Appearance-Based Loop Closure Detection in Real-Time for Large-Scale and Long-Term Operation

Mathieu Labbé, Student Member, IEEE, François Michaud, Member, IEEE

Abstract—In appearance-based localization and mapping, loop closure detection is the process used to determine if the current observation comes from a previously visited location or a new one. As the size of the internal map increases, so does the time required to compare new observations with all stored locations, eventually limiting real-time processing. This paper presents a real-time loop closure detection approach for large-scale and long-term operation. The approach is based on a memory management method, which limits the number of locations used for loop closure detection so that the computation time remains under real-time constraints. The idea consists of keeping the most recent and frequently observed locations in a Working Memory (WM) used for loop closure detection, and transferring the others into a Long-Term Memory (LTM). When a match is found between the current location and one stored in WM, associated locations stored in LTM can be updated and remembered for additional loop closure detections. Results demonstrate the approach’s adaptability and scalability using ten standard data sets from other appearance-based loop closure approaches, one custom data set using real images taken over a 2 km loop of our university campus, and one custom data set (7 hours) using virtual images from the racing video game “Need for Speed: Most Wanted”.

Index Terms—Appearance-based localization and mapping, place recognition, bag-of-words approach, dynamic Bayes filtering.

I. INTRODUCTION

AUTONOMOUS robots operating in real life settings must be able to navigate in large, unstructured, dynamic and unknown spaces. Simultaneous localization and mapping (SLAM) [1] is the capability required by robots to build and update a map of their operating environment and to localize themselves in it. A key feature in SLAM is to recognize previously visited locations. This process is also known as loop closure detection, referring to the fact that coming back to a previously visited location makes it possible to associate this location with another one recently visited.

For most of the probabilistic SLAM approaches [2]–[13], loop closure detection is done locally, i.e., matches are found between new observations and a limited region of the map, determined by the uncertainty associated with the robot’s position. Such approaches can be processed in real-time at 30 Hz [14] as long as the estimated position is valid, which cannot be guaranteed in real world situations [15]. As an exclusive or complementary alternative, appearance-based loop closure detection approaches [15]–[19] generally detect a loop closure by comparing a new location with all previously visited locations, independently of the estimated position of the robot. If no match is found, then a new location is added to the map. However, a robot operating in large areas for a long period of time will ultimately build a very large map, and the amount of time required to process new observations increases with the number of locations in the map. If computation time becomes larger than the acquisition time, a delay is introduced, making updating and processing the map difficult to achieve in real-time.

Our interest lies in developing a real-time appearance-based loop closure detection approach that can deal with large-scale and long-term operation. Our approach dynamically manages the locations used to detect loop closures, in order to limit the time required to search through previously visited locations. This paper describes our memory management approach to accomplish appearance-based loop closure detection, in a Bayesian framework, with real-time constraints for large-scale and long-term operation. Processing time, i.e., the time required to process an acquired image, is the criterion used to limit the number of locations kept in the robot’s Working Memory (WM). To identify the locations to keep in WM, the solution studied in this paper consists of keeping the most recent and frequently observed locations in WM, and transferring the others into Long-Term Memory (LTM). When a match is found between the current location and one stored in WM, associated locations stored in LTM can be remembered and updated. This idea is inspired from observations made by psychologists [20], [21] that people remember more the areas where they spent most of their time, compared to those where they spent less time. By following this heuristic, the compromise made between search time and space is therefore driven by the environment and the experiences of the robot.

Because our memory management mechanism ensures satisfaction of real-time constraints independently of the scale of the mapped environment, our approach is named Real-Time Appearance-Based Mapping (RTAB-Map). An earlier version of RTAB-Map has been presented in [22]. This paper presents in more details the improved version tested with a much wider set of conditions, and is organized as follows. Section II
reviews appearance-based loop closure detection approaches. Section III describes RTAB-Map. Section IV presents experimental results, and Section V presents limitations and possible extensions to our approach.

II. RELATED WORK

For global loop closure detection, vision is the sense generally used to derive observations from the environment because of the distinctiveness of features extracted from the environment [23]–[25], although successful large-scale mapping using laser range finder data is possible [26]. For vision-based mapping, the bag-of-words [27] approach is commonly used [16], [18], [19], [28], [29] and has shown to perform loop closure detection in real-time for paths of up to 1000 km [30]. The bag-of-words approach consists in representing each image by visual words taken from a dictionary. The visual words are usually made from local feature descriptors, such as Scale-Invariant Feature Transform (SIFT) [31]. These features have high dimensionality, making it important to quantize them into a vocabulary (or dictionary) for fast comparison instead of making direct comparisons between features. Popular quantization methods are Vocabulary Tree [28], Approximate K-Means [32] or K-D Tree [31]. By linking each word to related images, comparisons can be done efficiently over large data sets, as with the Term Frequency-Inverse Document Frequency (TF-IDF) approach [27].

The vocabulary can be constructed offline (using a training data set) or incrementally constructed online, although the first approach is usually preferred for real-time processing in large-scale environments. However, even if the image comparison using a pre-trained vocabulary (as in FAB-MAP 2.0 [30]) is fast, real-time satisfaction is effectively limited by the maximum size of the mapped environment. The number of comparisons can be decreased by considering only a selection of previously acquired images (referred to as key images) for the matching process, while keeping detection performance nearly the same compared to using all images [33]. Nevertheless, processing time for each image acquired still increases with the number of key images. In [34], a particle filter is used to detect transition between sets of locations referred to categories, but again, the cost of updating the place distribution increases with the number of categories.

Even if the robot is always revisiting the same locations in a closed environment, perceptual aliasing, changes that can occur in dynamic environments or the lack of discriminative information may affect the ability to recognize previously visited locations. This leads to the addition of new locations in the map and consequently influences the satisfaction of real-time constraints [35]. To limit the growth of images to match, pruning [36] or clustering [37] methods can be used to consolidate portions of the map that exceed a spatial density threshold. This limits growth over time, but not in relation to the size of the explored environment.

Finally, memory management approaches have been used in robot localization to increase recognition performance in dynamical environments [38] or to limit memory used [39]. In contrast, our memory management is used for real-time satisfaction in localization and mapping, where new locations are dynamically added over time.

III. REAL-TIME APPEARANCE-BASED MAPPING (RTAB-MAP)

The objective of our work is to provide an appearance-based localization and mapping solution independent of time and size, to achieve real-time loop closure detection for long-term operation in large environments. The idea resides in only using a limited number of locations for loop closure detection so that real-time constraints can be satisfied, while still gain access to locations of the entire map whenever necessary. When the number of locations in the map makes processing time for finding matches greater than a real-time threshold, our approach transfers locations less likely to cause loop closure detection from the robot’s WM to LTM, so that they do not take part in the detection of loop closures. However, if a loop closure is detected, neighbor locations can be retrieved and brought back into WM to be considered in future loop closure detections.

