Experimental Analysis of Sample-Based Maps for Long-Term SLAM

Peter Biber[1] and **Tom Duckett**[2]

[1]Dept. of Computer Science, WSI-GRIS
University of Tübingen
Tübingen, Germany
Email: biber@wsi-gris.uni-tuebingen.de

[2]Dept. of Computing and Informatics
University of Lincoln
Lincoln LN6 7TS, UK
Email: tduckett@lincoln.ac.uk

March 17, 2008

Abstract

This paper presents a system for long-term SLAM (simultaneous localization and mapping) by mobile service robots and its experimental evaluation in a real dynamic environment. To deal with the stability-plasticity dilemma (the trade-off between adaptation to new patterns and preservation of old patterns), the environment is represented at multiple timescales simultaneously (5 in our experiments). A sample-based representation is proposed, where older memories fade at different rates depending on the timescale, and robust statistics are used to interpret the samples. The dynamics of this representation are analysed in a five week experiment, measuring the relative influence of short- and long-term memories over time, and further demonstrating the robustness of the approach.

1 Introduction

Future service robots will be required to run autonomously in dynamic environments for really long periods of time. These robots will be required to live together with people and adapt to the changes that people make to the world.

The first excursion of a mobile service robot in a new environment will probably be guided by a human or other intelligent system, for example, to explain the meaning of different places, as a human employee would be introduced to

a new workplace. A basic problem for the robot here is *static SLAM*: simultaneous localization and mapping given one set of data. Most previous work on robotic map learning addresses this problem. However, this phase will only be a short moment in the lifetime of a service robot that is expected to operate for years. For a robot to survive in a dynamic and ever-changing world, lifelong learning is essential.

This paper addresses the problem of *long-term SLAM*: simultaneous localization and mapping in dynamic environments over possibly unlimited periods of time. A major challenge for long-term SLAM is that environments can change at different rates, and the changes can be gradual or abrupt. Changes may not be permanent: an object may have been moved, a package may have been left for a while, etc. It is therefore desirable for the robot to remember the old state too in case the change is only temporary. At the same time, there may be slow but inexorable changes such as plants growing or coloured paint fading. This challenge is related to a more general and well-known problem confronting every lifelong learning system, namely the *stability-plasticity dilemma* described by Grossberg [9]. Lifelong learning demands both adaption to new patterns and preservation of old patterns at the same time.

This work makes several contributions:

- We propose sample-based maps for mobile robots, using the sensor data themselves as primitives of the representation ("using the data as its own best model"). A novel representation based on *dynamic sample sets* enables adaptation of the map in a changing world. We show that this representation has a well-defined semantics and a probabilistic interpretation using robust statistics.

- To deal with the stability-plasticity dilemma, we propose simultaneous representation of the world at *multiple timescales*, using multiple maps with a spectrum of learning rates. This allows the robot to maintain multiple hypotheses in time about the state of the environment, for example, representing the world before, during and after a temporary object is left in a particular place.

- Based on these concepts, a complete system for long-term SLAM is presented. During localization the robot compares its current sensor data to all timescales in the map and chooses the timescale that best fits the data. In turn, the results of localization are used directly to adapt the map. Consequently localization is more robust and the map does not go out of date.

Further to our previously published results [5], this paper analyses the dynamics of the map representation in a long-term experiment using sensor data recorded by a mobile robot in a busy indoor environment over a period of five weeks. We measure the relative influence of short-term and longer-term memories over time, and further demonstrate the long-term stability and performance of the approach.

1.1 Related work

Traditional mapping and localization algorithms model the world as being static and try at most to detect and filter out moving objects such as people. Previous approaches to robotic mapping of dynamic environments can be grouped into three categories.

First, some approaches attempt to discriminate "dynamic" from "static" elements of the environment [6, 15, 16, 10]. For example, the RHINO tour guide robot [6] used an entropy filter to separate sensor readings corresponding to known objects such as walls from readings caused by dynamic obstacles such as people. A fixed pre-installed map was used for localization, while an occupancy grid was built on the fly to model dynamic objects and combined with the static map for path planning. However, in general, this approach cannot handle long-term changes to the environment. By contrast our work uses a spectrum of learning rates to handle changes occurring at different timescales.

Second, several authors have investigated aging of the map using some form of recency weighted averaging. Zimmer [18] presented a system that dynamically learns and updates the topology of a map during runtime and showed the ability of his model to adapt to changes. Yamauchi and Beer [17] developed a so-called adaptive place network, where a confidence value for each link in the network was updated in a recency-based manner based on successful or non-unsuccessful attempts to traverse the link, and links with low confidence were deleted. Andrade-Cetto and Senafeliu [1] developed an EKF-based map learning system that is able to forget landmarks that have disappeared, where an existence state associated with each landmark measures how often it has been seen.

Third, some authors propose richer world models, e.g. with semantics based on explicit identification of objects. Anguelov et al. [2] divide the environment into a static part and objects that can move such as chairs. However, these efforts are decoupled from map building and localization, and it is assumed that these parts work independently and perfectly. The aim of Anguelov's work is more on obtaining one higher level map, and not to adapt the map continuously as in this work. Stachniss and Burgard [13] proposed an approach to modeling dynamic environments by representing typical configurations or possible states in a local area, e.g. corresponding to open and closed doors. In our experiments, however, we found that such repeated configurations were rare compared to unexpected changes: most of the changes in our environment resulted in new configurations that had not been seen before.

