2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Image-Based Global Localization using VG-RAM Weightless Neural Networks

Lauro J. Lyrio Júnior, Thiago Oliveira-Santos, Avelino Forechi, Lucas Veronese, Claudine Badue, and Alberto F. De Souza

***Abstract*—Mapping and localization are fundamental problems in autonomous robotics. Autonomous robots need to know where they are in their area of operation to navigate through it and to perform activities of interest. In this paper, we propose an Image-Based Global Localization (VibGL) system that uses Virtual Generalizing Random Access Memory Weightless Neural Networks (VG-RAM WNN). For mapping, we employ a VG-RAM WNN that learns the world positions associated with the images captured along a trajectory. During the localization, new images from the trajectory are presented to the VG-RAM WNN, which outputs their positions in the world. We performed experiments with our VibGL system applied to the problem of localizing an autonomous car. Our experimental results show that the system is able to learn large maps (several kilometers in length) of real world environments and perform global localization with median pose precision of about 3m. Considering a tolerance of 10m VibGL is able to localize the car 95% of the time.**

I. Introduction

Mapping and localization are fundamental problems in autonomous robotics. Autonomous robots need to know where they are in their area of operation to navigate through it and to perform activities of interest. Therefore, they need maps of the environment and the ability to localize themselves in these maps using sensor data.

The Simultaneous Localization And Mapping (SLAM) problem [1] consists of organizing and/or transforming sensor data to create precise representations of the environment, i.e. maps, given a series of robot estimated positions and associated sensor data about the environment. A SLAM system addresses not only the localization, but also the mapping problem simultaneously. Many probabilistic approaches have been proposed to solve the mapping, localization, and SLAM problems [1]; however, some instances of these problems are more difficult to solve than others. Global localization, for example, is more challenging than position tracking [1], and mapping and localization are currently harder to perform with cameras than with Light Detection and Ranging (LIDAR) laser systems. Nevertheless, global localization is of fundamental importance for autonomous robotic systems that might be turned on anywhere in their area of operation (e.g., autonomous cars). Because cameras are much cheaper than LIDAR, the development of efficient localization and mapping techniques based on cameras is relevant for more widespread use of these techniques.

In this paper, we present a new Image-Based Global Localization approach employing Virtual Generalizing Random Access Memory (VG-RAM [2]) Weightless Neural Networks (WNN) for mapping and localization, dubbed VibGL (Fig. 1). VibGL efficiently solves the global localization problem using camera images, but it does not map the environment and simultaneously localizes the robot (it is not a SLAM system)[1]. Instead, it firstly learns a map of an environment using camera images and sensors' data, and later, it localizes the robot in this environment using camera images only. Humans are capable of easily memorizing images of places and labels associated with them (road names, addresses, etc.) as well as trajectories defined by sequences of images and corresponding poses. In later moments, they are able to remember labels or poses when the same images are seen again. Similarly, in the mapping phase, VibGL firstly receives images of the environment as well as positions (labels) where the images were captured using other sensors' data. Subsequently, it learns associations between images and positions that are used as a map of the environment. In the localization phase, VibGL receives images of the environment and uses previously acquired knowledge – "the map" – to output the positions representing the places the system believes these images were captured.

We have tested VibGL with a set of mapping and localization experiments using real-world datasets. These datasets consist of data from various sensors acquired systematically during laps performed by an autonomous car in a 3.57 km long circuit. These datasets were constructed for this work and are made publicly available with the corresponding ground-truth. Our results have shown that VibGL achieves 95% precision in real world settings with a tolerance error of about 10 meters.

This paper is organized as follows. After this introduction, in Section II, we present an overview of the related work. In Section III, we present a brief introduction of the neural network we employ, VG-RAM WNN. In Section IV, we explain our method for image-based global localization using VG-RAM WNNs. In Section V, we describe our

Lauro J. Lyrio Júnior, Thiago Oliveira-Santos, Avelino Forechi, Lucas Veronese, Claudine Badue and Alberto F. De Souza are with the Departamento de Informática, Universidade Federal do Espírito Santo, Vitoria, ES, 29075-910, Brazil (phone: +55-27-4009-2138; fax: +55-27-4009-5848; e-mail: alberto@lcad.inf.ufes.br).

This work was supported in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq, Brazil (grants 552630/2011-0, 308096/2010-0, and 314485/2009-0) and Fundação de Amparo à Pesquisa do Espírito Santo – FAPES, Brazil (grant 48511579/2009).

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

experimental methodology and, in Section VI, we analyze our experimental results. Our conclusions and directions for future work follow in Section VII.

Fig. 1. Examples of VibGL results for a full lap around the Universidade Federal do Espírito Santo (UFES). Images on the left and right of each image pair are samples of images received as input and returned as output by VibGL respectively. Red cars denote the poses estimated by VibGL. Green-marked images are samples of true-positives image-pose pairs outputted by VibGL, while the red-marked one is a sample of a false-positive.

