2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Improving Repeatability of Experiments by Automatic Evaluation of SLAM Algorithms

Francesco Amigoni[1], Valerio Castelli[1], and Matteo Luperto[2]

Abstract— **The development of good experimental methodologies for robotics takes often inspiration from general principles of experimental practice. *Repeatability* prescribes that experiments should involve several trials in order to guarantee that results are not achieved by chance, but are systematic, and statistically significant trends can be identified. In this paper, we propose an approach to improve the repeatability of experiments performed in robotics. In particular, we focus on the domain of SLAM (Simultaneous Localization And Mapping) and we introduce a system that exploits simulations to generate a large number of test data on which SLAM algorithms are automatically evaluated in order to obtain consistent results, according to the principle of repeatability.**

I. INTRODUCTION

Development of good experimental methodologies for robotics is a topic that has attracted increasing interest [1]. The discussion has evolved from early methodological proposals [2], [3] to a tangible impact on publications, with special issues [4] and special kinds of articles (reproducible articles or R-articles) [5]. Several practical solutions have been advanced to support good experimental methodologies, ranging from the use of datasets [6], [7], to the development of platforms for benchmarking [8], [9], and to the definition of robotic competitions [10], [11].

Among the several aspects that are involved in good experimental methodologies, the principles of reproducibility and repeatability are central. They refer to two similar but not fully overlapped characteristics of experimental practice [12]. *Reproducibility* is the possibility to verify, in an independent way, the results of an experiment. This means that experimenters, other than those claiming for the validity of the results, should be able to achieve the same results when starting from the same initial conditions, using the same type of instruments and parameters, and adopting the same experimental techniques. *Repeatability*, instead, refers to the fact that a single result is not sufficient to ensure the success of an experiment. A successful experiment must involve a number of trials, possibly performed at different times and in different places, in order to guarantee that results have not been achieved by chance, but are systematic, and that statistically significant trends can be identified.

In this paper we propose an approach that enhances the repeatability of experiments performed in robotics. In particular, we focus on the domain of SLAM (Simultaneous Localization And Mapping) [13] and we first show that the performance of SLAM algorithms presents some variability when the algorithms are applied to data collected with different runs in the same environment. This aspect is often disregarded and SLAM algorithms are usually evaluated on data acquired with single runs in different environments. We then introduce a system that exploits simulations to generate a large number of test data on which SLAM algorithms are automatically evaluated in order to obtain consistent results, according to the principle of repeatability. Our system is finally validated showing that a SLAM algorithm applied to the test data we generate shows a performance very similar to that obtained when the algorithm is applied to data coming from real robots.

This paper is organized as follows. The next section motivates the contribution we provide. Section III illustrates our proposed method and the system we developed, which is experimentally validated in Section IV. Section V concludes the paper.

II. MOTIVATION

In this section, we motivate the approach we present in this paper to improve the repeatability of experiments in robotics. We focus on a specific, yet significative and widely studied, domain, that of SLAM. Broadly speaking, in a SLAM problem, a robot should localize itself within a map of the environment that it is building at the same time, on the basis of data coming from its sensors, typically laser range scanners and encoders.

We consider a well-known SLAM algorithm based on particle filters, called GMapping [14]. In brief, it maintains a predefined number of hypotheses (particles) about the map of the environment and the pose of the robot, which are continuously updated according to the information provided by new observations (laser range scans and odometry readings). The selection of the particles that should be maintained or eliminated at each update step is based on a maximum likelihood probabilistic approach, so that particles that are less likely to represent the current knowledge of the robot (including the observations) tend to be replaced.

We also consider a commonly used metric to evaluate the performance of SLAM algorithms [15], that provides a measure of the translational and rotational components of the localization error, which is calculated by comparing the trajectory of the robot as reconstructed by a SLAM algorithm and the ground truth trajectory. The details of the metric are explained in Section III-B.

[1]F. Amigoni and V. Castelli are with the Artificial Intelligence and Robotics Laboratory, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy francesco.amigoni@polimi.it, valerio1.castelli@mail.polimi.it

[2]M. Luperto is with the Applied Intelligent Systems Laboratory, Università degli Studi di Milano, Via Festa del Perdono 7, 20122 Milano, Italy matteo.luperto@unimi.it

978-1-5386-8093-3/18/$31.00 ©2018 IEEE

TABLE I: Translational localization errors (in m) of GMapping over 12 runs in 5 indoor environments.

	building 1	building 2	building 3	building 4	building 5
average	0.191	0.195	0.225	0.262	0.195
std dev	0.027	0.040	0.032	0.034	0.052

Fig. 1 shows the translational localization errors of GMapping when it is applied to data acquired in 5 indoor environments. Each curve is relative to an environment and represents the translational localization errors obtained in 12 runs (with the same experimental setting: same robot configuration, same initial pose, same parameters for the algorithm, ...), ordered from the largest to the smallest. The mean and standard deviation of the errors are reported in Table I. It is clear that there is great variability (which is common to all the environments). As a consequence, the performance measured according to a single run is hardly informative about the average performance of GMapping in the environment. This is clearly in contrast with the principle of repeatability. The sources of this variability are the noise affecting the perceptions and the movements of the robot and the inherent stochasticity of GMapping. While it is possible that this variability could be less significant for some environments and some SLAM algorithms, the problem seems rather general and little investigated.