In contrast to all these systems our dynamic map addresses the stability-plasticity dilemma. We do not consider autonomous navigation or topological changes, rather our focus is on seeing self-localization and map learning as a never-ending cycle.

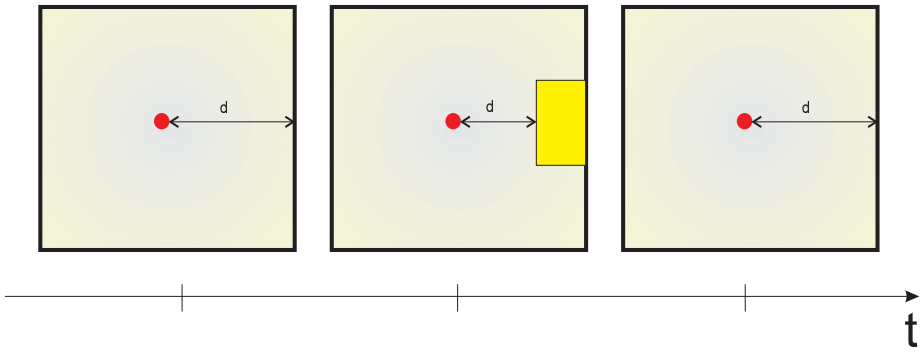

Figure 1: A simple example environment. After some time a cupboard is put in front of one wall, and some time later it is removed. The example map consists only of the distance d.

2 Motivation for Sample-Based Maps

To motivate the map representations proposed in this paper a simple toy environment is considered as shown in Fig. 1. The example environment is an empty room and the map to be learned is the distance from an arbitrary point to the nearest wall or object in a given direction. If the environment were static then learning would be simple: all deviations from the true value would be due to Gaussian measurement noise, and therefore the map would be perfectly represented by the sample mean and sample variance. For stationary distributions these statistics are sufficient (Duda et al. [8], chapter 3.6), i.e. they store the information content of all measurements, and it is not necessary to keep past measurements in memory.

Assume now that the environment is dynamic. After some time somebody puts a cupboard in front of the wall. The above method is not well-suited to this challenge. The time needed for the sample mean to approach the new true value would be proportional to the time spent measuring the wall in the old position. Such behaviour is not desired.

Requirement 1: The time taken by the map to adapt to a change should not depend on how much time has passed in absolute terms. Also, the initial state should have no special status or rank.

Recency weighted averaging is a solution to this problem from the field of reinforcement learning that is especially suited to non-stationary tracking problems (Sutton and Barto [14], chapter 2.6). Here the estimate for the distance d is updated after a new measurement according to:

$$d_{\text{new}} = (1 - \alpha) * d_{\text{old}} + \alpha * d_{measured} \tag{1}$$

Effectively this methods calculates a weighted sample average. The weight w_t

of a sample is thus dependent on its age t as

$$w_t = \alpha * e^{-\lambda t} \qquad \text{with } \lambda = -\ln(1-\alpha), \tag{2}$$

assuming that measurements are made at regular intervals. So the influence of old measurements decays according to a well-known law that governs many growing and decaying processes in nature, where a parameter λ determines the speed of adaption or learning rate.

Imagine that the cupboard is removed after a few days. If λ is small (e.g. 1 year^{-1}) only a small change in the map will occur. On the other hand for large λs (e.g. 1 s^{-1}) the map will converge to the new value immediately. In the first case the cupboard can be seen as an outlier and the quality of the distance estimate to the wall should not get worse. This reminds us of the notorious problem that statistics derived from least square formulations are not robust to outliers.

Requirement 2: Map learning should be robust to outliers.

The proposed solution of a sample-based representation can also be justified by another observation that lies in the nature of the problem: actual changes in the environment appear in the first instance as outliers. At the moment a measurement is made it is not possible to determine whether it is an outlier or not. Only after more time has passed and more measurements have been made can outliers be identified. Therefore any method that tries to identify outliers immediately after measurement is, in principle, not well-suited to map learning in dynamic environments.

Requirement 3: The map representation should be able to track multiple hypotheses until it can be determined whether a change has really happened or only outliers were measured.

Thus any method that represents the environment by a unimodal distribution cannot be considered an adequate solution.

Outliers aside, there is a further problem with recency weighted averaging: during learning in the example scenario the distance estimate would change gradually from "wall" to "cupboard", but none of these in-between estimates would correspond to any physical reality. In contrast to many other dynamic processes, the environmental changes considered here are not necessarily continuous but more often discrete and rapid. In the example it would be more natural for the estimate to represent either the distance to the wall or the distance to the cupboard.

Requirement 4: The map should yield only values that have actually been measured and should not create interpolated values that do not correspond to any past or current reality.

The following section describes the map representation developed in order to meet these requirements.

3 Sample-based representation

The classical answer to the above requirements is to apply robust statistics [11]. The median of a set of samples fulfils requests 2 (it ignores up to 50 percent of outliers) and 4. But there are no sufficient statistics to describe the median. Sufficient statistics preserve the information content of the whole data set, so the data itself can be discarded. But to calculate robust statistics we always need a full set of data. It is of course impossible to save all data ever recorded. Our solution here is a sample-based representation: we maintain a set of samples drawn from the data that approximates all recent data, thus also fulfilling requirements 1 and 3.