As an illustrative example used throughout this paper, Fig. 1 shows a graph representation of locations after three traversals of the same region. Each location is represented by an image signature, a time index (or age) and a weight, and locations are linked together in a graph by neighbor or loop closure links. These links represent locations near in time or in space, respectively.

Locations in LTM are not used for loop closure detection. Therefore, it is important to choose carefully which locations to transfer to LTM. A naive approach is to use a first-in-first-out (FIFO) policy, pruning the oldest locations from the map to respect real-time constraints. However, this sets a maximum sequence of locations that can be memorized when exploring an environment: if the processing time reaches the real-time threshold before loop closures can be detected, pruning the older locations will make it impossible to find a match. As an alternative, a location could be randomly picked, but it is preferable to keep in WM the locations that are more susceptible to be revisited. As explained in the introduction, the idea studied in this paper is based on the working hypothesis that locations seen more frequently than others are more likely to cause loop closure detections. Therefore, the number of time a location has been consecutively viewed is used to set its weight. When a transfer from WM to LTM is necessary, the location with the lowest weight is selected. If many locations have the same lowest weight, the oldest one is transferred.

![Fig. 1. Graph representation of locations. Vertical arrows are loop closure links and horizontal arrows are neighbor links. Dotted links show not detected loop closures. Black locations are those in LTM, white ones are in WM. Node 455 is the current acquired location.](image-url)
Fig. 2. RTAB-Map memory management model.

Algorithm 1 RTAB-Map

1: \( \text{time} \leftarrow \text{TIME NOW() } \) \( \triangleright \) TIME NOW() returns current time  
2: \( L_t \leftarrow \text{acquired image} \)  
3: \( L_t \leftarrow \text{LOCATION CREATION}(L_t) \)  
4: if \( z_t \) (of \( L_t \)) is a bad signature (using \( T_{\text{bad}} \)) then  
5: \hspace{1em} \text{Delete} \( L_t \)  
6: else  
7: \hspace{1em} \text{Insert} \( L_t \) into \( \text{STM} \), adding a neighbor link with \( L_{t-1} \)  
8: \hspace{1em} \text{Weight Update of} \( L_t \) in \( \text{STM} \) (using \( T_{\text{similarity}} \))  
9: \hspace{1em} if \( \text{STM} \)'s size reached its limit (\( T_{\text{STM}} \)) then  
10: \hspace{1em} \hspace{1em} \text{Move oldest location of} \( \text{STM} \) to \( \text{WM} \)  
11: \hspace{1em} \text{end if}  
12: \hspace{1em} \hspace{1em} \( p(S_i | L_t) \leftarrow \text{Bayesian Filter Update in} \ WM \) with \( L_t \)  
13: \hspace{1em} \hspace{1em} \text{Loop Closure Hypothesis Selection (} S_i = i \)  
14: \hspace{1em} \hspace{1em} if \( S_i = i \) is accepted (using \( T_{\text{loop}} \)) then  
15: \hspace{1em} \hspace{1em} \hspace{1em} \text{Add loop closure link between} \ L_t \ \text{and} \ L_i \)  
16: \hspace{1em} \text{end if}  
17: \hspace{1em} \hspace{1em} \text{Join} \ '\text{trash}'\text{'s thread} \ \triangleright \ \text{Thread started in} \ \text{TRANSFER()} \)  
18: \hspace{1em} \hspace{1em} \text{RETRIEVAL}(L_t) \ \triangleright \ \text{STM} \rightarrow \text{WM} \)  
19: \hspace{1em} \hspace{1em} \( p(T_{\text{time}}) \leftarrow \text{TIME NOW()} - \text{time} \) \( \triangleright \ \text{Processing time} \)  
20: \hspace{1em} \hspace{1em} \text{if} \( p(T_{\text{time}}) > T_{\text{time}} \) then  
21: \hspace{1em} \hspace{1em} \hspace{1em} \text{TRANSFER()} \ \triangleright \ \text{WM} \rightarrow \text{LTM} \)  
22: \hspace{1em} \text{end if}  
23: \text{end if}  

A. Location Creation

The bag-of-words approach [27] is used to create signature \( z_t \) of an image acquired at time \( t \). An image signature is represented by a set of visual words contained in a visual dictionary incrementally constructed online. We chose to use an incremental rather than a pre-trained dictionary to avoid having to go through a training step for the targeted environment. Even if an incremental dictionary is computationally more expensive, it can be used in real-time with our memory management approach.

Using OpenCV [40], Speeded-Up Robust Features (SURF) [41] are extracted from the image to derive visual words. Each SURF feature has a strength referred to as feature response. The feature response is used to select the most prominent features in the image. To avoid bad features, only those over a feature response of \( T_{\text{response}} \) are extracted. A maximum of \( T_{\text{maxFeatures}} \) SURF features with the highest feature response are kept to have nearly the same number of words across the images. If few SURF features are extracted (under a ratio \( T_{\text{bad}} \) of the average features per image), the signature is considered to be a bad signature and is not processed for loop closure detection. This happens when an image does not present discriminative features, such as a white wall in an indoor scene.

For good signatures, to find matches with words already in the dictionary, SURF features are compared using the distance ratio between the nearest and the second-nearest neighbor (called nearest neighbor distance ratio, NNDR). As in [31], two features are considered to be represented by the same word if the distance with the nearest neighbor is less than \( T_{\text{NNDR}} \) times the distance to the second-nearest neighbor. Because of the high dimensionality of SURF features, a randomized forest of kd-trees (using FLANN [42]) is used as the dictionary structure. This structure increases efficiency of the nearest-neighbor search when matching visual words from a new
signature with the ones in the dictionary. For each feature, when the best match in the dictionary is not satisfying the \( T_{\text{NNDR}} \) criterion, a new word is created with the feature’s descriptor and is added to the dictionary. At each iteration, as words are added or removed when locations are retrieved or transferred (as described in Section III-E and Section III-F), the kd-trees are reconstructed to always have a balanced dictionary.

A location \( L_t \) is then created with signature \( z_t \) and time index \( t \); its weight initialized to 0 and a bidirectional link in the graph with \( L_{t-1} \). The summary of the location creation procedure is shown in Algorithm 2.