II. Related Work

Much of the work in Robotics Vision in the last decade relied in visual features with certain degree of invariance to affine transformations [3, 4] (e.g. rotation, translation, scale) for providing robust landmarks for mapping and localization [5, 6]. Se et al. [5], for instance, developed a vision-based indoor mobile robot SLAM algorithm using stereo and SIFT, while Wolf et al. [6] used invariant features based on image histograms for indoor localization using cameras. Both approaches (and many similar ones) are mainly conceived as map-based indoor localization and may not be suitable for large outdoor environments, as our approach is. In addition, they are not able to perform continuous global localization as VibGL does.

Several more recent work focus in situations in which only the initial position of the robot is given. In the seminal work of Nister et al., for example [7], visual features present in pairs of consecutive video frames are matched and estimation of the camera motion is computed from the feature tracks. This technique (named visual odometry) is very useful to estimate the motion of a mobile system; however, visual odometry does not keep a map of the environment. Davison et al., in another seminal work [8], developed a SLAM algorithm that tracks a large set of image features from monocular or stereo video and builds a 3D map of features. Lategahn et al. [9] also track a large set of features from stereo images using an EKF SLAM and compute dense feature maps using them. A similar approach was proposed by Geiger et al. [10], where a sparse feature matcher in conjunction with a visual odometry algorithm were used for generating maps of consistent 3D point-clouds. In spite of their capabilities for visual odometry and/or map construction, none of these techniques is suitable for continuous global localization.

RatSLAM [11] is a biologically inspired SLAM approach that uses a simplified visual odometry in addition to appearance-based image template-matching for building maps consisting of simulated cells activations. The system performance was evaluated on a 66 km long urban street, with many loops. Results showed that RatSLAM is capable of building maps online, close loops and re-localize through sequences of familiar visual scenes, i.e. it is capable of global localization; however, this global localization requires several image frames.

RatSLAM was tested in conjunction with FAB-MAP [12] that is another appearance-based SLAM. FAB-MAP [13] is similar to our work, since it allows continuous global localization by detecting that an image is similar to a previously learned image. However, FAB-MAP is based in the bag-of-words image retrieval systems developed in the computer vision community [14] and its learning is costly, while VibGL is based on WNN that learns in one shot. SeqSLAM [15] is another state-of-the-art appearance-based SLAM that calculates the best candidate matching of an image within a segment of a sequence of previously seen images. Although this approach can handle normal and extreme conditions in environment appearance even for long running distances, it is not capable of continuous global localization as VibGL is.

III. VG-RAM WNN

VG-RAM WNN is a very effective machine learning technique that offers easy implementation and fast training procedure, thanks to its simplicity [2]. Such neural networks comprise a set of neural layers composed of VG-RAM neurons connected to other layers through synapses.

A basic network architecture comprises two layers: an input layer and a neural layer. Differently from weighted neural networks, that store knowledge in their synapses, in VG-RAM WNNs each neuron of a neural layer has a set of weightless synapses $\mathbf{S} = \{s_1, \ldots, s_p\}$. The data read from the corresponding input layer through the synapses are transformed in a vector of bits $I = \{i_1, \ldots, i_p\}$ (one bit per synapse). Each bit of this vector is computed using a synapse mapping function that transforms non-binary values from the input layer in binary values.

The VG-RAM WNN neurons store knowledge in private local memories that work as look-up tables and keep sets $\mathbf{L} = \{L_1, \ldots, L_j, \ldots, L_m\}$ of pairs $L_j = (I_j, t_j)$, where I_j is a binary input vector and t_j is its corresponding output label. The binary input vectors are extracted from the input layers via the set $\mathbf{S}$ of synapses of each neuron, while the output labels t are the learned neurons' output-values for each binary input vector I.

VG-RAM WNN supervised training and test work as follows. During training, an input pattern and its expected output pattern are set in the input layer and the output of the VG-RAM neural layer respectively. Firstly, each neuron

extracts a binary input vector I from the input layer, via its set of synapses **S** (one bit per synapse). Secondly, the expected output label t is set in the output of the corresponding neuron in the neural layer. Finally, this input-output pair $L = (I, t)$ is subsequently stored into the neuron's look-up table. During test, an input pattern is set in the input layer and each neuron extracts a binary input vector I from the given input pattern via its set of synapses **S**. The neurons subsequently use I to search and find, in their look-up tables, the input I_j, belonging to the learned input-output pairs $L_j = (I_j, t_j)$ that is the closest to the I vector extracted from the input layer. Finally, the output of the neuron receives the label value t_j of this L_j input-output pair. In case of more than one pair L_j with an input I_j at the same minimum distance of the extracted input I, the output value t_j is randomly chosen among them.

The search for the nearest input in each neuron's memory is performed sequentially and the distance is measured using the Hamming distance. It is important to note that the Hamming distance between two binary patterns can be efficiently computed at machine code level in current 64-bit CPUs and GPUs of personal computers using two instructions: one to identify the bits that differ in 64-bit segments of the two binary patterns, i.e. a bit-wise exclusive-or instruction; and another to count these bits, i.e. a population count instruction.