3.1 Dynamic sample sets

The state of the map is represented at discrete time steps t_i. The basic representation of the dynamic map is a set $\mathcal{S}(t_i)$ of n samples. A sample is just a measurement that has been recorded before time step t_i. A sample set is a function of time and is therefore the central concept that makes the map dynamic. Its temporal evolution is calculated using an update rule and measurements.

Let $\mathcal{M}$ be the set of measurements that have been made between two subsequent time steps t_i and t_{i+1}. $\mathcal{S}(t_{i+1})$ is then calculated by an update rule dependent on an update rate $0 \leq u \leq 1$ as follows:

- Remove $u * n$ randomly chosen samples from $\mathcal{S}(t_i)$.
- Replace them by $u * n$ randomly chosen samples from $\mathcal{M}$ to get $\mathcal{S}(t_{i+1})$.

This algorithm is applied if the sample set is already full (that is, it contains the maximum number n of samples). Initially a sample set is empty and until it is full an update just consists of adding $u * n$ randomly chosen samples.

3.2 Semantics of a sample set

Let s be a sample that has been added at time step t_i. The probability that a randomly chosen sample from the sample set $\mathcal{S}(t_i)$ has just been added like s is $u * n/n = u$. In each subsequent time step the probability of s remaining in the sample set is given by $1 - u$. So at time step $t_j > t_i$ the probability for s to be still in $\mathcal{S}(t_j)$ is given by $(1 - u)^{(t_j - t_i)}$. Thus we can make the following statement about the distribution of ages in a sample set:

At any time the probability for a sample to have been added to the sample set t time steps before is given by:

$$p(t) = u * e^{\ln(1-u)*t} \tag{3}$$

Thus the age of the samples is distributed just like the weights in recency weighted averaging. The distribution is dependent on a timescale parameter $\lambda = -\ln(1 - u)$, and has the following well-known properties: the mean life

time τ of a sample is given by: $\tau = \lambda^{-1}$, and the half-life is given by $t_{1/2} = \frac{\ln 2}{\lambda}$. This means that one half of the sample set is expected to be younger than the half-life and the other half older.

3.3 Probabilistic interpretation of a sample set

To actually use the map at a time step t a normal distribution $\mathcal{N}(\rho, \sigma^2)$ can be robustly estimated from the samples using the median and the median of absolute deviations:

$$\hat{\rho} = \text{median}(\mathcal{S}(t)) \quad (4)$$

$$\hat{\sigma} = 1.48 \text{ median}(|x - \hat{\rho}|, x \in \mathcal{S}(t)) \quad (5)$$

Additionally an outlier ratio can be estimated by declaring all samples within an interval of 3σ around ρ as inliers (99.7 % confidence level) and all others as outliers.

The representation and interpretation of a dynamic sample set are thus separated. This allows use of the map by techniques with simple low-dimensional measurement models like the unimodal model applied here. In our application this model is used for localization, so the localization module need not worry about more complicated multimodal probability density functions. But full information about the distribution of the map data is retained in the sample set, where it is really needed to represent multiple hypotheses.

3.4 Representation using multiple timescales

The obvious question at this point is "which timescale to choose?" As with spatial filtering the answer is "it depends." For the above toy example it may be useful to maintain two estimates, a more long-term one and a more short-term one. Accordingly, we propose to maintain the dynamic map simultaneously for several timescale parameters to cover the whole spectrum of possible changes.

In this context the relationship of the timescale parameter to actual time should also be discussed. In the toy example the sensor is stationary and samples at regular intervals. It is therefore easy to relate update ratios to hours or days. For an arbitrarily moving robot the situation is different. It cannot be said how long it will stay in a room or whether it will return to the same room in the same day. To relate an update ratio to the absolute time, we must wait in the order of the timescale to know how many measurements have been recorded during that time. Only then can samples be picked from the measurements according to the update ratio. Accordingly in our system the large timescale maps are updated only after a run or once a day and are called *long-term memory maps*. There is also a *short-term memory* map that is updated after each sensor reading. This map is characterized by a short half-life, short enough that the assumption of regular sampling is valid as long as the robot stays within a certain area.

As discussed in the introduction, simultaneous tracking of different timescales is intended to tackle the stability-plasticity dilemma. Each timescale corresponds to a position on an imaginary stability-plasticity scale. Maintaining

several timescales simultaneously is thus a way to be everywhere on that scale at the same time, allowing a map both to preserve old patterns and to adapt to new patterns.

3.5 Simulation of the toy example

The behaviour of a sample-based dynamic map is demonstrated and compared to recency-weighted averaging in a simulation of the toy example.

In this simulation the distance to the wall is 2 meters and the distance to the cupboard 1 meter. Measurements are simulated assuming normal noise with a standard deviation of 10 cm. At time $t = 10$ the cupboard is placed in front of the wall and at time $t = 20$ it is removed. Between two time-steps 20 measurements are made. The sample set consists of 20 samples and is initialized using the measurements of the first time-step. The recency-weighted estimate is initialized by the mean of these first 20 measurements. The algorithm is tested using three different update ratios: $u = 0.75$, $u = 0.25$ and $u = 0.05$. The update of the sample set takes place after each time-step. The recency-weighted average is updated directly after each measurement; the step-size parameter α is determined to correspond to the respective update ratio.

