Improved Appearance-Based Matching in Similar and Dynamic Environments using a Vocabulary Tree

Deon Sabatta[1,2]
[1]Mobile Intelligent Autonomous Systems Group
Council for Scientific and Industrial Research
Pretoria, South Africa

Davide Scaramuzza[2], Roland Siegwart[2]
[2]Autonomous Systems Lab
ETH, Zurich
Switzerland

***Abstract*— In this paper we present a topological map building algorithm based on a Vocabulary Tree that is robust to features present in dynamic or similar environments. The algorithm recognises incorrect loop closures, not supported by the odometry, and uses this information to update the feature weights in the tree to suppress further associations from these features. Two methods of adjusting these feature entropies are proposed, one decreasing entropy related to incorrect features in a uniform manner and the other proportional to the contribution of the said feature. Preliminary results showing the performance of the proposed method are presented where it is found that by adjusting the feature entropies, the number of incorrect associations can be reduced while improving the quality of the correct matches.**

I. Introduction

In the field of autonomous vehicles, mapping and localisation are some of the essential tasks, without which full autonomy would not be possible. One common class of localisation methods employs a form of image similarity to perform appearance-based localisation [1], [2], [3], [4]. When this image similarity relies purely on the presence of local features, and not their geometric arrangement, false associations may occur.

Office scenarios are classical examples of this phenomenon, where most offices often contain similar, if not identical, furniture and equipment but in different configurations. Appearance-based methods relying only on local features in these environments will form incorrect association hypotheses in these cases. The problem also appears in unstructured outdoor environments, where unique features often appear on objects that provide bad location information (eg. vehicles, signage).

Most appearance-based matching algorithms compensate for frequently observed features (and objects) though the use of some form of feature entropy that discourages matching based solely on these "common" features. However, this method does not cater for features that appear rarely but still provide poor appearance-based cues.

To overcome this problem, we present an algorithm based on the vocabulary tree by Nister et al. [5], that modifies the feature entropy to discourage the use of features that have previously given bad localisation hypotheses. This approach also caters for features present on objects that are likely to be moved around (eg. chairs, parked cars), learning that these features are not to be trusted for future metric localisation.

The next section contains a brief discussion of related work. Thereafter, we present the algorithm (Section III) and experimental work (Section IV). Finally, in Section V we close with the conclusions.

II. Related Work

Initially, Simultaneous Localisation and Mapping or SLAM algorithms focused predominately on range devices (such as sonar and laser range scanners) as their accuracy was good and their low data rates were easy to process. However, as environments grew in size, perceptual aliasing problems became apparent. Vision sensors, by comparison, give far more information and so, were able to better distinguish various environments, but at a much greater computational cost. Initial vision-based SLAM algorithms focused on easily computable global image properties such as histograms [6], image gradients [7], principal component analysis [8] and combinations of simple features [1]. The advent of reliable local feature detectors and descriptors, such as SIFT [9], allowed SLAM algorithms to build feature-based environment representations (eg. [10]). These local features were soon incorporated back into topological mapping algorithms where similarity between collections of features were used for localisation. The emergence of ever improving, fast, visual appearance matching techniques such as Video-Google [11] and later the Vocabulary Tree [5] and Fab-MAP [12], allowed real-time appearance-based map building and localisation using local features. These appearance-based methods typically include some form of feature entropy or importance weight to flag features as being common in the environment and not useful for association. However, these approaches fail to capture items that, while unique within their environments, appear several times in different places.

Several implementations integrated the appearance-based information with a graphSLAM representation to recover metric information. GraphSLAM, originally considered off-line [13], has also gained new popularity based on several algorithms to greatly speed up the convergence process. Algorithms such as Multi-level Relaxation [14] and Tree Network Optimisation [15] have closed exceedingly large loops

This work was supported by funding from the Council for Scientific and Industrial Research (CSIR), South Africa. D. Sabatta is a research engineer with the Mobile Intelligent Autonomous Systems research group of the CSIR. dsabatta@csir.co.za

successfully. These methods have lead to appearance-based SLAM algorithms that included metric information [16], [3]. While these methods have proved highly successful, the problem of incorrect loop closures are still a problem. As mentioned in the introduction, these problems occur increasingly in unstructured, outdoor and dynamic environments. To mitigate this, some algorithms have specified stringent matching thresholds to ensure that no incorrect loop closures are obtained at the cost of fewer total loop closures. This being said, the matching process is of a statistical nature, and therefore, this problem will always exist.