IV. VG-RAM Image-Based Global Localization

VibGL memorizes images and associated poses (i.e. "a map") and can later recover the poses from similar images of the same environment (localization in the map). For that, during training, VibGL learns a map of the environment, represented internally by the contents of the memories of its neurons, in a way that mimics the human ability of learning visual maps.

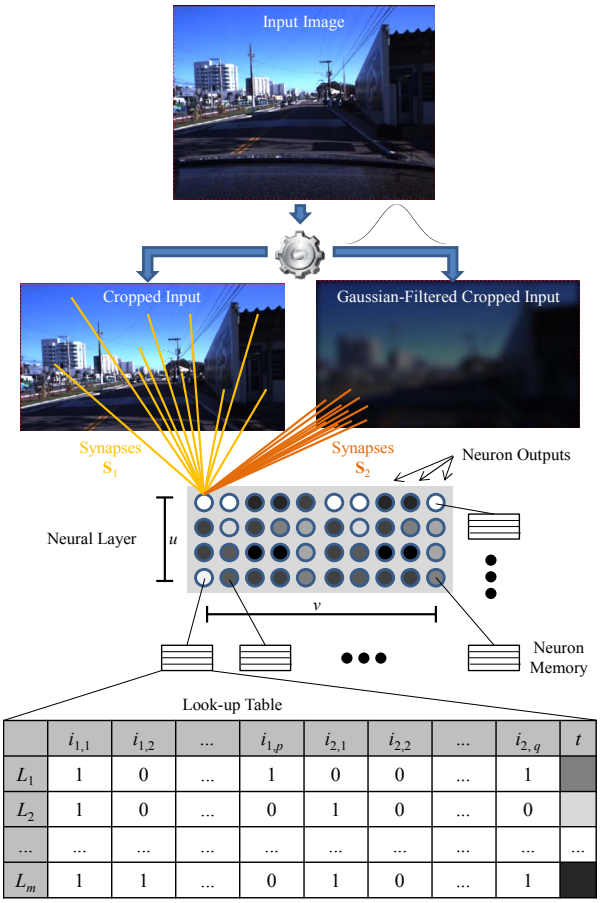
A. Overview

This section describes the core of the VibGL system. VibGL is designed to learn a sequence of images and associated poses that describes a trajectory according to a cockpit perspective. Doing so, it allows for global localization considering all learned locations in the trajectories, given a single image. Therefore, it can be used as a GPS replacement within the learned trajectories, but, instead of using satellites for localization, it uses visual cues from images.

B. VibGL Architecture

The VibGL employs a VG-RAM WNN architecture that captures holistic and feature-based aspects of input images by using two different synaptic interconnection patterns. Fig. 2 shows an overview of the VibGL system.

VibGL uses a single Neural Layer with $u \times v$ VG-RAM neurons with m-size memory (see Fig. 2). This Neural Layer is connected to two input layers, (i) Cropped Input and (ii) Gaussian-Filtered Cropped Input, according to two different synaptic interconnection patterns, $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively. $\mathbf{S}_1 = \{s_{1,1}, \ldots, s_{1,p}\}$ and $\mathbf{S}_2 = \{s_{2,1}, \ldots, s_{2,q}\}$ are subsets of $\mathbf{S} = \{s_{1,1}, \ldots, s_{1,p}, s_{2,1}, \ldots, s_{2,q}\}$, i.e., $\mathbf{S} = \mathbf{S}_1 \cup \mathbf{S}_2$, where **S** is the set of synapses of each neuron of the VibGL's Neural Layer.



	$i_{1,1}$	$i_{1,2}$	...	$i_{1,p}$	$i_{2,1}$	$i_{2,2}$	...	$i_{2,q}$	t
L_1	1	0	...	1	0	0	...	1	
L_2	1	0	...	0	1	0	...	0	
...	...	...	...	...	...	...	...	...	...
L_m	1	1	...	0	1	0	...	1	

Fig. 2. Illustration of the VibGL system. VibGL employs a $u \times v$ VG-RAM WNN Neural Layer of neurons with m-size memory. Each neuron is connected to two processed versions of the Input Image (Cropped Input and Gaussian-Filtered Cropped Input) through two sets of synapses, $\mathbf{S}_1$ and $\mathbf{S}_2$ (exemplified for one neuron in yellow and orange respectively). $\mathbf{S}_1 = \{s_{1,1}, \ldots, s_{1,p}\}$ and $\mathbf{S}_2 = \{s_{2,1}, \ldots, s_{2,q}\}$ are subsets of $\mathbf{S} = \{s_{1,1}, \ldots, s_{1,p}, s_{2,1}, \ldots, s_{2,q}\}$, i.e., $\mathbf{S} = \mathbf{S}_1 \cup \mathbf{S}_2$, where **S** is the set of synapses of each neuron. This set of synapses samples the neuron's inputs as a vector of bits $I = \{i_{1,1}, \ldots, i_{1,p}, i_{2,1}, \ldots, i_{2,q}\}$. The Neural Layer shows an example of activation pattern based on the binary input vectors I and labels t of the learned pairs $L = (I, t)$. Each neuron responds with the label t_j associated with the input I_j that is the closest to the binary input vector I extracted from the Cropped Input and the Gaussian-Filtered Cropped Input. The labels t are indexes to geo-tagged images.