Fig. 2 shows the results of both algorithms. Recency weighted averaging appears to work well for large update rates like $u = 0.75$ when old samples are forgotten rapidly. For smaller values of u it interpolates as expected towards the new value and introduces values that have never been measured. After the cupboard has been placed against the wall, the estimate using the smallest u (corresponding to a large timescale) is practically useless, since it represents neither of the two objects for the whole considered time.

The behaviour of the sample-based dynamic map mirrors our requests much better. The estimates for $u = 0.75$ and $u = 0.25$ switch almost during one time step from the wall to the cupboard and vice versa, where the values of u determine the delay for that switch. The long-term component is unaffected by the events, since the period during which the cupboard appeared was too short for it to be registered.

Note that the above toy example assumed that the position of the robot is exactly the same during updates to the map. In practice, in a real SLAM scenario the position of the robot will vary between visits to the same mapped location. Therefore it is necessary to project observations (laser scans) to the same local coordinate system before updates are carried out: this aspect is described in Section 4.1.

4 A complete system for long-term SLAM

This section describes the localization and mapping system developed using the above map representation. The learning system and the representation of the map is exactly as described in the toy example. Around it a localization and mapping system for a real robot equipped with laser range scanner and

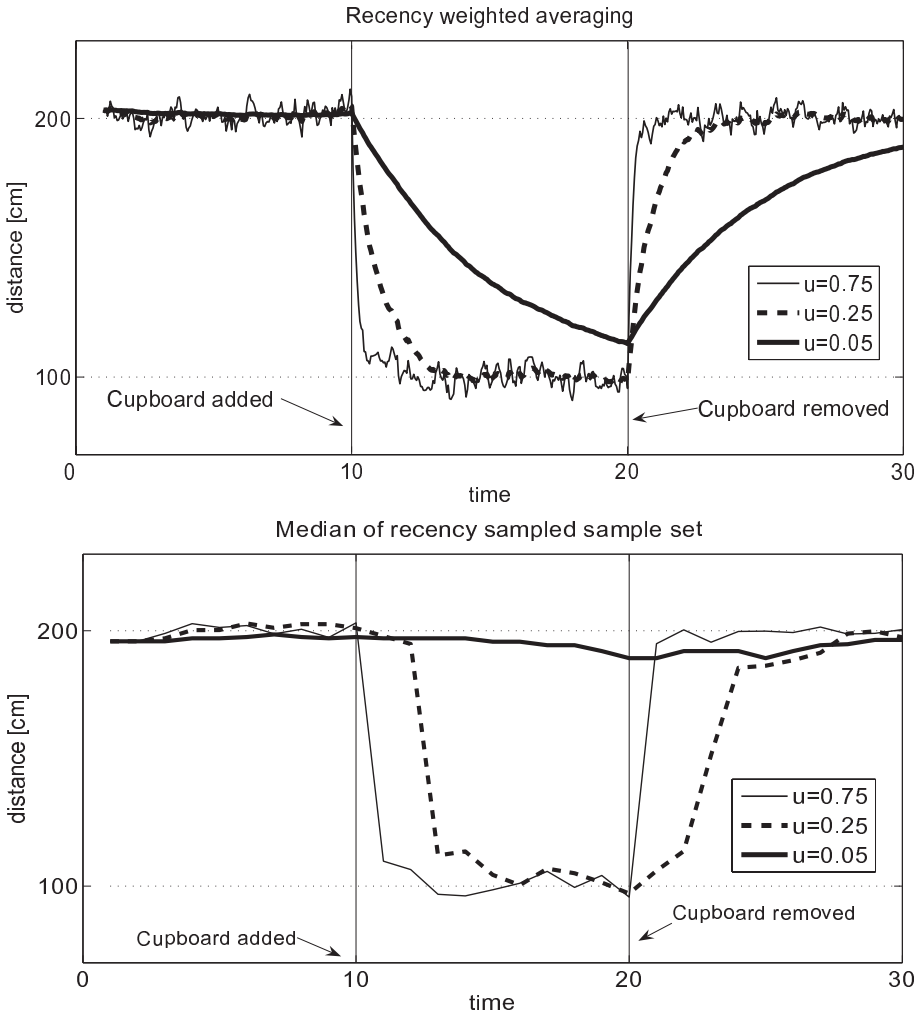

Figure 2: Using recency weighted averaging on toy example (top) and sample-based dynamic map (bottom) for three different timescales.