Indeed, evaluating SLAM algorithms on the basis of the performance measured in a single run (or in few runs) seems to be common also if, as we have shown, it does not provide a reliable assessment. For instance, in [14], the performance of GMapping is evaluated with a single run in three environments, while in [16] the authors consider a single simulation run per environment and three real-world runs in a test arena. In this work, we propose an original approach that automates the execution of data collection runs in simulated environments in order to inexpensively generate several test data on which SLAM algorithms are automatically evaluated.

III. OUR APPROACH

In this section, we present our proposal for improving the repeatability of experiments in robotics. We develop a system that generates a large number of test data on which

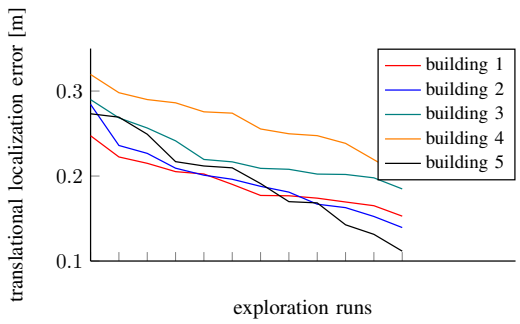


Fig. 1: Performance of GMapping in 12 runs in 5 indoor environments.

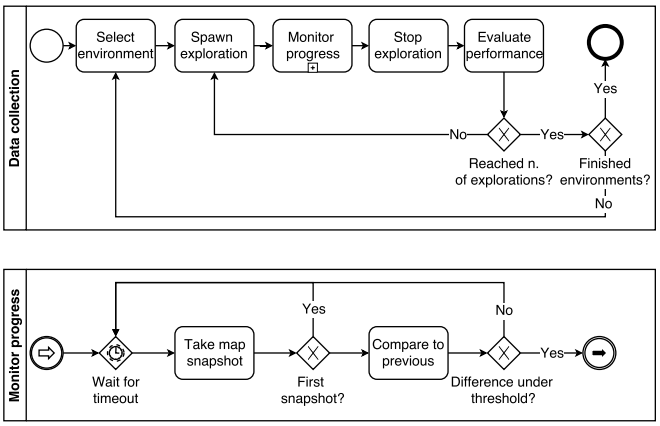


Fig. 2: The workflow of our system.

statistically significant conclusions can be drawn. We focus on SLAM algorithms, and on GMapping in particular, but many of the following considerations and results could be extended to other SLAM algorithms and to other domains, as well. An overview of the workflow of our system is shown in Fig. 2. Details are provided in the following of this section.

In general, running experiments for testing SLAM algorithms involves three main elements: the algorithm, the test data, and the metric. The *algorithm* we consider in this paper is GMapping [14], since it is very well-know and could be considered as a good representative of SLAM algorithms. The *test data* usually come from datasets (like Radish [6] and Rawseeds [7]) or from direct acquisition using real robots. However, these two collection methods provide test data relative to single runs (or to a small number of runs) and can be hardly applied to provide data corresponding to several runs. For this reason, we employ automated simulations to generate test data (Section III-A). Finally, in other to evaluate the quality of the results returned by GMapping when it is applied to the test data, we need a *metric* and an automatic way to apply it (Section III-B).

The code of our system is available[1]. Datasets are currently available upon request, amounting to more than 300 GB.

A. Automated Simulations

We first define a set of environments $\mathcal{E}$ where we collect the test data on which the performance of GMapping is evaluated. We consider a set of 100 indoor environments representing real world buildings belonging to different building types (like schools, offices, and university campuses) and having different sizes and shapes (to increase diversity of data). Each environment $E \in \mathcal{E}$ is represented as a `png` image, with a resolution of 0.05×0.05 meters per grid cell (pixel). Environments in $\mathcal{E}$ are taken from 3 different sources. $\mathcal{E}$ includes 11 of the 20 floor plans of [17], some of which are from the Radish repository (we select the version of environments without furniture). Their size ranges from $100\,\mathrm{m}^2$ to $1000\,\mathrm{m}^2$. $\mathcal{E}$ also includes 25 floor plans of

[1] `https://github.com/AIRLab-POLIMI/predictivebenchmarking`

buildings of the MIT university campus, from [18]. Their size ranges from $1000\,\mathrm{m}^2$ to $30\,000\,\mathrm{m}^2$. Finally, we complete $\mathcal{E}$ with our dataset of 64 floor plans [19], 26 offices and 38 schools, whose size ranges from $100\,\mathrm{m}^2$ to $10\,000\,\mathrm{m}^2$.

