ situation we used to test the image-to-image method. The system was allowed to continue searching for loop closures as the camera continued around the courtyard for a second lap. For the test, the system attempts relocalisation in every submap for every frame. The results of this test can be seen in Fig. 4(f). The system was able to relocalise relative to the map created in the first loop 20% of the time and no false positives were returned.

Our method takes 10-15ms to find potential matches to map features in each image using the randomised fern classifier. The remaining time is used to run RANSAC on the matches to determine the pose. This is usually found within a few milliseconds if a valid pose exists for those matches. This is fast enough to allow the algorithm to run on a single submap after the system has finished tracking in each frame.

V. DISCUSSION

We have tested three quite different approaches to detecting loop closure for monocular SLAM systems each of which has benefits and drawbacks, as well as tunable parameters which affect their performance.

The first algorithm, variable scale GCBB, detects correspondences between common features in two submaps using both the visual appearance of the features and their relative 3D position. The algorithm takes entire submaps into account rather than just the current view like the other two methods. However, we found the algorithm to be unsuitable since the system cannot guarantee common features in overlapping submaps. Also, the algorithm finds correspondences between submaps with no true common features. As well as the size of the correspondence set, thresholds exist for determining potential feature matches both in appearance and in metric space. Unfortunately, no values could be found to allow the algorithm to reliably distinguish true loop closures. The GCBB algorithm could be made to work better with more detailed information in the maps such as higher level geometry like planes or lines, or simply more dense point clouds. The algorithm should also take into account the angle a feature can be viewed at rather than treating each feature as an ideal point in 3D.

The image–to–image matching technique of Cummins *et al.* compares each current image with the previous images seen by the camera. It was able to detect the loop closure in the university courtyard sequence and in a test of the reliability using a second lap, a probability threshold could be found which removed all false positives while still finding 8% of the true positives. The other tunable parameters, the interest point detector threshold, and the visual word quantizing threshold were kept fixed during the experiment. By not using geometric information, the algorithm is even able to work when the metric map has inaccuracies. However, if the geometric position of the correspondences were taken into account, some of the obvious false positives would be detected as such (Fig. 6). With these extra checks, the system could achieve higher true positive rates like the image–to–map method.

The image–to–map matching technique detects potential correspondences to map features in the image and then

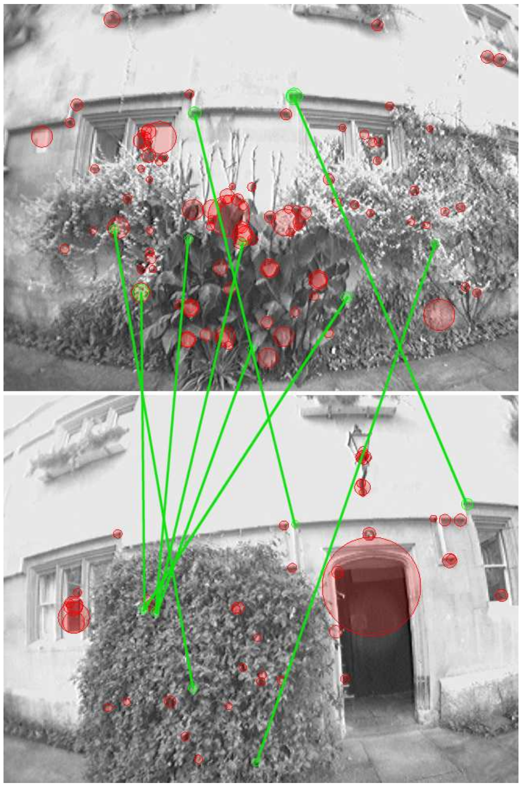


Fig. 6. Image-to-image method: False positive with matching probability of 99.9935%. The detected visual words are indicated in each image in green if they match the other image. This false positive could easily be discarded if the geometric information were known for the detected visual words.

finds a camera pose consistent with their 3D geometry. The method does not rely on the same features being reinitialised in the new submap as the loop is being closed like the map–to–map method, and by taking both the appearance and geometric information into account, it is able to achieve a higher true positive rate than the image–to–image method. The image–to–map method has thresholds for the interest point detector, the randomised fern classifier decision, the number of features for RANSAC concensus, and the minimum percentage of good observations during tracking immediately after a relocalisation. All of these were chosen during the development of the relocalisation module [14] to minimise false positives. Though this new method performs better than the image–to–image method in the university courtyard used here, it does not scale as well to city sized maps. The algorithm is very memory intensive requiring 1.25MB of RAM per feature for the randomised fern classifier, and the number of hypotheses for RANSAC to test grows with the number of map features. Extending the algorithm to work in larger environments remains for future work.

VI. Conclusion

An image–to–map loop closure detection method was presented. It makes use of our previous relocalisation system [14] which detects potential correspondences to map features using a randomised fern classifier which is trained during tracking. A set of correspondences are then found which is consistent with the map geometry using RANSAC and a three–point–pose algorithm. The algorithm can be used to detect loop closures or overlaps between independent maps so they can be merged. The relative transformation needed to align maps is computed using a common trajectory estimated in both maps, allowing maps of different scales to be aligned.

Experiments were performed to compare the performance of the image–to–map matching algorithm against loop closure methods based on both map–to–map and image–to–image matching. The map–to–map method was found to be unsuitable for the sparse maps used in monocular SLAM. The image–to–image method was successful and easy to use but is not complete if the relative metric pose is required. However, the best results were obtained by the image–to–map method which made most use of the available information by taking into account both the appearance and the geometry.

VII. Acknowledgments

We gratefully acknowledge the financial support of the EPSRC (grant GR/T24685, EP/D037077, and a studentship to BW) and the Royal Society (International Joint Project).

References

[1] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proc. European Conference on Computer Vision*, 2006.

[2] L. Clemente, A. Davison, I. Reid, J. Neira, and J. D. Tardós. Mapping large loops with a single hand-held camera. In *Robotics Science and Systems*, 2007.

[3] M. Cummins and P. Newman. Accelerated appearance-only SLAM. In *Proc. IEEE International Conference on Robotics and Automation*, 2008.

[4] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. IEEE International Conference on Computer Vision*, 2003.

[5] C. Estrada, J. Neira, and J. D. Tardós. Hierarchical SLAM: Real-time accurate mapping of large environments. *Transactions on Robotics*, 1(4), 2005.

[6] M. A. Fischler and R. C. Bolles. RANdom SAmple Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[8] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society*, 4(4):629–642, 1987.

[9] A. Howard, G. Sukhatme, and M. Matarić. Multi-robot mapping using manifold representations. In *Proc. International Conference on Robotics and Automation*, 2004.

[10] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, 2006.

[11] J. M. M. Montiel, J. Civera, and A. J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Proc. Robotics Science and Systems*, 2006.

[12] J. Neira and J. D. Tardós. Data association in stochastic mapping using the joint compatibility test. In *IEEE Transactions on Robotics and Automation*, pages 890–897, 2001.

[13] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *Proc. IEEE International Conference on Computer Vision*, 2003.

[14] B. Williams, G. Klein, and I. Reid. Real-time SLAM relocalisation. In *Proc. International Conference on Computer Vision*, 2007.

[15] B. Williams, P. Smith, and I. Reid. Automatic relocalisation for a single-camera simultaneous localisation and mapping system. In *Proc. International Conference on Robotics and Automation*, 2007.