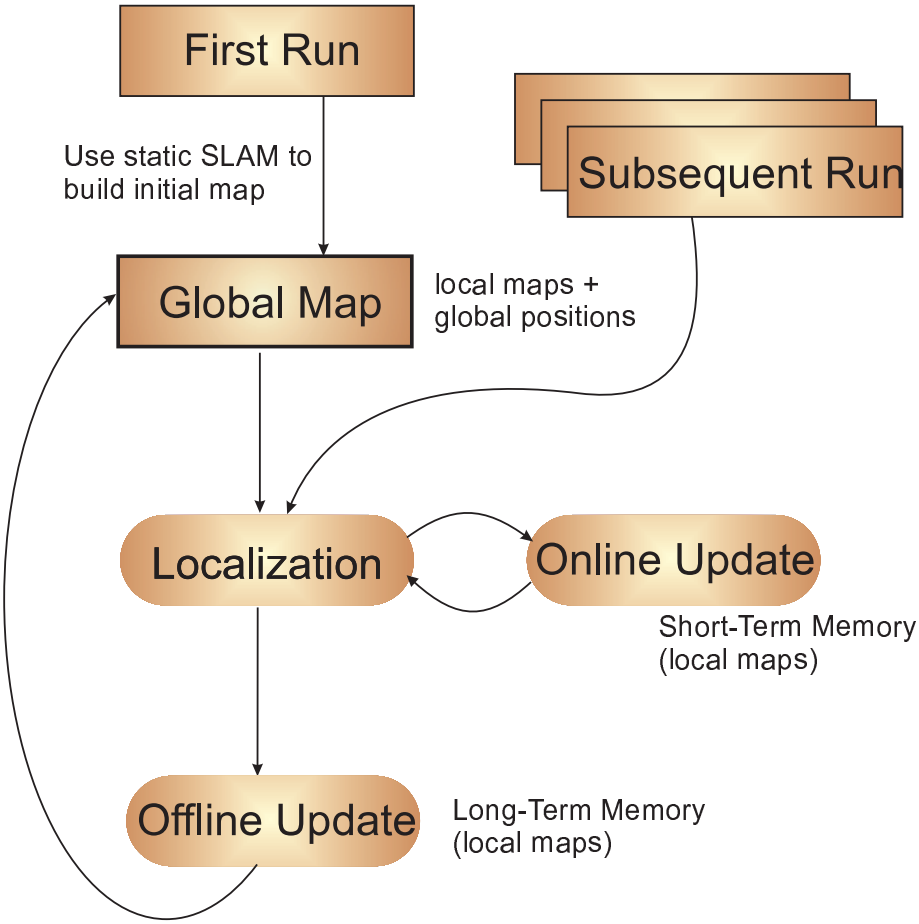

Figure 3: An overview of the whole localization and learning system. The dynamic map is both updated online during a run and offline after each run.

odometry has been built that has been shown to work robustly in extensive long-term experiments.

The data flow of the whole system is depicted in Fig. 3. First, an initial map is built from the first run through the new environment. This map is build by a classical static SLAM algorithm using laser scans and odometry as input and is based on scan matching [4]. The output is a set of selected laser scans with relations between them [12]. These laser scans form the initial set of *local maps*.

For localization the robot selects local maps near to its current position. One timescale within each local map is selected in a data-driven way: the timescale is selected that best explains the sensor data according to its learned perceptual model. The selected local maps are then converted into a point set called the *current map* and the current laser scan is then matched to this current map. Odometry information is used as a prior and as a bound to ensure robustness of matching.

After each localization step the *short-term maps* are updated online. The *long-term maps* are updated offline after each robot run or after each day. The update processes the data in a run based on the estimated robot trajectory. Pre-processing of the sensor data before map updating includes conversion of laser observations to the local coordinate systems of the nearby local maps. In the following subsections we provide technical details of the representation, local map selection, self-localization using scan matching, and learning.

4.1 Local maps

In our approach a local map is a generalization of a laser range scan and is linked to the global map by a position in global coordinates (see Fig. 4). A local map stores several sub-maps each corresponding to a different timescale. Each sub-map is like a 360 degree range scan from a constant position or reference point: it quantizes the continuous space of emanating rays from that position into a number of discrete bins. Finally each ray maintains a set of range values, corresponding to measured distances to objects, using the sample-based representation described in Section 3.

To use this representation in an actual SLAM implementation, it is necessary to project laser measurements from nearby positions (using the vehicle pose estimates from self-localization) to one position (i.e. the centre of the local map) and obtain all observations (rays) from there. An arbitrary 2-D point is mapped to a ray number and range value by finding the closest ray and taking the distance from the position of the local map to that point as the range value.

This definition allows the representation of a local map by a one-dimensional parameterization. Observations recorded near to a local map's position can be easily converted into this representation and the learning scheme introduced with the toy example can then be applied.

Table 1 shows the different timescales used in our experiments. The number of timescales and their properties were chosen according to the following considerations: short-term memory should react quickly to changes, so only a few data samples should be enough. Therefore the number of samples per ray should be