For each environment $E \in \mathcal{E}$ we collect, using simulations, a set of test data $\mathcal{D}_E$ to be fed to GMapping. Note that, in addition to reducing the costs of data collection, simulations also easily provide the ground truth of the trajectories followed by the robot that, as discussed in Section III-B, is required by the metric we employ.

Simulations are performed in Stage, using the ROS GMapping[2] and Navigation packages[3]. Mapping is performed using 40 particles and processing a new scan whenever the robot travels $1\,\mathrm{m}$, rotates $0.25\,\mathrm{rad}$, or $5\,\mathrm{s}$ have passed since the last update of the map. We employ a virtual robot equipped with a two-dimensional laser range scanner with a field of view of $270°$, an angular resolution of $0.5°$, and a range of $30\,\mathrm{m}$. In our simulations, we assume that the virtual robot has a translational odometry error of up to $0.01\,\mathrm{m/m}$ and a rotational odometry error of up to $2\,°/\mathrm{rad}$, which provide a reasonable approximation of the odometry accuracy of real wheeled robots. The actual amount of error is randomly chosen by the simulator (at the start of each run) with uniform probability in the range $[-0.005, +0.005]\mathrm{m/m}$ and $[-1, +1]°/\mathrm{rad}$, respectively. Although Stage, as any simulator, does not fully capture all the aspects of the real world, its use allows us to generate data easily. Moreover, as shown in Section IV, these data are quite similar (in order to evaluate GMapping performance) to those obtained with real robots.

Given an environment E and a starting pose for the robot (close to the center of the environment, the same for all runs), a run R_E explores E using the frontier-based exploration approach of [20] according to which the robot moves to the closest frontier, where a frontier is a region on the boundary between known and unknown space, collecting laser range scans and odometry readings at each time step (every $100\,\mathrm{ms}$ in our case). These (timestamped) data are both fed to the ROS GMapping node and stored in a ROS bag file[4]. A run R_E ends when two consecutive snapshots of the grid map produced by GMapping (taken every $120\,\mathrm{s}$ for small environments and every $600\,\mathrm{s}$ for large environments) are similar enough, according to the mean square error metric that evaluates the difference between images. Empirically, this automated criterion for termination amounts to fully map the environment E in almost all runs. Note that the exploration process finds frontiers on the map of the environment incrementally built by GMapping. As a consequence of this, and of the localization errors simulated in Stage, exploration runs follow different paths in the environment. These paths include several loop closures, although our online exploration approach does not actively seek to find them. (The use of offline approaches that optimize loop closures is an interesting future direction of work.)

[2] http://wiki.ros.org/gmapping
[3] http://wiki.ros.org/navigation
[4] http://wiki.ros.org/Bags

At the end of each run R_E we have thus the set $D(R_E)$ of data (laser range scans and odometry readings) collected along the path followed to cover environment E. These data are fed to GMapping that, at the end of the exploration, produces the grid map $M(R_E)$ and the estimated poses of the robot $x_{1:T_{R_E}}$ from time step 1 to T_{R_E} (the time step at which the exploration run R_E ended). The process is automatically iterated until a number of runs $|\mathcal{R}_E|$ (where $\mathcal{R}_E$ is the set of runs performed in E) are performed for each environment E, as we discuss in Section III-C. Eventually, for each environment $E \in \mathcal{E}$, we have the set $\mathcal{D}_E = \{D(R_E) \text{ for all } R_E \in \mathcal{R}_E\}$ of test data and the corresponding results produced by GMapping, namely the set of grid maps $\{M(R_E) \text{ for all } R_E \in \mathcal{R}_E\}$ and the set of estimated poses $\{x_{1:T_{R_E}} \text{ for all } R_E \in \mathcal{R}_E\}$. (We note that the test data $\mathcal{D}_E$ could be used to evaluate other SLAM algorithms without the need to re-run simulations.) We point out that the test data $\mathcal{D}_E$ are relative to the particular configuration of the virtual robot (and of its sensors) that we have considered. For example, changing the field of view or the range of the laser range scanner leads to generate a set of data that could be different. However, generating a new set of test data $\mathcal{D}_E$ for an environment E is relatively cheap (for example, in a large environment, like that of Fig. 6b, an exploration run requires, on average, 43.8 minutes). Similarly, the data $\{M(R_E)\}$ and $\{x_{1:T_{R_E}}\}$, representing the results of GMapping, depend on the configuration of the algorithm (and, of course, on test data $\mathcal{D}_E$).

B. Automated Metric

The research community has developed several metrics to assess the performance of SLAM algorithms. Some of them involve a comparison with the ground truth [16], [21], [22], while some others are based on evaluating the usefulness of the results (e.g., maps or localization) produced by SLAM algorithms [7], [23]. In our work, we employ a variant of the *localization error* metric proposed by [15], which measures the performance of SLAM algorithms according to their ability to accurately estimate the actual trajectory followed by the robot. The idea is to measure the deformation energy that is required to superimpose the estimated trajectory onto the ground truth trajectory: the smaller the energy, the higher the accuracy of the reconstruction. We choose this metric because it does not rely on any particular map representation format nor on any specific sensor.