Algorithm 2 Create location \( L \) with image \( I \)

1: procedure LOCATION_CREATION(I)
2: \( f \leftarrow \) detect a maximum of \( T_{\text{maxFeatures}} \) SURF features from image \( I \) with SURF feature response over \( T_{\text{response}} \)
3: \( d \leftarrow \) extract SURF descriptors from \( I \) with features \( f \)
4: \( z \leftarrow \) quantize descriptors \( d \) to dictionary (using \( T_{\text{NNDR}} \))
5: \( L \leftarrow \) create location with signature \( z \) and weight 0
6: return \( L \)
7: end procedure

B. Weight Update

To update the weight of the acquired location, \( L_t \) is compared to ones in STM from the oldest to the more recent ones, and similarity \( s \) is evaluated using (1):

\[
s(z_t, z_c) = \begin{cases} 
N_{\text{pair}}/N_{z_t}, & \text{if } N_{z_t} \geq N_{z_c} \\
N_{\text{pair}}/N_{z_c}, & \text{if } N_{z_t} < N_{z_c}
\end{cases}
\]

(1)

where \( N_{\text{pair}} \) is the number of matched word pairs between the compared location signatures, and where \( N_{z_t} \) and \( N_{z_c} \) are the total number of words of signature \( z_t \) and the compared signature \( z_c \) respectively. If \( s(z_t, z_c) \) is higher than a fixed similarity threshold \( T_{\text{similar}} \) (ratio between 0 and 1), the compared location \( L_c \) is merged into \( L_t \). Only the words from \( z_c \) are kept in the merged signature: \( z_t \) is cleared and \( z_c \) is copied into \( z_t \). To complete the merging process, the weight of \( L_t \) is increased by the weight of \( L_c \) plus one, the neighbor and loop closure links of \( L_c \) are redirected to \( L_t \), and \( L_c \) is deleted from STM.

C. Bayesian Filter Update

The role of the discrete Bayesian filter is to keep track of loop closure hypotheses by estimating the probability that the current location \( L_t \) matches one of an already visited location stored in the WM. Let \( S_t \) be a random variable representing the states of all loop closure hypotheses at time \( t \). \( S_t = i \) is the probability that \( L_t \) closes a loop with a past location \( L_i \), thus detecting that \( L_i \) and \( L_t \) represent the same location. \( S_t = -1 \) is the probability that \( L_t \) is a new location. The filter estimates the full posterior probability \( p(S_t|L_t^t) \) for all \( i = -1, \ldots, t_n \), where \( t_n \) is the time index associated with the newest location in WM, expressed as follows [29]:

\[
p(S_t|L_t^t) = \eta \ p(L_t|S_t) \sum_{i=-1}^{t_n} p(S_t|S_{t-1} = i) p(S_{t-1} = i|L_{t-1}^t)
\]

where \( \eta \) is a normalization term and \( L_l^t = L_{t-1}, \ldots, L_t \). Note that the sequence of locations \( L_l^t \) includes only the locations contained in WM and STM. Therefore, \( L_l^t \) changes over time as new locations are created or when some locations are retrieved from LTM or transferred to LTM, in contrast to the classical Bayesian filtering where such sequences are fixed.

The observation model \( p(L_t|S_t) \) is evaluated using a likelihood function \( \mathcal{L}(S_t|L_t) \) : the current location \( L_t \) is compared using (1) with locations corresponding to each loop closure state \( S_t = j \) where \( j = 0, \ldots, t_n \), giving a score \( s_j = s(z_t, z_j) \). The difference between each score \( s_j \) and the standard deviation \( \sigma \) is then normalized by the mean \( \mu \) of all non-null scores, as in (3) [29] :

\[
p(L_t|S_t = j) = \mathcal{L}(S_t = j|L_t) = \begin{cases} 
\frac{s_j - \mu}{\sigma}, & \text{if } s_j \geq \mu + \sigma \\
1, & \text{otherwise}
\end{cases}
\]

(3)

For the new location probability \( S_t = -1 \), the likelihood is evaluated using (4) :

\[
p(L_t|S_t = -1) = \mathcal{L}(S_t = -1|L_t) = \frac{\mu}{\sigma} + 1
\]

(4)

where the score is relative to \( \mu \) on \( \sigma \) ratio. If \( \mathcal{L}(S_t = -1|L_t) \) is high (i.e., \( L_t \) is not similar to a particular location in WM, as \( \sigma < \mu \)), then \( L_t \) is more likely to be a new location.

The transition model \( p(S_t|S_{t-1} = i) \) is used to predict the distribution of \( S_t \), given each state of the distribution \( S_{t-1} \) in accordance with the robot’s motion between \( t \) and \( t - 1 \). Combined with \( p(S_{t-1} = i|L_{t-1}^t) \) (i.e., the recursive part of the filter), this constitutes the belief of the next loop closure.

The transition model is expressed as in [29]:

1) \( p(S_t = -1|S_{t-1} = -1) = 0.9 \), the probability of a new location event at time \( t \) given that no loop closure occurred at time \( t - 1 \).
2) \( p(S_t = i|S_{t-1} = -1) = 0.1/N_{\text{WM}} \) with \( i \in [0; t_n] \), the probability of a loop closure event at time \( t \) given that no loop closure occurred at \( t - 1 \). \( N_{\text{WM}} \) is the number of locations in WM of the current iteration.
3) \( p(S_t = -1|S_{t-1} = j) = 0.1 \) with \( j \in [0; t_n] \), the probability of a new location event at time \( t \) given that a loop closure occurred at time \( t - 1 \) with \( j \).
4) \( p(S_t = i|S_{t-1} = j) \) with \( i, j \in [0; t_n] \), the probability of a loop closure event at time \( t \) given that a loop closure occurred at time \( t - 1 \) on a neighbor location. The probability is defined as a discretized Gaussian curve (\( \sigma = 1.6 \)) centered on \( j \) and where values are non-null for a neighborhood range of sixteen neighbors (for \( i = j - 16, \ldots, j + 16 \)). Within the graph, a location can have more than two adjacent neighbors (if it has a loop closure link) or some of them are not in WM (because they were transferred to LTM). The Gaussian’s values are set recursively by starting from \( i = j \) to
the end of the neighborhood range (i.e., sixteen), then
\[ p(S_t = i|L_j) \geq 0.9 \]

D. Loop Closure Hypothesis Selection

When \( p(S_t|L_i) \) has been updated and normalized, the highest loop closure hypothesis \( S_t = i \) of \( p(S_t|L_i) \) is accepted if (5) is satisfied, with the loop closure threshold \( T_{\text{loop}} \) set between 0 and 1. When a loop closure hypothesis is accepted, \( L_t \) is linked with the old location \( L_i \); the weight of \( L_t \) is increased by the one of \( L_i \), the weight of \( L_i \) is reset to 0, and a loop closure link is added between \( L_t \) and \( L_i \). The loop closure link is used to get neighbors of the old location during Retrieval (Section III-E) and to setup the transition model of the Bayes filter (Section III-C). Note that this hypothesis selection differs from our previous work [22]: the old parameter \( T_{\text{minHyp}} \) is no longer required and locations are not merged anymore on loop closures (only a link is added). Not merging locations helps to keep different signatures of the same location for better hypothesis estimation, which is important in a highly dynamic environment or when the environment changes gradually over time in a cyclic way (e.g., day-night, weather variations or seasons).

\[
\sum_{i=0}^{t_n} p(S_t = i|L_i) > T_{\text{loop}}
\]