In addition to the problems of perceptual aliasing, dynamic environments have posed their own problems within the SLAM community. Using features on dynamic obstacles leads to poor localisation and inconsistent maps. Current methods of handling dynamic environments centre largely around tracking moving objects within the immediate environment and removing them from the map [17], [18]. This poses a problem when the dynamic obstacles are stationary during mapping or move slowly over time. In contrast to the above approach, we attempt to identify incorrect loop closures that are not supported by the odometry. Through doing this we hope to be able to learn which features are typically found on dynamic objects, and exclude them from the mapping process, regardless of whether or not the object is moving when we observe it.

III. Algorithm

In this section we discuss the mapping algorithm that ultimately results in entropy reduction in the vocabulary tree for features that lead to poor localisation.

We begin with an image stream from a perspective camera from which we extract SIFT features. These images and their associated features are loaded into a vocabulary tree that performs appearance-based associations to all previous frames. When an appearance-based link is found, the relative motion between the two frames is estimated using Structure from Motion (SFM). This motion hypothesis is then used in a graph relaxation framework to determine its support by the odometry. If the motion hypothesis is deemed to be unlikely, given the prior map, the entropy in the vocabulary tree associated with the features that lead to the incorrect hypothesis are reduced to prevent future appearance-based associations by the same collection of features. This approach accentuates appearance-based matches using background information rather than the more salient *dynamic* features.

A. Feature Extraction and Map Building

We begin with a raw image stream from a perspective camera mounted on the front of a mobile platform. To reduce the computational load, we sample the images equally in space approximately 1m apart and compute SIFT features without the initial doubling of the images.

1) Map Representation: In our algorithm, our map is represented as a graph where nodes represent the pose of the robot, or equivalently, locations in space and edges represent known connections between poses. These connections are calculated from one of two sources, either directly from the measured odometry between poses (only for nearby poses) or by using Structure from Motion. Links between poses are represented by Gaussian distributions indicating the uncertainty in the relative motion between the two poses.

Following on the work of Frese et al. in [14], we represent the link between two poses by the vector connecting the two poses, rotated into the frame of the second pose; and the change in heading between the two poses. Given two poses $a = [a_x, a_y, a_\phi]^{\mathrm{T}}$ and $b = [b_x, b_y, b_\phi]^{\mathrm{T}}$, we determine the edge $f(a, b)$ as,

$$f(a,b) = \begin{bmatrix} (a_x - b_x)\cos(b_\phi) + (a_y - b_y)\sin(b_\phi) \\ -(a_x - b_x)\sin(b_\phi) + (a_y - b_y)\cos(b_\phi) \\ a_\phi - b_\phi \end{bmatrix}, \tag{1}$$

and define a suitable covariance matrix to capture the uncertainties in the platform model.

Given a map consisting of a set,

$$\mathcal{R} = \{(\mu_r, C_r) \mid r = 1 \cdots |\mathcal{R}|\},$$

of links and associated covariances between poses. A maximum-likelihood map hypothesis may be found by minimising the negative log likelihood or χ^2 energy function;

$$\chi^2(x) = \sum_{r \in \mathcal{R}} z_r^{\mathrm{T}} C_r^{-1} z_r, \tag{2}$$

where,

$$z_r = f\left(x_{a_r}, x_{b_r}\right) - \mu_r.$$

2) Vocabulary Tree: The appearance-based similarity scoring method we use is based on the Vocabulary Tree of Nister [5]. In this algorithm, the feature space is clustered using a hierarchical K-means algorithm to form a tree. Descriptors in the images are then replaced by a node sequence from the root of the cluster tree to one of the leaf nodes.

For each node in the tree, a feature entropy is calculated as,

$$w_i = \ln(N/N_i),$$

where N is the total number of documents used during training and N_i is the number of documents that had at least one descriptor pass through node i. As expected, the entropy weights increase from 0 at the root of the tree towards the leaf nodes.