In [15], the metric is defined as follows.

Definition III.1. *Let $x_{1:T}$ be the poses of the robot estimated by a SLAM algorithm from time step 1 to T during an exploration run of environment E; in our case, $x_t \in SE(2)$, with $SE(2)$ being the special Euclidean group of order 2.*
Let $x^*_{1:T}$ be the associated ground truth poses of the robot during mapping.
Let $\delta_{i,j} = x_j \ominus x_i$ be the relative transformation that moves the pose x_i onto x_j and let $\delta^*_{i,j} = x^*_j \ominus x^*_i$.
Finally, let δ be a set of N pairs of relative transformations

over all the exploration, $\delta = \{\langle \delta_{i,j}, \delta^*_{i,j} \rangle\}$.
The localization error *performance metric is defined as:*

$$\begin{aligned}\varepsilon(\delta) &= \frac{1}{N}\sum_{i,j}(\delta_{i,j} \ominus \delta^*_{i,j})^2 = \\ &= \frac{1}{N}\sum_{i,j}[trans(\delta_{i,j} \ominus \delta^*_{i,j})^2 + rot(\delta_{i,j} \ominus \delta^*_{i,j})^2] = \\ &= \varepsilon_t(\delta) + \varepsilon_r(\delta),\end{aligned}$$

where the sums are over the elements of δ, $\ominus$ *is the inverse of the standard motion composition operator, and* $trans(\cdot)$ *and* $rot(\cdot)$ *are used to separate the translational and rotational components of the error.*

In order to apply this metric, we need to address two issues. First, the metric as defined above is intrinsically devoted to evaluate a single run R_E in the environment E. Second, the metric requires to define the set δ of pairs of relative transformations.

Addressing the first issue amounts to moving from the evaluation of the performance measured on a single run in an environment E (namely, on test data $D(R_E)$) to the evaluation of performance measured on all the runs in E (namely, on test data $\mathcal{D}_E$). In principle, one would like to evaluate the *expected localization error* of a generic exploration run in an environment E.

Definition III.2. *Let* p_{δ_E} *be the probability of observing the set* δ_E *set of relative transformations during an exploration run in an environment* E.
The mean translational localization error *in* E, $\mathbb{E}[\varepsilon_t(E)]$, *is the expected value of the translational component of the localization error over all the possible exploration runs on E:*

$$\mathbb{E}[\varepsilon_t(E)] = \sum_{\delta_E} \varepsilon_t(\delta_E) * p_{\delta_E}.$$

The standard deviation of the translational localization error *in* E, $\sigma[\varepsilon_t(E)]$, *is:*

$$\sigma[\varepsilon_t(E)] = \sqrt{\mathbb{E}[\varepsilon_t(E)^2] - \mathbb{E}[\varepsilon_t(E)]^2}.$$

Similarly, the mean rotational localization error *in* E, $\mathbb{E}[\varepsilon_r(E)]$, *is the expected value of the rotational component of the localization error over all the possible exploration runs on E:*

$$\mathbb{E}[\varepsilon_r(E)] = \sum_{\delta_E} \varepsilon_r(\delta_E) * p_{\delta_E}.$$

The standard deviation of the rotational localization error *of environment* E, $\sigma[\varepsilon_r(E)]$, *is:*

$$\sigma[\varepsilon_r(E)] = \sqrt{\mathbb{E}[\varepsilon_r(E)^2] - \mathbb{E}[\varepsilon_r(E)]^2}.$$

We approximate the above quantities with their sampled versions, since the weak law of large numbers guarantees their convergence to the theoretical definitions as the number of exploration runs $|\mathcal{R}_E|$ in an environment E increases [24].

Definition III.3. *Let* $\mathcal{E}$ *be a set of environments,* $E \in \mathcal{E}$ *one of such environments, and* $\mathcal{R}_E$ *the set of exploration runs performed on* E.
The sample mean and sample standard deviation of the translational localization error *in* E *are:*

$$\overline{\varepsilon_t(E)} = \frac{\sum_{R_E \in \mathcal{R}_E} \varepsilon_t(\delta_{R_E})}{|\mathcal{R}_E|}$$

$$s(\varepsilon_t(E)) = \sqrt{\frac{\sum_{R_E \in \mathcal{R}_E} [\varepsilon_t(\delta_{R_E}) - \overline{\varepsilon_t(E)}]^2}{|\mathcal{R}_E|}}.$$

The sample mean and sample standard deviation of the rotational localization error *in* E *are defined as:*

$$\overline{\varepsilon_r(E)} = \frac{\sum_{R_E \in \mathcal{R}_E} \varepsilon_r(\delta_{R_E})}{|\mathcal{R}_E|}$$

$$s(\varepsilon_r(E)) = \sqrt{\frac{\sum_{R_E \in \mathcal{R}_E} [\varepsilon_r(\delta_{R_E}) - \overline{\varepsilon_r(E)}]^2}{|\mathcal{R}_E|}}.$$