Each neuron samples the Cropped Input and the Gaussian-Filtered Cropped Input in two different ways: holistically, with $\mathbf{S}_1$; and feature-based, with $\mathbf{S}_2$. The set of synapses $\mathbf{S}_1$ samples the Cropped Input holistically because it is defined according to a uniform random interconnection pattern that covers the whole Cropped Input; while $\mathbf{S}_2$ samples the Gaussian-Filtered Cropped Input featured-based because it is defined according to a Normal distribution centered in the position of the neuron mapped to this input (see Fig. 2 and [16] for details about the feature-based synaptic interconnection pattern).

The synaptic mapping function that maps non-binary image pixels to binary values is a *minchinton cell type-0* [17] that works as follows. Each pixel is treated as an integer $y = b \times 256 \times 256 + g \times 256 + r$, where b, g, and r are the blue, green and red color channels. The non-binary pixel value y read by each synapse is subtracted from the non-binary pixel value y read by the subsequent synapse in the set of synapses of each neuron, $\mathbf{S} = \{s_{1,1}, \ldots, s_{1,p}, s_{2,1}, \ldots, s_{2,q}\}$. The value read by the last synapse, $s_{2,q}$, is subtracted from the value read by the first, $s_{1,1}$. If a negative value is obtained, the bit corresponding to that synapse is set to one; otherwise, it is set to zero.

The two input layers, Cropped Input and Gaussian-Filtered Cropped Input, are processed versions of the Input Image. While the Cropped Input is simply a region of interest defined in the input image, the Gaussian-Filtered Cropped Input is the result of a Gaussian filter applied to this region of interest (see Fig. 2 for an example).

The region of interest was defined in order to remove irrelevant pixel information from the input image. In our case, the bottom of the image is cropped out to eliminate static part of the car roof visible in the field of view of a mounted-on camera. The Gaussian filter, in the other hand, is used as a low-pass image filter. Since a feature-based synaptic interconnection pattern is used to sample this input layer, high-frequency attenuation is necessary to remove spurious high-frequency information irrelevant for localization.

During VibGL training, the system learns images and associated camera-poses. Let $\mathbf{T} = \{T_1, \ldots, T_j, \ldots, T_{|\mathbf{T}|}\}$ be a set of pairs $T_j = (\text{image}_j, \text{pose}_j)$ presented to VibGL. The image_j of each pair T_j is set as the VibGL's Input Image and the corresponding index j is copied to the output of each neuron of VibGL's Neuron Layer. Then, all neurons are trained to output j when sampling from image_j via Cropped Input and Gaussian-Filtered Cropped Input images. After training, the index j learned by the neurons can be used for recovering pose_j or image_j.

During test, given a query image, VibGL infers a pose based on the previously acquired knowledge. A query image is set as VibGL's Input Image and all neurons compute their outputs, which are indexes (32-bit integers). Each neuron infers an index based on the input binary vectors extracted by their synapses. The number of votes for each index is counted and the network outputs the index with the largest count.

V. Experimental Setup

This section presents the experimental setup used to evaluate the VibGL system. It starts describing the autonomous vehicle platform used to acquire the datasets, follows presenting the datasets themselves, and finishes describing the methodology used in the experiments.

A. Autonomous Vehicle Platform

We collected the data to evaluate the performance of the VibGL system using the *Intelligent and Autonomous Robotic Automobile* – IARA (Fig. 3). IARA is an experimental robotic platform based on a Ford Escape Hybrid that is currently being developed at *Laboratório de Computação de Alto Desempenho* (High-Performance Computing Laboratory – www.lcad.inf.ufes.br) of *Universidade Federal do Espírito Santo* - UFES (Federal University of Espírito Santo – Brazil).

Fig. 3. Intelligent and Autonomous Robotic Automobile (IARA) with the mounted-on Point Grey Bumblebee XB3 camera (marked in green) used in experiments. Learn more about IARA at www.lcad.inf.ufes.br.

Our robotic platform has several high-end sensors, including: two Point Grey Bumblebee XB3 stereo cameras and two Bumblebee 2 stereo cameras; one Light Detection and Ranging (LIDAR) Velodyne HDL 32-E; and one GPS-aided Attitude and Heading Reference System (AHRS/GPS) Xsens MTiG (see Fig. 3). To process the data coming from the sensors, the platform has four Dell Precision R5500 (2 Intel Xeon 2.13 GHZ, 12 GB RAM, 2 SSDs of 120GB on RAID0 and GPU cards Tesla C2050). We implemented many software modules for IARA that currently allows for its autonomous operation (such as modules for mapping, localization, obstacle avoidance, navigation, etc.; see video of IARA autonomous operation at http://youtu.be/4rFCjrFdR7o and videos about other IARA's software modules at http://www.youtube.com/user/lcadufes).