To calculate the similarity score between two images, we determine the number of times each node in the vocabulary tree is touched when descending the image descriptors through the tree structure. This count is multiplied by the node entropy and the vector is normalised in the Euclidean sense. This results in an image vector with as many elements as nodes in the tree.

To compare two images, we calculate the image vectors based on the feature clustering and entropy and then calculate the cosine similarity between these two vectors.[1] This results

[1]For the case of normalised vectors, this is equivalent to the scalar-vector or dot product.

in an image similarity score between 0 and 1 where 0 implies no common feature descriptors and 1 implies an identical set of feature descriptors between the two images.

Several methods exist to improve the performance of this image similarity computation. The interested reader is referred to [5].

3) Map Building Algorithm: The map building algorithm is centred around a vocabulary tree where the comparison and insertion stages are separated by a guard-band to prevent premature associations with recently added frames. The pseudo-code for the map building algorithm is given in Algorithm 1.

Algorithm 1 Map-Building Algorithm using Vocabulary Tree Associations

Require: $Guardband > 0$, Matching threshold $thresh > 0$ and a vocabulary tree VT.
  $score \leftarrow \text{zeros}(1, Guardband)$
  for all $image$ **in** $Image\ Sequence$ **do**
    Add odometry link between current and previous frame to map
    $s \leftarrow$ maximum similarity score for $image$ in VT.
    if $score(1) == \max(score)$ and $score(1) > threshold$ **then**
      Add association link to map
      $score \leftarrow \text{zeros}(1, Guardband)$
    else
      Add image associated with $score(1)$ to VT.
    end if
    $score \leftarrow [score(2 : \mathbf{end})\ s]$
  end for

B. Motion Estimation from SFM

When an appearance-based match has been found in the environment, the edge in (1) is estimated from the matched frames using Structure from Motion. The operation of the algorithm is as follows:

1) Calculate the fundamental matrix relating the two images using the 7-point algorithm in a RANSAC (Random Sampling Consensus) framework. Extract the rotation and translation information between the two frames from the fundamental matrix. [19].
2) Use this motion hypothesis to reconstruct the 3D locations of the feature points [19].
3) Use a perspective n-Point algorithm [20] to estimate the location of a third frame taken at a known distance from one of the initial frames.[2]
4) Use the estimated translation vs. known translation extracted from odometry to calculate unknown scale in initial reconstruction.
5) Scale the initial motion hypothesis from step 1 using the scale computed in the previous step.

[2]Typically a neighbouring frame from the input image sequence is used where the odometry is considered to be reasonably accurate.

The inliers returned by the RANSAC algorithm in step 1 above are maintained for use later when calculating the uncertainty and adjusting the feature entropy.

1) Uncertainty Calculation: Before this motion hypothesis can be integrated into the map of Section III-A.1, an associated uncertainty, in the form of a covariance matrix, needs to be estimated. Due to the complexity of the SFM algorithm, we calculate the uncertainty using a Monte-Carlo simulation method.

To accomplish this, we assume that the transfer function from the feature point locations to the final SFM hypothesis is locally linear. We then apply a zero-mean gaussian perturbation to the feature points and recompute the motion hypothesis using these new feature locations. The estimated covariance of the motion model may then be calculated from multiple trials and normalised by the known covariance of the input noise.

To determine an estimate of the uncertainty in the locations of the detected features, we use the fundamental matrix from the SFM motion hypothesis calculated above to find optimal feature locations $\hat{x_i}$. These points satisfy the fundamental matrix relation $\hat{x_i}^{\mathrm{T}} F \hat{x_i} = 0$ and minimise the Euclidian distance $||\hat{x_i} - x_i||$ between the original and optimal feature locations. The covariance of these offsets, $\hat{x_i} - x_i$, is then used as an estimate of the noise present on the detected features.

C. Map Relaxation and Hypothesis Validation

The purpose of map relaxation is to find the values for the robot poses $\{x_1, x_2, \ldots, x_n\}$ that minimises the Chi-Squared function (2) given the mean and covariance of all measured translations between two poses.