We now turn to the second issue discussed above, namely the determination of the set $\delta = \{\langle \delta_{i,j}, \delta^*_{i,j} \rangle\}$ of pairs of relative transformations. In each pair, the relative transformation $\delta_{i,j}$ between two poses as estimated by the SLAM algorithm is associated to the relative transformation $\delta^*_{i,j}$ between the ground truth poses. In [15], human expertise is exploited to determine the N pairs of relative transformations in δ. After a run, a human operator analyzes the pairs of laser range scans acquired by the robot to determine which ones refer to the same part of the environment and manually aligns them. (This is done in order to cope with the difficulty of collecting ground truth trajectories in real world scenarios.) The amount of displacement required for the alignment is stored as the ground truth of the relative transformation $\delta^*_{i,j}$ between the poses x_i and x_j from which the laser range scans have been acquired. The human operator can match laser range scans at semantically relevant places (e.g., loop closures), providing ground truth for global consistency. Clearly, this method does not efficiently scale as the numbers of laser range scans, runs, and environments increase.

Since we are using simulations, we can assume to have the ground truth trajectories followed by the robot. Hence, we propose a new way to determine δ_{R_E} of Definition III.3 that is independent of human intervention. Although, in principle, δ_{R_E} could contain all the possible pairs of relative transformations (i.e., for all the i and j in $1 : T_{R_E}$), this solution is unpractical, because the size of δ_{R_E} would be quadratic in the number of poses on the robot's trajectory.

We propose to build δ_{R_E} by randomly sampling a set of relative transformations, whose size trade-offs between sampling quality and computational complexity. The procedure is based on the central limit theorem to approximate the sampling distribution with a normal distribution [25]. The quality of the sampling is relative to the accuracy of the estimation of the localization error. More precisely, we set the confidence level and the margin of error of the estimation and

we determine the number of relative transformations sampled for estimating the localization error as:

$$N = \frac{z_{\alpha/2} * s^2}{d^2}, \tag{1}$$

where s^2 is the usual unbiased estimator of the population variance, d is the margin of error, α is the complement of the desired confidence level and $z_{\alpha/2}$ is its associated z-score. To validate our approach, we empirically verify the distribution normality assumption on a representative set of environments. Fig. 3 shows the sample distribution of the translational localization error $\varepsilon_t()$ in two of these environments. The distributions are obtained by repeatedly extracting 200 different samples of relative transformations imposing a 99 % confidence level and a margin of error of ± 0.02 m. It is evident that the shape of the distributions is approximately normal. The above process is sound if we assume the relative transformations to be independent and identically distributed random variables. In principle, this may not be the case for all pairs of relative transformations, for example, relative transformations that involve pairs of poses that are close to each other are similar and not independent. However, the number of possible relative transformations is so large that, given any two random relative transformations, the likelihood that they are dependent can be assumed negligible for all practical purposes.

To show that sampling relative transformations leads to a metric that actually captures the quality of SLAM results, Fig. 4 shows a good and a bad map of the same environment, with the bad map being visibly broken with a room that is significantly misaligned. This visual difference is correctly reflected by the metric, as the translational and rotational localization errors of the good map are of $0.54\,\mathrm{m}$ and $0.02\,\mathrm{rad}$, respectively, while those of the bad map are $2.42\,\mathrm{m}$ and $0.29\,\mathrm{rad}$, respectively.

In summary, given data relative to all runs $\mathcal{R}_E$ performed in the environment E, we calculate mean and standard deviation of $\varepsilon_t(E)$ and $\varepsilon_r(E)$, namely of the two components of the localization error, according to Definition III.3.

According to what discussed at the end of Section III-A, the values of the mean and standard deviation of $\varepsilon_t()$ and $\varepsilon_r()$ depend on the virtual robot configuration. For example, for the environment of Fig. 4, the value of $\overline{\varepsilon_t()}$ is $0.68\,\mathrm{m}$ if the range of the laser range scanner is $30\,\mathrm{m}$ and $0.91\,\mathrm{m}$ if the range is $15\,\mathrm{m}$. (The intuitive explanation is that, with a reduced range, the robot travels a longer distance and the error increases.)

C. Size of the Sample of Runs $|\mathcal{R}_E|$

Given an environment $E \in \mathcal{E}$, the values of $\overline{\varepsilon_t(E)}$, $s(\varepsilon_t(E))$, $\overline{\varepsilon_r(E)}$, and $s(\varepsilon_r(E))$ can provide good approximations of the true mean and standard deviation of the two components of the localization error, namely can satisfy the repeatability principle, if they are calculated over an enough large number of runs $|\mathcal{R}_E|$.