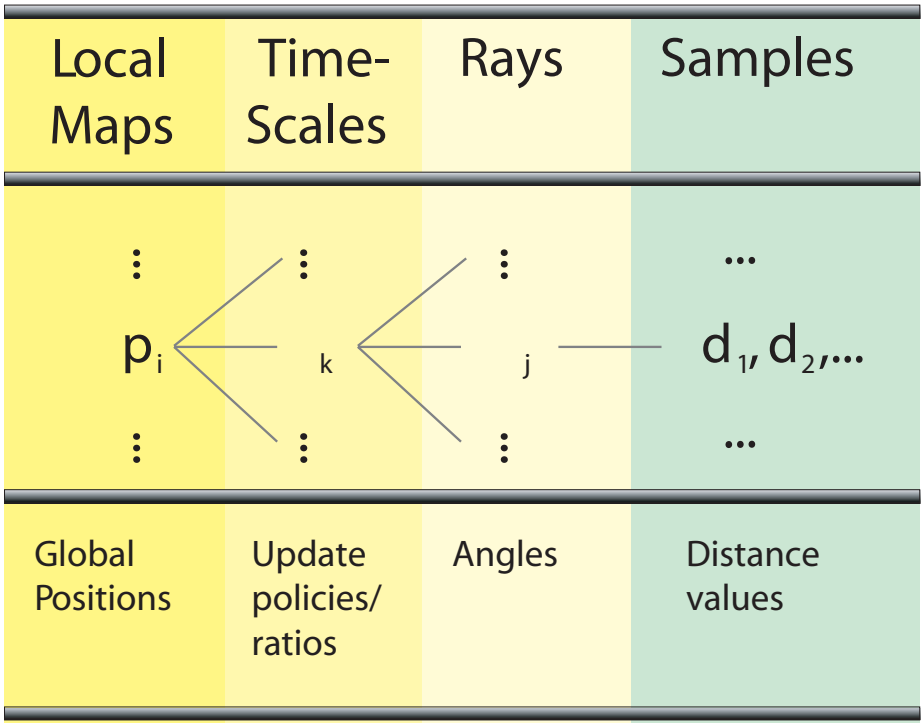

Figure 4: Internal map representation. The dynamic map consists of a set of local maps. Each local map maintains several sub-maps representing different timescale parameters. Each sub-map is represented by a set of samples for each angle.

small and the update ratio high. This, in turn, entails a relatively low accuracy. By contrast, long-term memory should not react to temporary variations, and adapt only if something has changed consistently. At the same time the static parts of the environment should be modeled with increased accuracy. This is achieved by decreasing the update ratio and increasing both the spatial resolution (number of rays per degree) and number of samples for the longer-term timescales. The use of robust statistics in combination with the large number of samples then provides both accuracy and robustness against outliers.

4.2 Perceptual model for local map selection

The sample sets are used to derive a *perceptual model* for the sensor input, that is to estimate the probability of a laser scan given a pose and a local map at a certain timescale (i.e., a set of samples). As outlined in Section 3.1 we derive a mixture model from the sample set. The probability of a range value measured at the position of a local map for any ray is estimated as

$$p(d) = (1 - p_{\text{outlier}})p_{\text{normal}}(d) + p_{\text{outlier}} * p_{\text{uniform}}(d), \tag{6}$$

where

$$p_{\text{normal}} \sim \mathcal{N}(\rho, \sigma^2) \tag{7}$$
$$p_{\text{uniform}} \sim \mathcal{U}(0, \text{maxRange}) \tag{8}$$

That is, p_{normal} is normally distributed and p_{uniform} uniformly distributed with parameters and mixture factor determined as in Section 3.3 from the samples of the ray considered (see Fig. 5). The log-likelihood of a whole scan is calculated by adding the logarithms of p for each range scan reading. To calculate the likelihood of a scan taken near the position of a local map the scan readings are transformed as if taken from that position.

4.3 Localization

The localization algorithm tracks the position of the robot over time. There is always only one single estimate for the robot's position and it is assumed that the starting position is known. The problem of global localization or robot kidnapping is not addressed here.

A single localization step consists of two main parts. A point set called the *current map* is synthesized by selecting those timescales that best fit the data according to the perceptual model and the current position estimate. The current map is then used to localize the robot at the next time step based on a scan matching scheme that incorporates odometry information [3]. After scan matching, a new current map is built using the resulting position estimate and so on. The central interaction between map and localization occurs here: the sensor data is used to select the most likely model of the current environment from the available timescales.

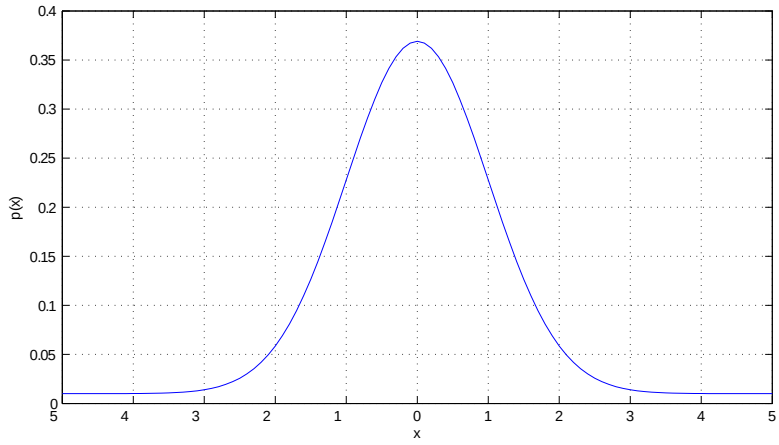

Figure 5: The perceptual model used to a interpret a sample set: a mixture of a normal distribution and an uniformly distributed outlier distribution (*Gaussian+Offset*). The model is determined by the mean ρ and variance σ of the Gaussian, the mixture weight (that is, the estimated outlier ratio) and a bounded domain (the compact support) for the outlier distribution. The resulting model is shown here for $\rho = 0, \sigma = 1$ and $[-5..5]$ as support. For the laser scanner the bounded domain is [0..maxRange] where maxRange is the maximum range of the laser scanner that is considered to be a valid measurement.

For a more formal description the following notation is used in the remainder: $\hat{x}_t$ is the posterior estimate of the robot's pose (x, y, ϕ) and $\hat{x}_t^-$ is the prior estimate at time t. The laser measurements at time step t are denoted by z_t.

The current map for time $t+1$ is built **after** the posterior estimate $\hat{x}_t$ has been calculated, using the following steps:

1. Local maps within a pre-specified Euclidean distance (4 meters in our experiments) of the robot's pose $\hat{x}_t$ are determined to form the set of *nearby local maps*.