In the case where no loop closures have been observed, this is equivalent to the measured odometry and $\chi^2(x) = 0$. However, when a loop closure is observed and integrated into the map representation, the accumulated error in the odometry between the two poses that are involved in the loop closure must be distributed over the odometry relations. This is equivalent to minimising the equation in (2).

To do this we use the multi-level relaxation algorithm of Frese et al. [14]. Relaxation refers to the tendency of the graph to slowly ease into a configuration that represents a local minimum of (2). This is achieved by calculating a locally linear representation of the $\chi^2(x)$ function, about the current graph and solving for the value of x that yields a local minimum using the Gauss-Seidel method.

1) Hypothesis Validation: Integral to our algorithm is the ability to determine that a proposed motion hypothesis is valid. To do this, we consider the likelihood of the map after the loop closure hypothesis has been integrated, as well as the likelihood of the loop closure.

To evaluate the "*correctness*" of a loop closure hypothesis, we consider the likelihood of the graph before and after the loop closure has been integrated. Assuming all the links in the graph are represented by Gaussian random variables, we

may calculate the likelihood of the map as,

$$P_M(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{3|\mathcal{R}|}{2}} \prod_{r\in\mathcal{R}} \det(C_r)} \exp\left\{-\frac{1}{2}\chi^2(\mathbf{x})\right\}. \tag{3}$$

In the case where there are no loop closures, the map follows the odometry exactly and $\chi^2(\mathbf{x})$ is zero. In this case, the likelihood function $P_M(\mathbf{x})$ takes on its maximum value. When a loop closure is integrated into the map, the exponential term in (3) decreases, which could result in a lower likelihood, even for correct associations. To overcome this, we scale the map likelihood function by the probability of a correct loop closure to obtain a joint likelihood,

$$P_{M\cap O} = P_{M|O} P_O, \tag{4}$$

where $P_{M|O}$ is the map likelihood once the loop closure has been added and P_O is the probability of the loop-closure hypothesis being correct. We estimate P_O as,

$$P_O = m/M,$$

where m is the number of feature matches that satisfy the SFM motion hypothesis and M the total number of matches between the two images.

This allows us to compare the likelihoods of the map representation with and without the loop closure hypothesis by comparing $P_{M-\{O\}} = P_M \times (1 - P_O)$ and $P_{M\cap O}$. The higher value is then chosen as the correct hypothesis.

D. Feature Entropy Adjustment

Once we have determined that a set of features has lead to an incorrect loop closure hypothesis, we would like to reduce the entropy of these features in the vocabulary tree to inhibit future matches from the same set of features. To do this we propose two methods. The first is to reduce the entropy of the features in a uniform manner related only to their magnitude. The second, is to reduce the entropy of a feature depending on its contribution to the incorrect motion hypothesis.

1) Uniform Entropy Adjustment: To perform uniform entropy adjustment, we consider the sets of descriptors of the features in both images corresponding to the motion hypothesis individually. For each set of descriptors, we determine the list of nodes in the vocabulary tree that the descriptors passed through during classification. For each of these nodes, we multiply the node entropy by a constant scaling factor k.

2) Weighted Entropy Adjustment: In this method, we modify the entropy of the various nodes of the vocabulary tree by a factor related to the contribution of that node to the image similarity score. This allows us to, within reason, specify the desired similarity score between two images.

We begin with the definition of an image vector calculated for image k as, $\mathbf{s}^k = \{s_i^k \mid s_i^k = n_i^k w_i\}$, where n_i^k is the number of times node i is touched by descriptors of image k and w_i is the entropy of node i. We then define a contribution vector for image k as $\mathbf{c}^k = \{c_i^k \mid c_i^k = \hat{n}_i^k / n_i^k\}$, where $\hat{n}_i^k$ is the number of times node i is touched by descriptors that satisfy the SFM hypothesis. The contribution vector represents the degree to which each element of the image vector contributed to the matching score caused by the "*bad*" features. We will try to adjust the entropies of the nodes involved to reach a desired similarity score d.