We cannot in general assume that different runs of GMapping in the same environment are independent from each other. However, Chebyshev's weak law of large numbers guarantees the convergence of the sample mean to the true mean under the assumption that the covariances tend to be zero on average [24]. Then, we assume the distribution of the sample mean to be approximately normal and we exploit the same formulation of Equation (1) to obtain $|\mathcal{R}_E|$.

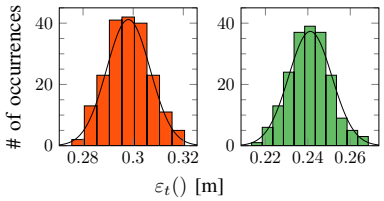


Fig. 3: Distribution of the translational localization error $\varepsilon_t()$ in two environments.

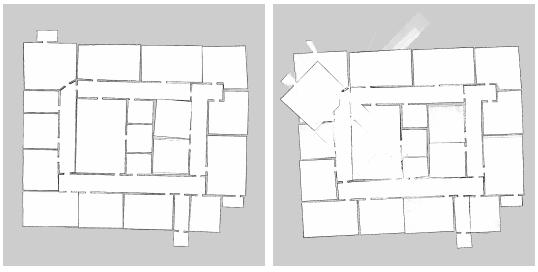
Given E, the estimation of the sample size $|\mathcal{R}_E|$ is performed as follows. The process starts with an initial estimate of the variance of the localization error, obtained from a small sample of 10 runs. We use this value to compute an initial estimate of the number of required runs. We then perform that number of runs and compute a new estimate of the variance and its associated sample size, iteratively repeating the process until the newly estimated sample size is not larger than the number of already performed runs. In our case, we end up with different values of $|\mathcal{R}_E|$ for different environments E, with an average of $|\mathcal{R}_E| = 36$ (and a total of about $3,600$ simulated exploration runs in Stage).

Note that the sample size N_t required for an accurate estimate of the translational localization error may differ from the sample size N_r required to accurately estimate the rotational localization error; in this case, we consider the maximum of N_t and N_r.

IV. EXPERIMENTAL VALIDATION

In this section, we show the effectiveness of the proposed approach and we validate it.

Fig. 5 shows the translational localization errors of GMapping in all the runs performed in the 100 environments of $\mathcal{E}$ (the rotational localization errors are similar). The variability of the performance in any given environment and the presence of some outliers are evident, reinforcing the need



(a) good map (b) bad map

Fig. 4: A good and a bad map of the same environment.

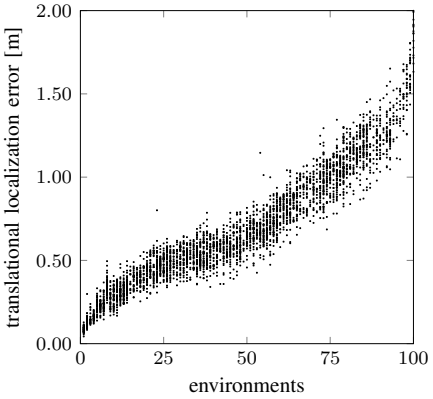
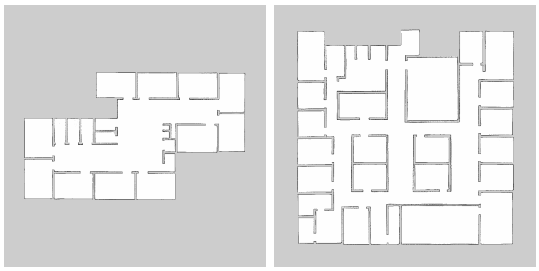


Fig. 5: The translational localization errors for all the runs in our environments.



(a) E_a, $1400\,\mathrm{m}^2$ (b) E_b, $2400\,\mathrm{m}^2$

Fig. 6: Two environments for which the performance of GMapping measured with a single run is not informative.

to adopt an experimental methodology, like that embedded in our system, that satisfies the repeatability principle.

Fig. 6 shows an example of the utility of the proposed system. In environment E_a of Fig. 6a, GMapping has a mean translational localization error of $\overline{\varepsilon_t(E_a)} = 0.31\,\mathrm{m}$, but the translational localization error of one of the runs is $0.43\,\mathrm{m}$. In environment E_b of Fig. 6b, GMapping has a mean translational localization error of $\overline{\varepsilon_t(E_b)} = 0.52\,\mathrm{m}$, but the translational localization error of one of the runs is $0.39\,\mathrm{m}$. Therefore, looking only at the two single runs, one could conclude that GMapping performs better in E_b than in E_a, but, on average (and with a statistically sound number of runs, see Section III-C), the opposite is true.

We now validate our approach based on simulation by comparing results obtained with it against those obtained with real robots. We use some datasets collected with real robots (both publicly available and acquired in our lab) and we implement simulated versions of the same settings. Note that, in order to use the metric of Section III-B, the datasets we use for validation must include the ground truth trajectories of the robot.