2. For all local maps in the set of nearby local maps and for all timescales: calculate log-likelihood according to the perceptual model for range scan z_t.

3. Select the local map timescale with best log-likelihood and add it to the set of *current local maps*. The chosen timescale is marked as selected in that local map and the local map is removed from the set of nearby local maps.

4. Remove those range readings from z_t that have a high likelihood according to the selected model. A threshold is used here.

5. Go to 2 until the number of range readings in z_t is larger than a pre-specified size (20 in our experiments) and the set of nearby local maps is not empty.

6. Transform the selected local maps into a set of points, using the median as a range value. Only readings with an estimated variance no larger than a threshold (5 cm in our experiments) are converted to points.

If after this process more than half of the range scan readings have a low likelihood (same threshold as in step 4), the system switches into a safe mode, where all nearby local maps with the most long-term timescale are used to derive the current map. If this happens the localization algorithm may have lost its position or the environment may have changed too much to be represented accurately by any of the timescales. Using long-term maps in such situations is a safe fallback as they will be affected much less by temporary localization errors than the short-term memory maps (because they have the greater plasticity). The choice of long-term maps in such situations can also be motivated by the fact that they have lower learning rates. Since higher learning values tend to result in faster learning but greater later variability and thus lower long-term performance, small learning rates promise the best long-term performance [14]. In our experiments this mode was activated in approximately 0.5 percent of all cases.

The actual localization step then uses this current map in an approach that fuses scan matching and odometry. After the current map is built the prior estimate $\hat{x}_{t+1}^-$ is obtained by projecting forward the old posterior estimate $\hat{x}_t$ according to the odometry. The uncertainty of the odometry prediction is given

by a covariance matrix that was acquired experimentally in this work. Scan matching is then performed by minimizing an energy function E_S as described in [4]. The a priori estimate $\hat{x}^-_{t+1}$ and its estimated uncertainty (given by the covariance matrix) is incorporated as a prior E_O. This odometry prior is defined as $E_O(x) = (x - \hat{x}^-_{t+1})^t C_O^{-1} (x - \hat{x}^-_{t+1})$, where C_O is the covariance matrix of the odometry estimate. So finally a term

$$E_S + \kappa * E_O \tag{9}$$

is minimized.

Instead of determining a constant value for κ we employ an adaptive scheme for the following reason: In most cases the result from scan matching is so much more exact than odometry that incorporation of the prior E_O would only worsen the result, as is confirmed by the relation between the covariance matrices of scan matching and odometry. But, scan matching could also converge to a wrong solution (for example, it could match two doors with different opening angles instead of matching the walls if the robot is very close to the door). Such a case would also result in a well-conditioned covariance matrix. So the uncertainty of the scan matching result is not described well by the unimodal Gaussian associated with the covariance matrix for such cases. In conclusion it can be said that the scan match estimate is very accurate but can converge to the wrong solution. On the other hand the odometry is not very accurate but the description of its uncertainty by a covariance matrix models the reality well over the short distances considered here.

The sensor fusion scheme we use therefore forces the scan matching algorithm to converge to a solution within a region given by the uncertainty of the odometry as follows:

1. Set $\kappa = 0$.
2. Minimize $E_S + \kappa * E_O$.
3. If result within two standard deviations according to odometry then finish.
4. $\kappa = \kappa * 2$ and goto 2.

This scheme is reminiscent of the Levenberg-Marquardt algorithm (e.g. [7]) and was indeed inspired by it.

4.4 Learning: online and offline updates

Again, Table 1 shows the different timescales used in our experiments. The short-term memory map (λ_1) is updated after each localization step. All local maps whose centres are near (< 2.5 m) to the current position are considered for update. The range-finder readings are converted to polar coordinates, as described in Section 4.1, and then used to update the sample sets of the local maps, as described in Section 3.1. The robust estimates for the perceptual model parameters are then updated online. If there is no local map within 2.5 m of the

	Update ratio (u) /interval	Timescale ($t_{1/2}$) ($t_{1/2}$)	n_{Rays}	n_{Samples}
λ_1	0.2 / always	≈ 3.1	360	5
λ_2	0.8 / per run	≈ 0.43 runs	360	10
λ_3	0.8 / daily	≈ 0.43 days	720	50
λ_4	0.2 / daily	≈ 3.1 days	1440	100
λ_5	0.05 / daily	≈ 13.5 days	1440	100

Table 1: The different timescales of the sub-maps contained in one local map. λ_1 is the short-term memory map, λ_2-λ_5 are the long-term memory maps. The half-life for λ_1 is given in terms of the sensor's scanning frequency, that is 3.1 is the time needed to record 3.1 laser scans.

current position estimate, then a new local map is added at that position and initialized using the values of the current scan. For the long-term memory maps (λ_2-λ_5) the information (triples of local map, range scan and pose estimate) is stored and evaluated offline after a run (λ_2) or after a day (λ_3-λ_5), as indicated in the table, but with exactly the same method otherwise.