To build the datasets used in this work, we used IARA's frontal Bumblebee XB3 left camera to capture images (640x480 pixels), and IARA's fused-odometry module to capture associated poses (6 degrees of freedom – 6D). IARA's fused-odometry module employs a Monte Carlo particle filter [1] to fuse sensor data coming from the AHRS/GPS and from the visual-odometry module. The visual-odometry module employs the LibViso2 library [18] to compute IARA's basic odometry data (6D pose) from images collected from IARA's frontal Bumblebee XB3 stereo camera. The poses computed by our fused-odometry module have precision of about 2.5m (GPS Circular Error Probability equals to 2.5m).

B. Datasets

For the experiments, we have used two laps data acquired in different dates. Basically, for each lap, IARA was driven with an average speed of 30 km/h around UFES campus. A full lap around the university campus has an extension of about 3.57 km. During the laps, image and pose data were synchronously acquired.

The first lap data was recorded in October 3rd 2012 (UFES-2012) and comprises a sequence of 15,306 image-pose pairs, while the second lap data was recorded in September 16th 2013 (UFES-2013) and comprises a sequence of 9,017 image-pose pairs. The difference in days between the recording of the first and the second lap data is almost one year. Such time difference resulted in a challenging testing scenario since it captured substantial changes in the campus environment. Such changes includes differences in traffic conditions, number of pedestrians, and alternative routes taken due to construction work obstructions on the road. Also, there were substantial building infrastructure modifications alongside the roads in between dataset recording.

The correspondences between the two lap data were established using the Euclidean distance between image-pose pairs to define a ground truth. In average, the distances between UFES-2012 and UFES-2013 corresponding image-pose pairs is 1.56m ($\sigma = \pm 1.14$). To evaluate the effect of learning different numbers of images (the more images VibGL learns, the more labels it has to differentiate) the lap data were sampled at four different intervals: 1m, 5m, 10m, and 15m. After sampling the UFES-2012, four datasets were created: 1-meter spacing dataset with a total of 2,485 image-pose pairs, a 5-meter dataset with a total of 639 image-pose pairs, a 10-meter dataset with 331 image-pose pairs, and a 15-meter dataset with 224 image-pose pairs. The same was done with the UFES-2013 resulting in four datasets with a total of 2,689, 670, 345 and 233 image-pose pairs. All datasets mentioned above are available at: http://www.lcad.inf.ufes.br/log.

C. Experimental Methodology

In order to validate our system, we have run a set of localization experiments. The two lap data (UFES-2012 and UFES-2013) were divided into training and test datasets. In all experiments, the training and test datasets were from different dates. One of them was used to teach the system about the trajectory (training the system), and the other was used to test the accuracy of the system by estimating poses along the learned trajectory. We measured the Euclidean distance from the estimated pose to the associated ground truth pose in the testing set.

To increase the testing scenario, we have crossed the two lap data by running four tests with the 1-meter, 5-meter, 10-meter, and 15-meter spacing datasets sampled from UFES-2012 as training and the 1-meter spacing dataset sampled from UFES-2013 as testing, and four tests with the 1-meter, 5-meter, 10-meter, and 15-meter spacing datasets sampled from UFES-2013 as training and the 1-meter spacing dataset sampled from UFES-2012 as testing.

VI. Results and Discussions

In this section, we show and discuss the outcomes of our experiments. The results are presented in four parts: classification accuracy; positioning error; true and false positives; and finally, qualitative results.

A. Classification Accuracy

This subsection shows the relationship between the amount of frames learned by the VibGL system and its classification accuracy. We measured the system classification accuracy in terms of how close the VibGL's estimated image-pose pair, T_e, is to the correct image-pose pair, T_c, for a given query image, T_q. The image-pose pairs T_e and T_c belong to the training dataset, while the image T_q belongs to the test dataset. The correct pair, T_c, was defined through the established ground truth correspondence with T_q. Ideally, T_e is equal to T_c if VibGL is correct in its estimate, since both image-pose pairs T_c and T_e belongs to the training dataset.

Fig. 4 shows the classification accuracy results obtained using UFES-2012 dataset for training and UFES-2013 dataset for testing. The vertical axis represents the percentage of image-pose pairs T_e that were within an established maximum number-of-frames distance from the image-pose pair T_c. The number-of-frames distance is equal to the amount of image-pose pairs that one has to go forward or backwards in the training dataset to find T_c from the corresponding T_e. The horizontal axis represents the number-of-frames distance. Finally, the curves of the graph of Fig. 4 show how the accuracy increases with the allowed maximum number-of-frames distance for the different training datasets.