We first consider the dataset of [26], which is composed of four runs executed by a real robot in an L-shaped industrial hall of size $10 \times 12\ \mathrm{m}^2$. Ground truth data of the robot trajectories are obtained with a motion capture system and are reported in the dataset. We extract two runs from the dataset.

TABLE II: Translational ([m]) and rotational ([rad]) components of the localization error for the dataset of [26].

	empty hall		furniture	
	$\varepsilon_t()$	$\varepsilon_r()$	$\varepsilon_t()$	$\varepsilon_r()$
dataset (real robot)	0.189	0.058	0.267	0.070
simulation	0.223	0.030	0.245	0.045

In the first one, the environment is an empty hall without furniture, while in the second one the same environment is furnished. In both cases, the robot is configured in the same way but, unfortunately, the characteristics of the laser range scanner are not explicitly reported. So, we analyzed the raw scans from the dataset, inferring that the field of view should be $80°$, the angular resolution $1°$, the range $30\,\mathrm{m}$, and the frequency $10\,\mathrm{Hz}$. Since they are not explicitly reported, we assume the translational and rotational odometry errors of the robot to be upper bounded by $0.01\,\mathrm{m/m}$ and $2\,°/\mathrm{rad}$, respectively. We replicate the two settings in our simulation framework of Section III-A, using the configuration of the robot employed in [26]. Both the data from the dataset (collected by the real robot) and the data collected in our simulations are fed to GMapping and the reconstructed trajectories are compared to the ground truth trajectory using the metric of Section III-B. The components of the localization error are reported in Table II (the simulation data are averaged over 10 runs). Comparing the values in table, we can claim that, for this setting, results obtained in our simulations are a rather good approximation of those obtained with a real robot.

We consider another dataset collected in the Artificial Intelligence and Robotics Laboratory (AIRLab) at the Politecnico di Milano. The environment has a size of 9×9 m^2 (see Fig. 7 right) and is covered by an OptiTrack motion capture system to record the ground truth trajectory of the robot. We use a three-wheeled differential drive robot, called Robocom, equipped with a SICK LMS100 laser range scanner, with a field of view of $270°$, an angular resolution of $0.25°$, a range of $20\,\mathrm{m}$, and a frequency of $50\,\mathrm{Hz}$ (Fig. 7 left). The translational and rotational errors affecting the odometry of Robocom are manually estimated to be not larger than $0.01\,\mathrm{m/m}$ and $4\,°/\mathrm{rad}$, respectively. In the real world, we perform 10 runs, each involving the autonomous exploration of the area (Fig. 7 center) following the same frontier-based approach of Section III-A. We also perform 10 simulations, as before, reproducing the environment in our simulation framework and using the Robocom configuration (Fig. 7 right). The comparison of the components of the localization error made by GMapping when applied to data coming from Robocom and from the simulations is shown in Table III. The difference between the $\varepsilon_t()$ of the simulations and that of the real robot is rather small (a difference of $2.5\,\mathrm{cm}$ on the mean and less than $1\,\mathrm{cm}$ on the standard deviation). The difference relative to $\varepsilon_r()$ is more significant and can be explained with the poor rotation mechanism of Robocom, which sometimes introduces errors larger than the $4\,°/\mathrm{rad}$ threshold we estimated. Overall, the results of Tables II and III show that the performance of GMapping obtained

TABLE III: Translational and rotational components of the localization error for the AIRLab experiment.

	translational error [m]		rotational error [rad]	
	$\varepsilon_t()$	$s(\varepsilon_t())$	$\varepsilon_r()$	$s(\varepsilon_r())$
Robocom	0.086	0.026	0.066	0.010
simulation	0.101	0.019	0.022	0.004

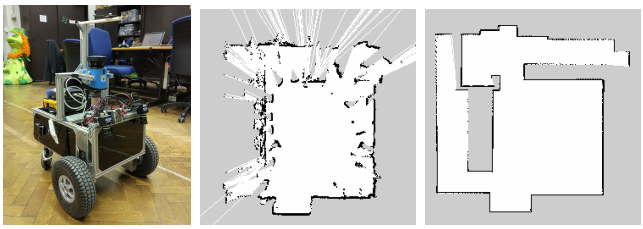


Fig. 7: Robocom (left). The map built by Robocom in the AIRLab (center). The map build in a simulation run (right).

with data collected with our simulator is comparable to that obtained with data collected with real robots. This outcome suggests the validity of our simulation-based approach to automatically evaluate SLAM algorithms.

V. CONCLUSIONS

In this paper we have presented an approach to address the limited repeatability of experiments performed to evaluate SLAM algorithms. The proposed system exploits simulations to generate a large amount of test data with relatively small effort and automates the evaluation of SLAM algorithms. The validation has shown that GMapping performs similarly on test data collected by real robots and on test data generated with our approach. Note that the availability of several test data, relative to different runs in the same environment and to different environments, promotes also the reproducibility of experimental results.