E. Retrieval

After loop closure detection, neighbors not in WM of the location with the highest loop closure hypothesis are transferred back from LTM to WM. In this work, LTM is implemented as a SQLite3 database, following the schema illustrated in Fig. 3. In the link table, the \textit{link type} tells if it is a neighbor link or a loop closure link. When locations are retrieved from LTM, the visual dictionary is updated with the words associated with the corresponding retrieved signatures. Common words from the retrieved signatures still exist in the dictionary; therefore, a reference is added between these words and the corresponding signatures. For words that are no longer present in the dictionary (because they were removed from the dictionary when the corresponding locations were transferred [ref. Section III-F]), they are matched (using the same \( T_{\text{NNDR}} \) criterion of Section III-A) to those in the dictionary to check if more recent words represent the same SURF descriptors. This step is important because the new words added from the new signature \( z_t \) may be identical to the previously transferred words. For matched words, the old words are replaced by the new ones in the retrieved signatures. However, all references in LTM are not immediately changed because this operation is expensive in terms of computational time. Instead, they are changed as other signatures are retrieved, and the old words are permanently removed from LTM when the system is shut down. If some words are still unmatched, they are simply added to dictionary.

Algorithm 3 summarizes the Retrieval process. Because loading locations from the database is time consuming, a maximum of two locations are retrieved at each iteration (chosen inside the neighboring range defined in Section III-C). When more than two locations can be retrieved, nearby locations in time (direct neighbors of the hypothesis) are prioritized over nearby locations in space (neighbors added through loop closures). In Fig. 1 for instance, if location 116 is the highest loop closure hypothesis, location 118 will be retrieved before location 23. This order is particularly important when the robot is moving, where retrieving next locations in time is more appropriate than those in space. However, if the robot stays still for some time, all nearby locations in time will be retrieved, followed by nearby locations in space (i.e., 23, 24, 22, 25).

![Database representation of the LTM.](image)

**Algorithm 3 Retrieve neighbors of \( L \) from LTM to WM**

1: \textbf{procedure} RETRIEVAL(L)
2: \( L_r[] \leftarrow \text{load a maximum of two neighbors of } L \text{ from LTM with their respective signatures } z_r[] \)
3: \( \text{Add references to } L_r[] \text{ for words in } z_r[] \text{ still in dictionary} \)
4: \( \text{Match old words (not anymore in dictionary) of } z_r[] \text{ to current ones in dictionary (using } T_{\text{NNDR}} \text{ criterion)} \)
5: \( \text{Not matched old words of } z_r[] \text{ are added to dictionary} \)
6: \( \text{Insert } L_r[] \text{ into WM} \)
7: \textbf{end procedure}

F. Transfer

When processing time for an image is greater than \( T_{\text{time}} \), the oldest locations of the lowest weighted ones are transferred from WM to LTM. To be able to evaluate appropriately loop closure hypotheses using the discrete Bayesian filter, neighbors of the highest loop closure hypothesis are not allowed to be transferred. The number of these locations is however limited to the finite number of nearby locations in time (accordingly to neighboring range defined in Section III-C), to avoid ‘immunizing’ all nearby locations in space (which are indefinite in terms of numbers). \( T_{\text{time}} \) is set empirically to allow the robot to process the perceived images in real-time. Higher \( T_{\text{time}} \) means that more locations (and implicitly more words) can remain in WM, and more loop closure hypotheses can be kept to better represent the overall environment. \( T_{\text{time}} \) must therefore be determined according to the robot’s CPU capabilities, computational load and operating environment. If \( T_{\text{time}} \) is determined to be higher than the image acquisition time, the algorithm intrinsically uses an image rate corresponding to \( T_{\text{time}} \), with 100% CPU usage.

Because the most expensive step of RTAB-Map is to rebuild the visual dictionary (line 2 of Algorithm 2), processing time per acquired image can be regulated by changing the dictionary size, which indirectly influences the WM size. Algorithm 4 presents how the visual dictionary is modified during the Transfer process. A signature of a location transferred to LTM removes its word references from the visual dictionary.
If a word does not have reference to a signature in WM anymore, it is transferred into LTM. While the number of words transferred from the dictionary is less than the number of words added from $L_t$ or the retrieved locations, more locations are transferred from WM to LTM. At the end of this process, the dictionary size is smaller than before the new words from $L_t$ and retrieved locations were added, thus reducing the time required to update the dictionary for the next acquired image. Saving the transferred locations into the database is done asynchronously using a background thread, leading to a minimal time overhead for the next iteration when joining the thread on line 17 of Algorithm 1 ($pTime$ then includes transferring time).

The way to transfer locations into LTM influences long-term operation when WM reaches its maximum size, in particular when a region (a set of locations) is seen more often than others. Normally, at least one location in a new region needs to receive a high weight through the Weight Update to replace an old and high weighted one in WM, in order to detect loop closures when revisiting this region. However, if there is no location in the new region that receives a high weight, loop closures could be impossible to detect unless the robot comes back to a high weighted location in an old region, and then moves from there to the new one. To handle this situation and to improve from our previous work [22], a subset of the highest weighted locations (defined by $T_{recent} \times N_{WM}$) after the last loop closure detected are not allowed to be transferred to LTM. This way, when the robot explores a new region, there are always high weighted locations of this region in WM until a loop closure is detected. If the number of locations after the last loop closure detected in WM exceeds $T_{recent} \times N_{WM}$, these locations can be transferred like the ones in WM (i.e., with the criterion of the oldest of the lowest weighted locations). $T_{recent}$ is a ratio fixed between 0 and 1. A high $T_{recent}$ means that more locations after the last loop closure detected are kept in WM, which also leads to a transfer of a higher number of old locations to LTM.

**Algorithm 4** Transfer locations from WM to LTM

1:   procedure TRANSFER( )
2:      nwt ← 0  \> number of words transferred
3:      nwa ← number of new words added by $L_t$ and retrieved locations
4:   while nwt < nwa do
5:      $L_t$ ← select a transferable location in WM (by weight and age), ignoring retrieved locations and those in recent WM (using $T_{recent}$)
6:      Move $L_t$ to trash
7:      Move words $w_i$ which have no more references to any locations in WM to trash
8:      nwt ← nwt + SIZE($w_i$)
9:   end while
10:   Start trash’s thread to empty trash to LTM
11: end procedure

**IV. RESULTS**

Performance of RTAB-Map is evaluated in terms of precision-recall metrics [30]. Precision is the ratio of true positive loop closure detections to the total number of detections. Recall is defined as the ratio of true positive loop closure detections to the number of ground truth loop closures. To situate what can be considered good recall performance, for metric SLAM, recall of around 20% to 30% at 100% precision (i.e., with no false positives) is sufficient to detect most loop closure events when the detections have uniform spatial distributions [30]. Note however that the need to maximize recall depends highly on the SLAM method associated with the loop closure detection approach. If metric SLAM with excellent odometry is used, a recall ratio of about 1% could be sufficient. For less accurate odometry (and even no odometry), a higher recall ratio would be required.