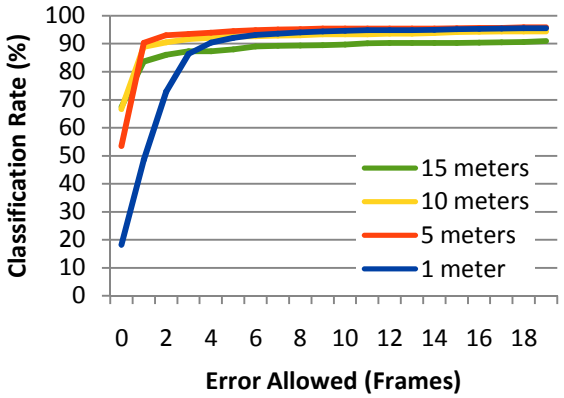


Fig. 4. Classification accuracy for different maximum number-of-frames allowed using UFES-2012 dataset for training and UFES-2013 dataset for test.

As the graph of Fig. 4 shows, VibGL's classification accuracy increases with the maximum allowed number-of-frames and reaches a plateau at about 5 frames for all training datasets. However, for the UFES-2012 1-meter spacing training dataset, the VibGL classification uncertainty is large in the beginning of the curve due to the similarity between images in the near-by image-pose pairs. If one does not accept any system error (number-of-frames allowed equal zero), the accuracy is only about 18% when the system is trained with the 1-meter spacing dataset. But, if one accepts as correct an image-pose pair up to 5 frames ahead or behind the correct image-pose pair, T_c, the accuracy increases to about 92%. On the other hand, when using a dataset with a larger spacing between image-pose pairs for training, the system accuracy increases more sharply. For example, when the system is trained with the 5-meter spacing dataset, with an

allowed number-of-frames equal to 1, the classification rate is about 90%.

Although the VibGL might show better accuracy when trained with large-spaced datasets, the positioning error of the system increases. This happens because one frame of error for the 1-meter training dataset represents a much smaller error in meters than one frame of error with large-spaced training dataset (e.g., 10m).

Fig. 5 shows the classification accuracy results obtained using UFES-2013 dataset for training and UFES-2012 dataset for testing. As the graph of Fig. 5 shows, swapping training and test datasets does not change the VibGL performance behavior.

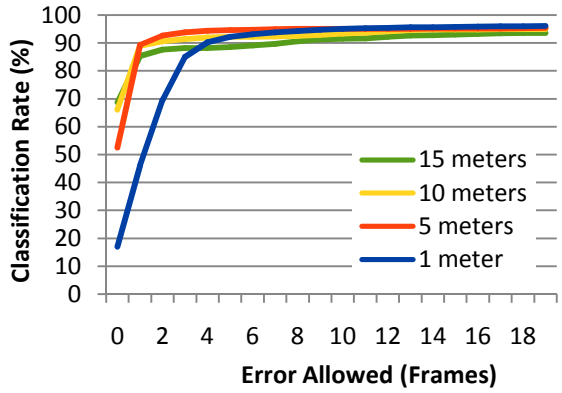


Fig. 5. Classification accuracy for different maximum number-of-frames allowed using UFES-2013 dataset for training and UFES-2012 dataset for test.

B. *Positioning Error*

We performed experiments to evaluate the relationship between the spacing between image-pose pairs learned by VibGL and the positioning error of its estimated poses.

The results of these experiments are shown as box-plots having median, inter-quartile range and whiskers of the error distribution for the 1-meter, 5-meter, 10-meter and 15-meter training datasets. Box-plots are shown only for the setup UFES-2012 as training and UFES-2013 as test because the experiments present the same behavior in both directions.

The horizontal axis of Fig. 6 shows training datasets spacing intervals, while the vertical axis shows the distance of the estimated image-pose pair T_e to the given image-pose pair T_q. As the graph of Fig. 6 shows, the positioning error increases as the spacing between training image-pose pairs increases, but not linearly. The median error is larger than 1m for the 1-meter spacing training dataset, but smaller than the spacing of the other datasets. The performance of VibGL with the 1-meter dataset can be explained by the fact that the datasets themselves are imprecise due to the poor accuracy of our data collection framework. As mentioned before, the distances between UFES-2012 and UFES-2013 corresponding image-pose pairs is 1.56m ($\sigma = \pm 1.14$).

The horizontal axis of Fig. 7 shows training datasets spacing intervals, while the vertical axis shows the distance of the estimated image-pose pair T_e to the correct image-pose pair T_c.

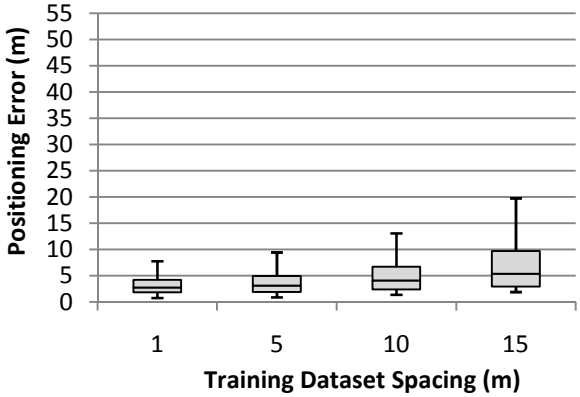


Fig. 6. Positioning Error Distribution between T_e and T_q using the UFES-2012 dataset for training and the UFES-2013 dataset for testing.