While we have considered a specific algorithm (GMapping) and a specific simulator (Stage), most modules of our system could be generalized with small adjustments to other SLAM algorithms and other simulators, and, in principle, to other domains. Preliminary results obtained with Karto SLAM[5] seem to confirm the findings of this paper. A drawback of the proposed approach, as it is currently structured, is that the test data it generates depend on the configuration of the virtual robot and of its sensors. Making the approach more platform-independent is one of the challenges for future work.

REFERENCES

[1] F. Amigoni and V. Schiaffonati, "Models and experiments in robotics," in *Springer Handbook of Model-Based Science*, L. Magnani and T. Bertolotti, Eds. Springer, 2017, pp. 799–815.

[2] F. Bonsignorio, J. Hallam, and A. del Pobil, "Gem guidelines," http://www.heronrobots.com/EuronGEMSig/downloads/GemSigGuidelinesBeta.pdf, last visited July 2018.

[3] F. Amigoni, M. Reggiani, and V. Schiaffonati, "An insightful comparison between experiments in mobile robotics and in science," *Auton Robot*, vol. 27, no. 4, pp. 313–325, 2009.

[4] F. Bonsignorio and A. del Pobil, "Toward replicable and measurable robotics research [from the guest editors]," *IEEE RAM*, vol. 22, no. 3, pp. 32–35, 2015.

[5] F. Bonsignorio, "A new kind of article for reproducible research in intelligent robotics [from the field]," *IEEE RAM*, vol. 24, no. 3, pp. 178–182, 2017.

[6] A. Howard and N. Roy, "The robotics data set repository (radish)," 2003. [Online]. Available: http://radish.sourceforge.net/

[7] G. Fontana, M. Matteucci, and D. Sorrenti, "Rawseeds: Building a benchmarking toolkit for autonomous robotics," in *Methods and Experimental Techniques in Computer Engineering*, F. Amigoni and V. Schiaffonati, Eds. Springer, 2014, pp. 55–68.

[8] J. Weisz, Y. Huang, F. Lier, S. Sethumadhavan, and P. Allen, "RoboBench: Towards sustainable robotics system benchmarking," in *Proc. ICRA*, 2016, pp. 3383–3389.

[9] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The Robotarium: A remotely accessible swarm robotics research testbed," in *Proc. ICRA*, 2017, pp. 1699–1706.

[10] F. Amigoni, E. Bastianelli, J. Berghofer, A. Bonarini, G. Fontana, N. Hochgeschwender, L. Iocchi, G. Kraetzschmar, P. Lima, M. Matteucci, P. Miraldo, D. Nardi, and V. Schiaffonati, "Competitions for benchmarking," *IEEE RAM*, vol. 22, no. 3, pp. 53–61, 2015.

[11] L. Iocchi, D. Holz, J. Ruiz-del-Solar, K. Sugiura, and T. van der Zant, "RoboCup@Home: Analysis and results of evolving competitions for domestic and service robots," *Artif Intell*, vol. 229, pp. 258–281, 2015.

[12] I. Hacking, *Representing and Intervening*. Cambridge University Press, 1983.

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

[14] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE T Robot*, vol. 23, no. 1, pp. 34–46, 2007.

[15] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner, "On measuring the accuracy of SLAM algorithms," *Auton Robot*, vol. 27, no. 4, pp. 387–407, 2009.

[16] J. Santos, D. Portugal, and R. Rocha, "An evaluation of 2D SLAM techniques available in Robot Operating System," in *Proc. SSRR*, 2013, pp. 1–6.

[17] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele, "Room segmentation: Survey, implementation, and analysis," in *Proc. ICRA*, 2016, pp. 1019–1026.

[18] E. Whiting, J. Battat, and S. Teller, "Generating a topological model of multi-building environments from floorplans," in *Proc. CAADFutures*, 2007, pp. 115–128.

[19] M. Luperto, A. Quattrini Li, and F. Amigoni, "A system for building semantic maps of indoor environments exploiting the concept of building typology," in *Proc. RoboCup*, 2013, pp. 504–515.

[20] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. CIRA*, 1997, pp. 146–151.

[21] B. Balaguer, S. Carpin, and S. Balakirsky, "Towards quantitative comparisons of robot algorithms: Experiences with SLAM in simulation and real world systems," in *IROS Workshop on Performance Evaluation and Benchmarking for Intelligent Robots and Systems*, 2007.

[22] S. Schwertfeger and A. Birk, "Map evaluation using matched topology graphs," *Auton Robot*, vol. 40, no. 5, pp. 761–787, 2016.

[23] T. Colleens, J. Colleens, and D. Ryan, "Occupancy grid mapping: An empirical evaluation," in *Proc. MED*, 2007, pp. 1–6.

[24] S. Karlin and H. Taylor, *A First Course in Stochastic Processes*. Academic Press, 1975.

[25] P. Billingsley, *Probability and Measure*. Wiley, 1995.

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IROS*, 2012, pp. 573–580.

[5] http://wiki.ros.org/slam_karto