Using a MacBook Pro 2.66 GHz Intel Core i7 and a 128 Gb solid state hard drive, experimentation is done on ten community data sets and two custom data sets using parameters presented in Table I. These parameters were set empirically over all data sets to give good overall recall performances (at precision of 100%), and remained the same (if not otherwise stated) to evaluate the adaptability of RTAB-Map. The only SURF parameter changed between experiments is $T_{response}$, which is set based on the image size. $T_{time}$ is set accordingly to image rate of the data sets. As a rule of thumb, $T_{time}$ can be about 200 to 400 ms smaller than the image acquisition rate at 1 Hz, to ensure that all images are processed under the image acquisition rate, and even if the processing time goes over $T_{time}$ ($T_{time}$ then corresponds to the average processing time of an image by RTAB-Map). So, for an image acquisition rate of 1 Hz, $T_{time}$ can be set between 600 ms to 800 ms. For each experiment, we identify the minimum $T_{loop}$ that maximizes recall performance at 100% precision. With the use of these bounded data sets, the LTM database could have been placed directly in the computer RAM, but it was located on the hard drive to simulate a more realistic setup for timing performances. Processing time $pTime$ is evaluated without the SURF features extraction step (lines 3-4 of Algorithm 2), which is done by the camera’s thread in the implementation.

**A. Community Data Sets**

We conducted tests with the following community data sets: NewCollege (NC) and CityCentre (CiC) [16]; Lip6Indoor (L6I) and Lip6Outdoor (L6O) [29]; 70 km [30]; New-CollegeOmini (NCO) [43]; CrowdedCanteen (CrC) [44]; BicoccaIndoor-2009-02-25b (BI), BovisaOutdoor-2008-10-04 (BO) and BovisaMixed-2008-10-06 (BM) [45] data sets. NC and CiC data sets contain images acquired from two cameras (left and right), totaling 2146 and 2474 images respectively of size $640 \times 480$. Because RTAB-Map takes only one image as input in its current implementation, the images from the two cameras were merged into one, resulting in 1073 and 1237

---

**TABLE I**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{STM}$</td>
<td>30</td>
</tr>
<tr>
<td>$T_{similarity}$</td>
<td>20%</td>
</tr>
<tr>
<td>$T_{recent}$</td>
<td>20%</td>
</tr>
</tbody>
</table>

**SURF Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension</td>
<td>64</td>
</tr>
<tr>
<td>$T_{NDIR}$</td>
<td>0.8</td>
</tr>
<tr>
<td>$T_{maxFeatures}$</td>
<td>400</td>
</tr>
<tr>
<td>$T_{head}$</td>
<td>0.2</td>
</tr>
</tbody>
</table>
images respectively of size $1280 \times 480$. For data sets with an image rate depending on vehicle movement or over 2 Hz, some images were removed to have approximately an image rate of 1 Hz (i.e., keeping 5511 of the 9575 panoramic images for the 70 km data set). For NCO and CrD data sets, because they contain panoramic images taken by a vehicle slower than for the 70 km data set, $T_{\text{similarity}}$ is increased to 35% (compared to 20% for all other data sets). $T_{\text{time}}$ is set to 1.4 s for images acquired every 2 seconds (0.5 Hz), to 0.7 s for images acquired every second (1 Hz) and to 0.35 s for images acquired every half second (2 Hz).

Table II summarizes experimental conditions and results. Recall performance corresponds to the maximum recall performance observed at 100% precision, and precision-recall curves are shown in Fig. 4. Compared to other approaches that also used these data sets, RTAB-Map achieves better recall performances at 100% precision while respecting real-time constraints: the maximum processing time $\mu T_{\text{time}}$ is always under the image acquisition time.

### B. Université de Sherbrooke (UdeS) Data Set

The data set used for this test is made of images taken over a 2 km loop of the Université de Sherbrooke (UdeS) campus, traversed twice, as illustrated by Fig. 5. A total of 5395 images of $640 \times 480$ resolution at 1 Hz are used from the original capture (21 582 images at 4 Hz) with a handheld webcam, over 90 minutes. The data set contains a variety of environment conditions: indoor and outdoor, roads, parkings, pedestrian paths, trees, a football field, with differences in illumination and camera orientation. To study the ability of RTAB-Map to transfer and to retrieve locations based on their occurrences, we stopped at 13 waypoints during which the camera remained still for around 20 to 90 seconds, leading to 20 to 90 images of the same location. After processing the images of the first traversal, it is expected that the related locations will still be in WM and that RTAB-Map will be able to retrieve nearby locations from LTM to identify loop closures. $T_{\text{response}}$ is set to 150.

Table III presents results using different $T_{\text{time}}$ to show the effect of memory management on recall performances. When $T_{\text{time}} = \infty$, all locations are kept in WM; therefore, loop closure detection is done over all previously visited locations. The resulting maximum processing time is 13.6 seconds, which makes it impossible to use online in real-time (image acquisition time is 1 sec). With $T_{\text{time}} < 0.75$ s, processing is done in real-time (i.e., processing time is always lower than the image acquisition time).

For $T_{\text{time}} \in [0.95; 0.45]$ s, recall varies between 43% and 54%, and this variation is caused by the choice of locations kept in WM: small computation time variations and different $T_{\text{time}}$ explain why some locations are transferred or retrieved in some experiments while they are not in others. Lower recall performance at $T_{\text{time}} = \infty$ is caused by the presence of a large dictionary: as words are added to dictionary, the matching distance between SURF features becomes smaller because of the $T_{\text{NNDR}}$ matching criterion. Although this provides more precise matches, the matching process is more sensitive to noise. Slight changes in illumination add noise in the SURF features extracted, which can explain why a better recall performance is obtained using a smaller dictionary. For $T_{\text{time}} < 0.45$ s, the WM size becomes too small, and RTAB-Map is unable to detect as many loop closures: when $T_{\text{time}}$ is reached (because the retrieved locations cannot be immediately transferred), old locations with large weights are transferred instead (including those representing the waypoints), making it difficult to detect loop closures if Retrieval does not bring back appropriate locations.