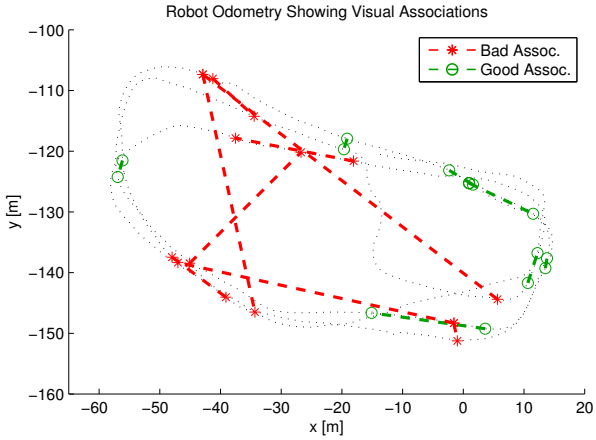

Fig. 1. Map of robot path showing visual associations obtained from a conventional vocabulary tree using the map building algorithm.

Given two images defined by image vectors $\mathbf{s}^1$ and $\mathbf{s}^2$, the process involved in updating the feature entropies to achieve the desired score may be summarised as follows:

1) Determine the amount by which the total score must be altered as $\Delta = d - \sum_i s_i^1 s_i^2$.
2) Calculate the contribution of each of the terms of the score vector to the "*bad*" match as $s_i' = s_i^1 s_i^2 \sqrt{c_i^1 c_i^2}$.
3) Scale the individual term contribution by the desired change in score as, $s_i'' = \left(1 + \frac{\Delta}{\Sigma s_i'}\right) s_i'$.
4) Calculate the scaling factor required to effect the desired change in component score terms as, $k_i = s_i'' / s_i^1 s_i^2$.
5) Modify the entropy weight of each node in the tree by multiplying it with the scale factor k_i.

When calculating the final scaling coefficients, care should be taken to avoid NaN and infinite values resulting from a divide by zero. The scaling terms should only be applied to nodes where the contribution term $\sqrt{c_i^1 c_i^2}$ is non-zero.

IV. Experimental Results

To simulate a dynamic environment where the robot would often encounter a feature rich object that could lead to poor loop closing performance, we drove the platform around a parking lot for several loops while periodically placing a checker board in the robot's field of view. The robot was driven over roughly 600m collecting 370 images. After feature extraction, we obtained an average of 251.9 features per frame with a standard deviation of approximately 98.

We started by constructing a vocabulary tree for the features extracted from the images and used this together with the map building algorithm of Section III-A.3 with a matching threshold of 0.25 and a guard band of 10. The

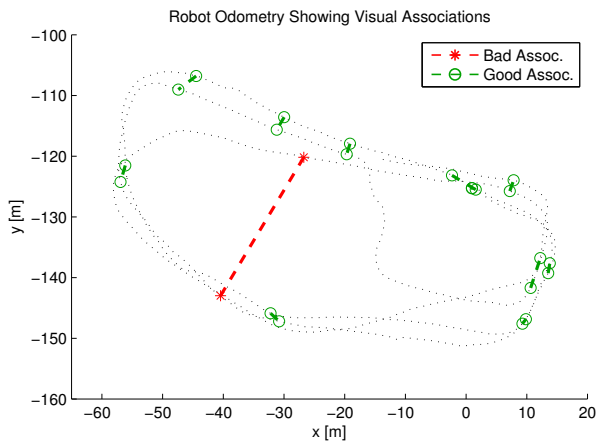

Fig. 2. Updated map showing associations obtained after unreliable features have been learnt.

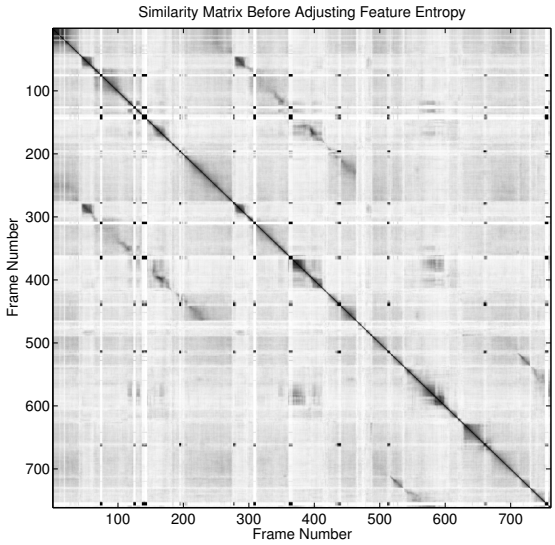

Fig. 3. Similarity matrix of frames in the dataset computed using the original vocabulary tree feature entropy weights.