As the graph of Fig. 7 shows, the larger the spacing in between the training frames, the smaller the positioning error is. This can be seen by analyzing the distribution of error for the 5, 10 and 15-meters plots, where the median error is around 0 and the majority of errors are within the spacing used for training (i.e., within one frame).

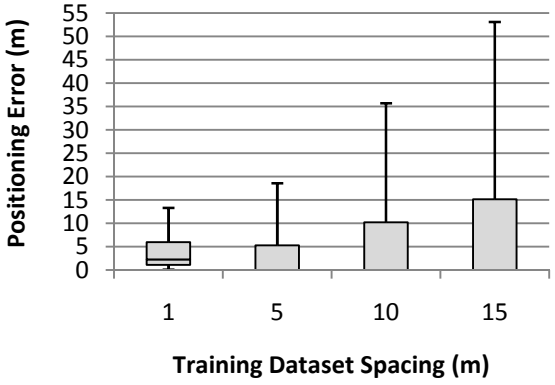


Fig. 7. Positioning Error Distribution between T_e and T_c using the UFES-2012 dataset for training and the UFES-2013 dataset for testing.

While Fig. 6 shows a continuous error value between the estimated pose at T_e and the given pose T_q, Fig. 7 shows a discretized error value between the estimated pose at T_e and the correct image-pose T_c.

C. *True and False Positives*

After analyzing classification accuracy and positioning error, in this section, we examine the true and false positives of VibGL for the 1-meter training dataset and 1-meter test dataset, with a 10m error tolerance. This allows us to identify patterns of positioning errors in the datasets.

Fig. 8 shows the results for the experiments with the UFES-2012 dataset used for training and the UFES-2013 dataset for test. In this figure, blue dots represent estimated positions that were inside the tolerance of 10m from the correct image-pose pair, i.e., true positives. Red circles represent the estimated image-pose pairs that were outside the tolerance of 10m, i.e., false positives. Red circles are

connected to their corresponding correct image-pose pairs (green crosses) through a line.

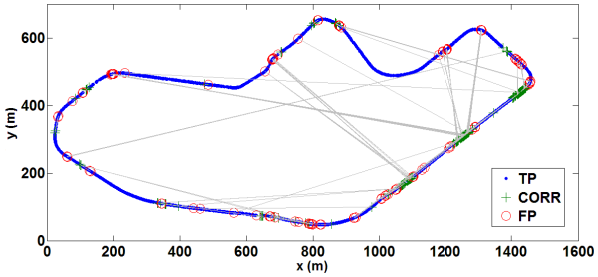


Fig. 8. Ground truth plot using UFES-2012 dataset in the training procedure. Blue dots represent true positives (TP); lines connect false positives (FP) represented by red circles and their respective ground-truth correspondences represented by green crosses (CORR).

Fig. 9 shows the results for the experiments with the UFES-2013 dataset used for training and the UFES-2012 dataset for test.

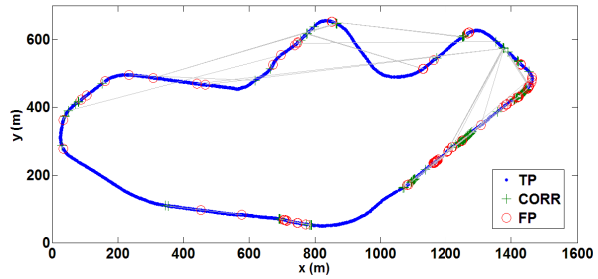


Fig. 9. Ground truth plot using UFES-2013 dataset in the training procedure. Blue dots represent true positives (TP); lines connect false positives (FP) represented by red circles and their respective ground-truth correspondences represented by green crosses (CORR).

From the plots in Fig. 8 and Fig. 9, it is possible to see three regions where most errors occurred in a sequence (the three big green zones on the right side of the Fig. 8 and Fig. 9). Further analysis of these regions showed they correspond to three points where the car went off-route in one of the datasets. Since there were obstructions on the road at the period, the car was forced to leave the main street and, therefore, generated shifted images (see Fig. 10). Such differences in the training and test images leaded the VibGL system to misclassifications and, consequently, to failure in localizing IARA correctly on those regions.

D. Qualitative Results

To visualize the qualitative results for VibGL's estimated positions, we extracted two samples of matched frames along the UFES campus: the first one having five true positive samples (Fig. 11), and the second one having three false positive samples (Fig. 12).

Fig. 11 shows examples of true positive frames using the UFES-2013 dataset as training dataset. As it can be seen, the frames were matched despite changes in sunlight position, shadows, road infrastructure (first, second and fourth rows), car movements (third row) and loss of leaves on the trees (fourth row).

Fig. 10. IARA's original trajectory (left image, UFES-2012 dataset). Shifted image generated when the car went off-route (right image, UFES-2013 dataset).

Fig. 12 shows examples of false positive frames using the UFES-2013 as training dataset. The system fails at places with certain similarity, e.g., the sky-shape in the first and fifth rows looks the same. Moreover, in the first row, the two frames were captured on a curve within the same road paving. In the second row, we can see the above-mentioned off-route problem, when the car needs to get out of the route.