To illustrate more closely the results obtained for $T_{\text{time}} = 0.7$ s, at the end of the trial each waypoint is represented in WM by at least one location with a high weight. The other locations with smaller weights that are still in WM are the ones where the images did not change much over time (like the football field). At its maximum, the dictionary has 56K visual words for 250 locations (220 in WM + 30 in STM). With $T_{\text{maxFeatures}} = 400$, there are then about 225 unique words and 175 common words per location. The high number of unique words is attributable to $T_{\text{NNDR}}$ criterion on feature quantization. Fig. 5 illustrates recall performance over the 2 km loop, categorized using three colored paths:

---

**Fig. 4.** Precision-recall curves for each data set.

**Fig. 5.** UdeS data set aerial view. The first traversal is represented by the dotted line. The second traversal is represented by a line located around the first. The start/end point is represented by the circle. The small white dots in the waypoint ID numbers represent camera orientation at this location. Recall performance is from the test case with $T_{\text{time}} = 0.7$ s.
TABLE II
EXPERIMENTAL CONDITIONS AND RESULTS OF RTAB-MAP ON COMMUNITY DATA SETS

<table>
<thead>
<tr>
<th>Data set</th>
<th>NC</th>
<th>CiC</th>
<th>L6I</th>
<th>L6O</th>
<th>70 km</th>
<th>NCO</th>
<th>CrC</th>
<th>BI</th>
<th>BO</th>
<th>BM</th>
</tr>
</thead>
<tbody>
<tr>
<td># images</td>
<td>1073</td>
<td>1237</td>
<td>388</td>
<td>531</td>
<td>1626</td>
<td>692</td>
<td>1757</td>
<td>2277</td>
<td>2147</td>
<td></td>
</tr>
<tr>
<td>Image size</td>
<td>1280x480</td>
<td>1280x480</td>
<td>240x192</td>
<td>240x192</td>
<td>1600x460²</td>
<td>2048x618</td>
<td>480x270</td>
<td>320x240</td>
<td>320x240</td>
<td></td>
</tr>
<tr>
<td>Image rate</td>
<td>≈0.5 Hz</td>
<td>≈0.5 Hz</td>
<td>1 Hz</td>
<td>0.5 Hz</td>
<td>1 Hz</td>
<td>0.5 Hz</td>
<td>1 Hz</td>
<td>0.5 Hz</td>
<td>1 Hz</td>
<td></td>
</tr>
<tr>
<td>$T_{response}$ (s)</td>
<td>1000</td>
<td>1000</td>
<td>10</td>
<td>10</td>
<td>1000</td>
<td>1000</td>
<td>75</td>
<td>50</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>$T_{time}$ (s)</td>
<td>1.4</td>
<td>1.4</td>
<td>0.7</td>
<td>1.4</td>
<td>0.7</td>
<td>0.7</td>
<td>0.35</td>
<td>0.7</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>Max $pTime$ (s)</td>
<td>1.77</td>
<td>1.76</td>
<td>0.72</td>
<td>1.73</td>
<td>0.94</td>
<td>0.83</td>
<td>0.43</td>
<td>0.83</td>
<td>0.84</td>
<td></td>
</tr>
<tr>
<td>Max dict. size $\times 10^3$</td>
<td>111</td>
<td>113</td>
<td>52</td>
<td>111</td>
<td>56</td>
<td>56</td>
<td>27</td>
<td>57</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>Max WM size</td>
<td>415</td>
<td>388</td>
<td>333</td>
<td>372</td>
<td>160</td>
<td>262</td>
<td>98</td>
<td>220</td>
<td>208</td>
<td></td>
</tr>
<tr>
<td>Min $T_{loop}$</td>
<td>0.09</td>
<td>0.08</td>
<td>0.12</td>
<td>0.07</td>
<td>0.12</td>
<td>0.11</td>
<td>0.08</td>
<td>0.53</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>Precision (%)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Recall (%) from other approaches [ref. #]

TABLE III
EXPERIMENTAL RESULTS OF RTAB-MAP FOR THE UDE'S DATA SET (5395 IMAGES OF SIZE 640x480 TAKEN AT 1 Hz)

<table>
<thead>
<tr>
<th>$T_{time}$ (s)</th>
<th>$\infty$</th>
<th>0.95</th>
<th>0.90</th>
<th>0.85</th>
<th>0.80</th>
<th>0.75</th>
<th>0.70</th>
<th>0.65</th>
<th>0.60</th>
<th>0.55</th>
<th>0.50</th>
<th>0.45</th>
<th>0.40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max $pTime$ (s)</td>
<td>13.6</td>
<td>1.23</td>
<td>1.15</td>
<td>1.10</td>
<td>1.03</td>
<td>0.98</td>
<td>0.92</td>
<td>0.87</td>
<td>0.83</td>
<td>0.77</td>
<td>0.71</td>
<td>0.60</td>
<td>0.52</td>
</tr>
<tr>
<td>Max dict. size $\times 10^3$</td>
<td>712</td>
<td>76</td>
<td>72</td>
<td>69</td>
<td>65</td>
<td>60</td>
<td>56</td>
<td>52</td>
<td>48</td>
<td>44</td>
<td>40</td>
<td>36</td>
<td>32</td>
</tr>
<tr>
<td>Max WM size</td>
<td>2875</td>
<td>300</td>
<td>274</td>
<td>265</td>
<td>258</td>
<td>238</td>
<td>220</td>
<td>200</td>
<td>178</td>
<td>164</td>
<td>153</td>
<td>133</td>
<td>115</td>
</tr>
<tr>
<td>Min $T_{loop}$</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Precision (%)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Recall (%)</td>
<td>39</td>
<td>54</td>
<td>47</td>
<td>49</td>
<td>48</td>
<td>49</td>
<td>53</td>
<td>54</td>
<td>50</td>
<td>47</td>
<td>42</td>
<td>43</td>
<td>16</td>
</tr>
</tbody>
</table>

Fig. 6. Examples (with visual words represented as circles) of loop closure hypotheses that are insufficient (under $T_{loop}$) to detect loop closures, caused by changes in illumination conditions (a) or camera orientation (b and c). New words are colored in green or yellow, and words already in the dictionary are colored in red or blue.

Fig. 7. Processing time in terms of locations processed over time with the UdeS data set. $T_{time}$ is set at 0.7 s, as shown by the horizontal line.

- **Green** paths identify valid loop closure hypotheses. Ground truth was labeled manually, based on similar visual appearance and proximity between images. The images from the second traversal are not taken at exactly the same position and the same angle compared to the first traversal. Therefore, a match within a margin of 10 locations is considered acceptable for loop closure.

- **Yellow** paths indicate an insufficient loop closure probability under $T_{loop}$. However, a Yellow path also means that Retrieval works as expected, i.e., RTAB-Map is able to retrieve appropriate transferred signatures (those near ground truth loop closures) as previously visited locations are encountered. Fig. 6 a) illustrates an example: a change in illumination conditions from the first traversal (top image) caused a change in the visual words extracted from the image of the second traversal (bottom image). Most of the words extracted in the top image are from the tree on the left, compared to the building section in the bottom image.

- **Red** paths identify when there is no location in WM which could be matched with the current location. A transition from a Yellow to a Red path occurs as follows. The likelihood of the observed location with the corresponding location of the first traversal still in WM is too low (i.e., words extracted are too different or are too common) and is lower than with other locations in the map. Because the associated loop closure hypothesis is not the highest one anymore, nearby locations of the real loop closure cannot be retrieved from LTM to

---

2The 5-view omni-directional images were stitched using The Panorama Factory software.