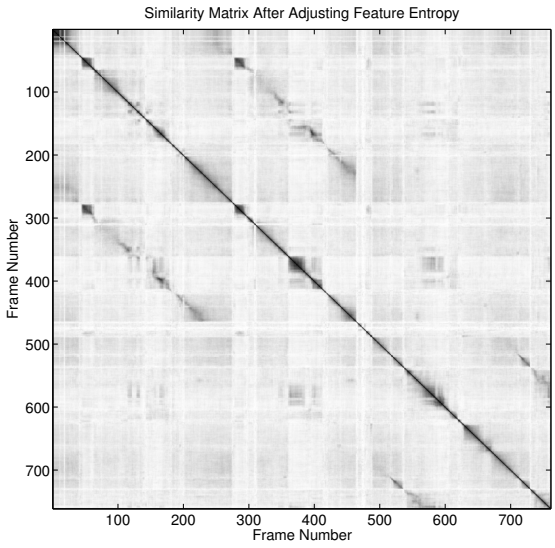

Fig. 4. Similarity matrix of frames in the dataset after the feature entropy of unreliable features has been adjusted.

resulting map and associations may be seen in Figure 1. In this map, the vocabulary tree made seven correct and nine incorrect associations. These associations could have been greatly improved by increasing the matching threshold; however, a low threshold was chosen to illustrate the performance of the suggested algorithm. In this map, the bottom-right association that appears to be correct failed as the SFM algorithm proposed a motion hypothesis based on the checker boards in the scene, rather than the background information. This led to a poor relative distance estimate that the mapping algorithm threw out.

After performing entropy reduction using the poor associations with the weighted entropy adjustment of Section III-D.2, we obtain the refined map in Figure 2. In this map we have one incorrect and eleven correct associations. We also notice that the quality of the matches has been greatly improved — including the addition of several matches not seen before. This is due to the maximum similarity score selection of the map building algorithm. While the association scores of regions containing a checker board are higher than neighbouring similarities, the algorithm rejects the correct matches with the lower score.

The performance of the matching algorithm before and after feature entropy updates can also be seen by comparing the similarity matrices generated for frames in the dataset using the original and updated feature entropies. The similarity matrix for the dataset using the original feature entropies returned by the vocabulary tree is shown in Figure 3. In this similarity matrix we can see all the off-diagonal matches relating to scenes where the checker board was visible. Once the feature entropy has been updated according to the proposed algorithm, the similarity matrix in Figure 4 is obtained. In this figure, we see that the general structure of the similarity matrix has not been affected that much; however, the off-diagonal spots (and hence the incorrect associations) have been removed.

To illustrate the effect of incorrect associations on the similarity score, we consider the matching example shown in Figure 5. In this example, we have extracted an image from the dataset that includes a checker board and consider the similarity score between this image and an image of the background scene as well as one of only a checker board. Similarity scores between images 1 and 2 are represented by solid lines in the graph while those between images 2 and 3 are represented by a dashed line. Using the initial vocabulary tree feature entropies, the similarity score associated with the checker board is much higher than of that with the background features. As the algorithm identifies poor matches, the feature entropies are updated and the similarity

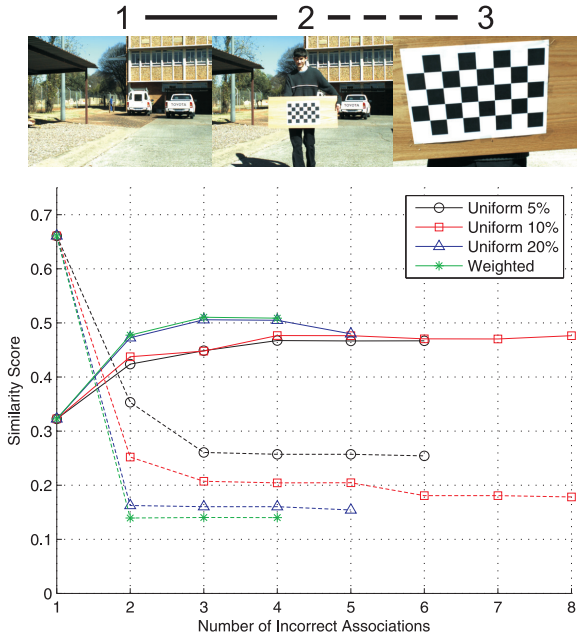

Fig. 5. Plot showing the evolution of similarity scores as the algorithm detects poor matches in the dataset.