An online demo video shows the VibGL's performance on a complete lap around the university campus (see video at http://youtu.be/czxSMb0irw4). In the video, we used the 1 meter-spacing UFES-2013 dataset for training and the 5 meter-spacing UFES-2012 dataset for testing.

VII. Conclusion and Future Work

In this work, we present a new Image-Based Global Localization approach based on VG-RAM WNN, named VibGL. VibGL is designed for mapping and localization, and efficiently solves the global localization problem [1]. For that, it first learns a map of an environment and, later, localizes the robot in the same environment using camera images only. In the mapping phase, VibGL firstly receives images of the environment as well as the associated poses of the camera that captured the images. Subsequently, it learns neural associations between images and poses, which describe the environment as a map. In the localization phase, VibGL receives images of the environment and uses previously acquired knowledge, the "map", to output the positions where the system believes these images were captured.

We have tested VibGL with a set of mapping and localization experiments using real-world datasets. Our results have shown that VibGL is capable of learning large maps (several kilometers in length) and, later, localizing a robot in these maps with median pose precision of about 3m. Considering a tolerance of 10m, the system is able to correctly localize the robot 95% of the time (test cases).

As future work, we plan to compare VibGL with existing other image retrieval methods. In addition, we intend to extend VibGL to a full neural SLAM system.

Fig. 11. True positive qualitative results using the UFES-2013 dataset as training dataset. The left column represents the trained frames and right column the VibGL's output frames.

Fig. 12. False positive qualitative results using the UFES-2013 dataset as training dataset. The left column represents the trained frames and right column the VibGL's output frames.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, Mass.: MIT Press, 2005.

[2] T. Ludermir, A. Carvalho, A. Braga, and M. Souto, 'Weightless neural models: A review of current and past works', *Neural Computing Surveys*, vol. 2, pp. 41–61, 1999.

[3] D. G. Lowe, 'Object recognition from local scale-invariant features', in *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999*, 1999, vol. 2, pp. 1150–1157 vol.2.

[4] H. Bay, T. Tuytelaars, and L. V. Gool, 'SURF: Speeded Up Robust Features', in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Springer Berlin Heidelberg, 2006, pp. 404–417.

[5] S. Se, D. Lowe, and J. Little, 'Vision-based Mobile Robot Localization And Mapping using Scale-Invariant Features', in *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA*, 2001, pp. 2051–2058.

[6] J. Wolf, W. Burgard, and H. Burkhardt, 'Using an Image Retrieval System for Vision-Based Mobile Robot Localization', in *Image and Video Retrieval*, M. S. Lew, N. Sebe, and J. P. Eakins, Eds. Springer Berlin Heidelberg, 2002, pp. 108–119.

[7] D. Nister, O. Naroditsky, and J. Bergen, 'Visual odometry', in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004*, 2004, vol. 1, pp. I–652–I–659 Vol.1.

[8] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, 'MonoSLAM: Real-Time Single Camera SLAM', *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[9] H. Lategahn, A. Geiger, and B. Kitt, 'Visual SLAM for autonomous ground vehicles', in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1732–1737.

[10] A. Geiger, J. Ziegler, and C. Stiller, 'StereoScan: Dense 3d reconstruction in real-time', in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 963–968.

[11] M. J. Milford and G. F. Wyeth, 'Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System', *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1038–1053, 2008.

[12] A. J. Glover, W. P. Maddern, M. J. Milford, and G. F. Wyeth, 'FAB-MAP + RatSLAM: Appearance-based SLAM for multiple times of day', in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 3507–3512.

[13] M. Cummins and P. Newman, 'FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance', *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, Jan. 2008.

[14] J. Sivic and A. Zisserman, 'Video Google: a text retrieval approach to object matching in videos', in *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, 2003, pp. 1470–1477 vol.2.

[15] M. J. Milford and G. F. Wyeth, 'SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights', in *2012 IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 1643–1649.

[16] A. F. D. Souza, C. Badue, F. Pedroni, E. Oliveira, S. S. Dias, H. Oliveira, and S. F. de Souza, 'Face Recognition with VG-RAM Weightless Neural Networks', in *Artificial Neural Networks - ICANN 2008*, V. Kůrková, R. Neruda, and J. Koutník, Eds. Springer Berlin Heidelberg, 2008, pp. 951–960.

[17] R. J. Mitchell, J. M. Bishop, S. K. Box, and J. F. Hawker, 'Comparison of some methods for processing Grey Level data in weightless networks', in *RAM-based neural networks*, J. Austin, Ed. Singapore: World Scientific Publishing Co Pte Ltd, 1998, pp. 66–71.

[18] H. Lategahn, A. Geiger, B. Kitt, and C. Stiller, 'Motion-without-structure: Real-time multipose optimization for accurate visual odometry', in *2012 IEEE Intelligent Vehicles Symposium (IV)*, 2012, pp. 649–654.