Preprint submitted to IEEE Transactions on Robotics. Received: April 23, 2012 11:11:20 PST
WM. Therefore, the next observed locations do not have any locations in WM which can be used to find loop closures. Fig. 6 b) and c) illustrate what happens at the beginning of two Red paths (before the waypoints 6 and 10 respectively): over several consecutive images, the camera was not oriented in the same direction as in the first traversal, and RTAB-Map was not able to retrieve neighbor locations from LTM because the new acquired locations were more similar to locations in another part of the map. However, the wrong loop closure hypotheses (or false positives) during a Red path stayed low under $T_{\text{loop}}$, and thus they were not accepted. A transition from a Red to a Yellow path happens when the camera returns to a location still in WM, increasing the associated loop closure hypothesis to become the highest and resulting in a retrieval of neighbor locations from which loop closures can be detected.

- **Other** indicates paths different from the ones taken during the first traversal, and therefore there are no loop closures to find.

Finally, Fig. 7 shows the processing time for each acquired image with $T_{\text{time}} = 0.7\text{s}$. As expected, once the processing time has reached $T_{\text{time}} = 0.7\text{s}$ (after $\sim 500$ images), the memory management is triggered and the processing time remains close to $T_{\text{time}}$, with an average processing time of 0.67$s$ and a maximum processing time of 0.92$s$.

C. “Need for Speed: Most Wanted” (NFSMW) Data Set

For this experiment, the video game “Need for Speed: Most Wanted” (NFSMW) was used to acquire images by driving around (while respecting speed limits) city Area 1 about one hundred times and then moving to Area 2 for another hundred

Fig. 8. Map of NFSMW data set showing Area 1 and Area 2.

![Fig. 8. Map of NFSMW data set showing Area 1 and Area 2.](image)

Fig. 9. Samples of the NFSMW data set. In a), the sun shines come from the east (bottom) or the west (top); note the high color and contrast differences for the left and right buildings caused by the dynamic shadows. In b), four different atmospheric conditions are shown for the same location (over 30 minutes).

![Fig. 9. Samples of the NFSMW data set.](image)

traversals. Fig. 8 illustrates these two areas. This data set was generated to evaluate RTAB-Map performance in two specific conditions:

1) Frequently observing the same locations;
2) Moving to new locations after observing the same area for a long period of time.

Even though the environment is synthetic, there are many large changes in illumination conditions (sun and shadows move slowly; there are also bright sunrises and random storms) that makes it very challenging for loop closure detection over long-term operation. Fig. 9 illustrates examples of such changes. A total of 25098 images of $640\times 480$ size were taken at 1 Hz, totaling 7 hours of driving. Because of the presence of the head-up display in the images and that the lower portions are generally made of common and repetitive road textures, the upper 10% and lower 40% of the images are not used for SURF features extraction. $T_{\text{response}}$ is set to 150.

The upper portion of Fig. 10 illustrates recall performances computed for each traversal. The performances for the 102 traversals of Area 1 and the 103 traversals of Area 2 are delimited by the red vertical line. Recall of 0% is observed for the first traversal of Area 1 and the first traversal of Area 2, as expected. Recall variations are caused by changes in illumination conditions during the traversals: if a storm happens when sunny locations are retrieved, loop closures are not detected until the storm finishes or a darker version of the encountered locations are retrieved. Generally, Red paths end on road intersections, which corresponds to locations with higher weights (caused by having the vehicle stop) and where neighbors (by loop closures) with different illumination conditions are retrieved. For Area 1 and Area 2, recall at 100% precision varies from 61% to 100% and from 57% to 100%, respectively.

The lower portion of Fig. 10 shows recall performances over time at 100% precision, considering all loop closures detected from $t = 0$. After encountered most of the changes in illumination conditions between $t = 0$ to $t = 5000$ (which corresponds to 41 traversals), the average recall performance
stabilizes around 90%, for a minimum $T_{loop} = 0.10$. With a ground truth of 24800 loop closures and 90% recall, there are 2480 duplicated locations in the global graph of the 280 locations of Area 1 and Area 2. These duplicated locations create new paths in the global map. Fig. 1 illustrates such a case: locations 453 and 454 are duplicates of 114 and 115 respectively because the environment changed too much; location 455 then have two paths representing the same real location. In practice, it is likely that one of the paths will eventually be transferred in LTM, keeping only one version of the real location in WM. However, keeping two paths representing the same locations in WM may be beneficial, especially in dynamical environments with cyclic atmospheric changes like in this data set: locations in dark conditions could have almost no features similar to their versions in bright conditions, then loop closures are found alternately between dark and bright versions.

Looking more closely at the transition between the two areas, when moving to Area 2, the WM only contains locations of Area 1 with large weights. If a set of the new locations would not have been kept in the recent part of WM (as explained in Section III-F), loop closures would have been impossible to detect if no location received a high weight (from Weight Update) to replace locations in WM from Area 1. After the first traversal of the Area 2, the recent part of WM was populated mostly by locations representing road intersections (having higher weights because the vehicle stopped). The first loop closure detected on the second traversal was found on the first intersection encountered during the first traversal of Area 2. Next locations were then retrieved, and a recall of 100% at 100% precision was achieved for the second traversal.

Regarding processing time, once $T_{time} = 0.7 \text{s}$ is reached for the first time after 201 images processed, a mean time of 0.71 s is achieved for the rest of the experiment. The maximum processing time is 0.89 s, thus respecting the real-time constraint of 1 Hz. At the end of the experiment, there were 17 locations of the Area 1 and 39 locations of the Area 2 with high weights in WM, distributed mainly on road intersections.

V. DISCUSSION

Results presented in Section IV suggest that RTAB-Map can achieve good recall performances in real-time at 100% precision over diverse and large-scale environments. Real-time constraints can be satisfied independently of the scale of the environment, which is very important for long-term online mapping.

Overall, using similarity occurrences reveal to be a simple and functional method to determine which locations to keep in WM. Obviously, it has limitations when an area is seen only one time before moving to a new area for a long time. To illustrate, we conducted a trial using the NFSMW setup by doing only one traversal of Area 1 and then moving for one hundred traversals of Area 2. After the first traversal of Area 1, the highest weight of a location is 2. After 33 traversals of Area 2, all locations of Area 1 were transferred to LTM (in comparison to 17 locations remaining in WM after 103 traversals of Area 2 in Section IV-C). The number of traversals of Area 2 required to transfer all locations of Area 1 in LTM depends on the weight assigned to locations during the first traversal. Returning to Area 1 then leads to the creation of duplicated locations that cannot be associated to the locations of Area 1 stored in LTM (unless they were revisited backward from the entry point of Area 2), and these duplicated locations, if visited frequently, can remain in WM to be used in future loop closures. This illustrates the compromise to be made to satisfy real-time constraints: it may happen that infrequently visited locations get transferred to LTM without being able to be remembered back, but at the same time such locations are not used in the loop closure detection process, allowing to speed up the process using only locations that have more chance to be revisited. Such compromise is therefore driven by the environment and the experiences of the robot. Note that other methods to assign weights to locations could be imagined, such as having the system learn which locations are important (assigning directly a high weight to these locations) based on events, the robot’s internal states, or even from user inputs.