score to the background image improves while the match to the checker board degrades.

The result is shown for uniform entropy reductions of 5%, 10% and 20% as well as for the weighted entropy adjustment. As expected, the lower the percentage in the uniform entropy adjustment, the slower the similarity score declines and the more incorrect associations are required. The weighted entropy adjustment algorithm also has the advantage of being able to specify the desired upper bound on the similarity score, while the uniform entropy adjustment will perform differently depending on the initial similarity score.

The slight decrease in similarity score between images 1 and 2 towards the end of the 20% and weighted adjustment cases is as a result of the last remaining incorrect association in the map. This association is attributed mostly to features associated with windows similar to those on the building in image 1. As such, the entropies of these features are reduced resulting in the observed reduction in similarity score.

V. Conclusions

In this paper, we have presented a method for updating the feature entropy weights in a vocabulary tree to cater for features that provide poor metric localisation information. Two methods for updating the feature entropy were provided. One that reduces feature entropy in a uniform manner regardless of the contribution to the poor match and the second that adjusts feature entropy depending on its contribution. The results of the algorithm on a preliminary dataset show that it is capable of reducing the number of incorrect associations while simultaneously improving the match quality for good associations. This method should work in areas where pure appearance-based methods could fail based on the presence of similar objects, such as office environments. The approach also lends itself to applications in environments where appearance-based matches are highly influenced by dynamic objects.

References

[1] P. Lamon, A. Tapus, E. Glauser, N. Tomatis, and R. Siegwart. Environmental modeling with fingerprint sequences for topological global localization. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3781–3786, October 2003.

[2] P. Newman, D. Cole, and K. Ho. Outdoor SLAM using visual appearance and laser ranging. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, pages 1180–1187, May 2006.

[3] M. J. Milford and G. F. Wyeth. Mapping a suburb with a single camera using a biologically inspired SLAM system. *IEEE Transactions on Robotics*, 24(5):1038–1053, October 2008.

[4] Booij O, B. Terwijn, Z. Zivkovic, and B. Krose. Navigation using an appearance based topological map. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pages 3927–3932, April 2007.

[5] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168, 2006.

[6] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, volume 2, pages 1023–1029, 2000.

[7] T. N. Tan and K. D. Baker. Efficient image gradient based vehicle localisation. *IEEE Transactions on Image Processing*, 9(8):1343–1356, August 2000.

[8] V. Colin de Verdiere and J. L. Crowley. Local appearance space for recognition of navigation landmarks. *Robotics and Autonomous Systems*, 31(1):61–69, April 2000.

[9] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157, September 1999.

[10] S. Se, D. G. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8):735–758, 2001.

[11] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477, October 2003.

[12] M. Cummins and P. Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, 27(6):647–665, 2008.

[13] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[14] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, April 2005.

[15] U. Frese. Treemap: An O(log n) algorithm for indoor simultaneous localization and mapping. *Autonomous Robots*, 21(2):103–122, September 2006.

[16] H. Andreasson, T. Duckett, and A. J. Lilienthal. A minimalistic approach to appearance-based visual SLAM. *IEEE Transactions on Robotics*, 24(5):1–11, October 2008.

[17] D. Wolf and G. S. Sukhatme. Online simultaneous localization and mapping in dynamic environments. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, volume 2, pages 1301–1307, May 2004.

[18] Chieh-Chih Wang, C. Thorpe, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, volume 1, pages 842–849, September 2003.

[19] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004.

[20] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An accurate O(n) solution to the PnP problem. *International Journal of Computer Vision*, 81(2):155–166, February 2009.