In RTAB-Map, LTM’s growth influences loop closure detection performance over large-scale and long-term operation. To understand such influence, let’s define $w_w$, the number of words in WM and STM (i.e., the visual dictionary), $n_w$, the number of locations in WM, $w_l$, the number of words in LTM and $n_l$, the number of locations in LTM. Time complexity for each step of Algorithm 1 is given as follows:

- **Location Creation:** rebalancing the dictionary is $O(w_w \log w_w)$ and features quantization is $O(\log w_w)$. The SURF features extraction can be considered $O(1)$ as image size is fixed.
- **Weight Update:** updating weight is $O(1)$ as the number of locations in STM is fixed.
- **Bayesian Filter Update:** computing observation is $O(n_w)$ and belief if $O(n_w^2)$.
- **Loop Closure Hypothesis Selection:** hypothesis selection is $O(n_w)$.
- **Retrieval:** words quantization is $O(\log w_w)$. Database selection query is $O(\log w_w + n_l)$. Database insertion query is $O(\log w_w + n_l)$.

When $T_{time}$ is reached, WM size remains fixed, bounding time complexities associated to $w_w$ and $n_w$. However, for Retrieval and Transfer, time complexities also depend on LTM, and LTM size is not bounded. With the growth of LTM, $T_{time}$ is more likely to be reached, and WM size will gradually decrease over time to satisfy real-time constraints. Theoretically, WM size may eventually become null, disabling loop closure detection. In practice, though, the logarithmic growth in time complexity caused by LTM is very small and WM size is not affected. For the NFSMW experiment, top of Fig. 11 shows the total database access time required to retrieve and to transfer locations for each RTAB-Map iteration, for up to 7 hours of use. Time growth is unnoticeable. At the end of experiment, the database size is 3.1 GB with 6.3 million words and 25098 locations, and all merged and bad locations were also kept in the database for debugging purpose (this would
also have influenced unfavorably the WM size). The bottom of Fig. 11 shows the WM size over time. The higher variations of WM size after around 12000 locations are mainly caused by environmental changes from Area 1 to Area 2, and not because of database access time: the mean of WM size remains the same as for Area 1. If necessary, a solution to LTM size would be to limit database growth by permanently removing offline some locations from the database. For instance, paths leading to the same high weighted locations could be eliminated based on the sum of the weights of locations in the paths. If the number of distant high weighted locations gets very high, important locations could ultimately be deleted, resulting in a dismembered global map (i.e., many disconnected smaller maps) if weight is the primary transfer criterion. Pruning the oldest locations of the database (independently of the weight) may be better to preserve a unique global map, at the cost of forgetting important old locations.

In dynamic environments, the performance of RTAB-Map is also highly dependent on the quality of the SURF features extracted. We observed in our experiments that SURF features are relatively sensible to changes in illumination and shadows, reducing the number of discriminative features for more “garbage features” (or common words) in images. In RTAB-Map, at least one discriminative feature in the environment is required to find a loop closure, but if there are many “garbage features”, this means that other locations also receive high likelihoods, thus shadowing the weight of the discriminative feature. In this case, Eq. 4 scores high (for a new location probability) because the standard deviation of the likelihood scores is small comparatively to the mean, thus loop closure is not found because the likelihood for the appropriate loop closure is smaller than for the new location probability. Feature weighting may help in such cases by assigning a high weight to discriminative features and a lower weight to “garbage features”. However, we think that doing so would lead to more false positives. By considering all features with the same weight in RTAB-Map, many discriminative features are required for a location to score higher than others if many “garbage features” are present, decreasing the chance of false positives. We prefer avoiding to find a loop closure in such condition (like in environments populated with many dynamic objects or people) rather than increasing the probability to accept a false positive.

As additional improvements, exploiting sparseness of the Bayesian filter [30], more efficient dictionary structures (other than kd-trees) or better ways to rebalance a dictionary may be used to speed up the process, but our focus in this paper is on real-time constraints satisfaction (i.e., what should be done when computation time reaches the real-time limit), and not optimizing complexities depending on WM size. Finally, to overcome the occurrences of Red paths caused by changes in camera orientation (see Section IV-B), active localization could be triggered by detecting decreasing hypotheses, which could make the system move the camera in the right direction to let RTAB-Map retrieve appropriate old locations from LTM to WM.

VI. CONCLUSION

Results presented in this paper suggest that RTAB-Map, a loop closure detection approach based on a memory management mechanism, is able to meet real-time constraints needed for large-scale and long-term operation. While keeping a relatively constant number of locations in WM, real-time processing is achieved for each new image acquired. Retrieval is a key feature that allows RTAB-Map to reach adequate recall ratio even when transferring a high proportion of the perceived locations in LTM, which are not used for loop closure detection. In future work, in addition to possible extensions outlined in Section V, we plan to study how RTAB-Map can be combined to other approaches to implement a complete Simultaneous Localization and Mapping system.

REFERENCES


Mathieu Labbé received the B.Sc.A. degree in computer engineering and the M.Sc.A. degree in electrical engineering from the Université de Sherbrooke, Sherbrooke, Quebec Canada, in 2008 and 2010, respectively. He is currently working toward the Ph.D. degree in electrical engineering at the same university. His research interests include computer vision, autonomous robotics and robot learning.

François Michaud (M’90) received his bachelor’s degree (’92), Masters degree (’93) and Ph.D. degree (’96) in electrical engineering from the Université de Sherbrooke, Québec Canada.

After completing postdoctoral work at Brandeis University, Waltham MA (’97), he became a faculty member in the Department of Electrical Engineering and Computer Engineering of the Université de Sherbrooke, and founded IntRoLab (formerly LA-BORIUS), a research laboratory working on designing intelligent autonomous systems that can assist humans in living environments. His research interests are in architectural methodologies for intelligent decision-making, design of autonomous mobile robots, social robotics, robots for children with autism, robot learning and intelligent systems.

Prof. Michaud held a Canada Research Chair (2001-11) in Mobile Robots and Autonomous Intelligent Systems, and the Director of the Interdisciplinary Institute for Technological Innovation (3IT). He is a member of IEEE, AAAI and OIQ (Ordre des ingénieurs du Québec). In 2003, he received the Young Engineer Achievement Award from the Canadian Council of Professional Engineers.