5 Experimental evaluation

5.1 Experimental Setup

The complete map learning system was tested extensively in an indoor environment consisting of a robotics laboratory with three rooms, a corridor with Ph.D. students' offices and a hallway containing stairs, chairs and tables. Over a period of five weeks the robot was steered manually through this environment from a constant start position. Typically three runs per day were performed; one in the morning, one after lunch and one in the early evening. A SICK LMS 200 laser scanner was used, and a total of around 100000 laser scans together with odometry data were recorded in 75 runs with a total distance of 9.6 km. The environment was not prepared in any way nor were people instructed somehow (that would have been impossible due to the heavy traffic of students in the hallway especially around lunchtime). The initial map built by the static SLAM algorithm comprising 76 local maps is shown in Fig. 6. Over 90 local maps were used in total, since the number of local map grows with time as new areas are visited, as described in Section 4.4. Referring to Table 1, each local map in our chosen representation needs to store 329 400 samples, meaning a total memory requirement of around 0.63 MB per local map if two bytes are used per sample, or approximately 60 MB in total for our setup.

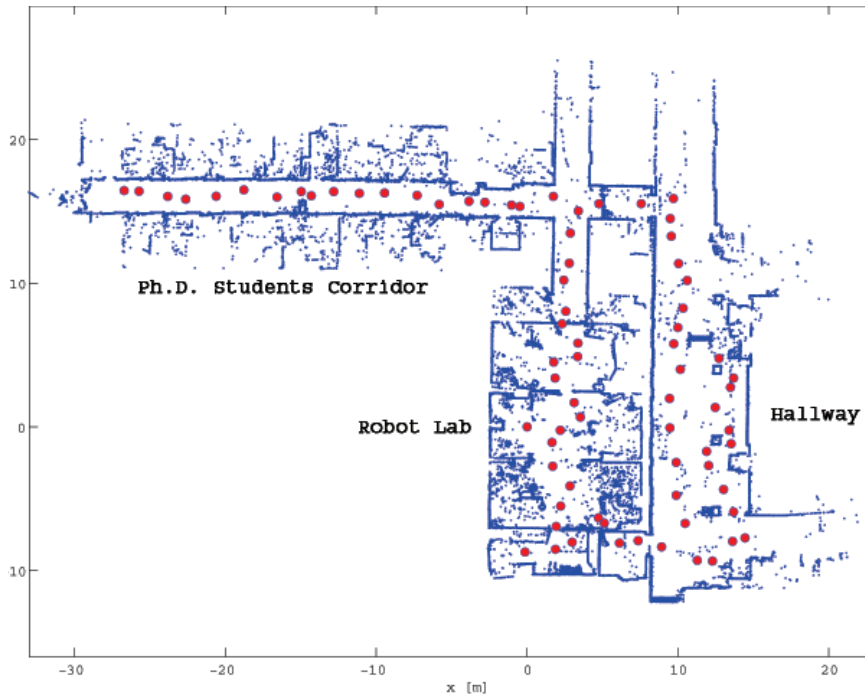

Figure 6: The initial map of the environment (obtained by a static SLAM approach) in which the experiment was conducted. The filled circles mark the positions of local maps.

5.2 Qualitative results

The most important result is that the dynamic map was stable over time and did not diverge. The accuracy of its local maps increased over time; this could be verified visually, for example, by looking at the straightness of walls. In parts of the environment where changes often occur, static parts like walls emerge while moving objects like chairs that could be observed in the initial map disappear. This could be observed, for example, in the robot lab. Figure 7 shows the most long term map (λ_5) of the middle room of the lab on three different days (Oct 18, Nov 1 and Nov 19) along with the local map that is updated after each run (λ_2) on Nov 19. It can be seen that the static aspects improve, although on Nov 19, for example, the lab looks quite different in some parts, as can be seen on the rightmost map. For visualization only points are shown for which the probabilistic model yields a standard deviation estimate smaller than 10 cm.

Major structural changes happened rarely as one might expect in this kind of environment, but one such change and the reaction of the dynamic map is shown in Figs. 8 and 9. Another structural change was the installation of new radiators in the hallway where some tables were also moved. Many changes on a smaller timescale (a few days or less) occurred frequently in the robotics lab, where movable "walls" and other robots often appeared at different positions as other researchers performed their experiments. Other frequent changes occurred in the hallway, e.g. chairs were often moved. These challenges were handled well by the dynamic map, with the short-term map adapting quickly to the changes.

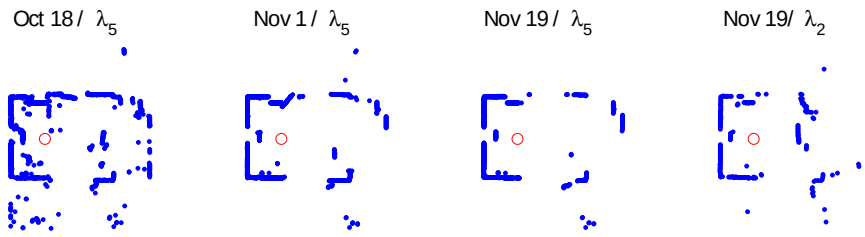

Figure 7: The most long-term submaps (λ_5 in Table 1) of an example local map (the middle room of the robot lab). The circle marks the centre of the local map. It can be seen that static aspects improve over time, although the environment sometimes looks quite different, e.g. on the last day as can be seen in the rightmost submap (which is a local map with a small half time, λ_2 in Table 1)

5.3 Quantitative performance measures

While these results may be satisfying enough from a theoretical point of view, a practitioner may still doubt whether such a technology is really needed, as

Figure 8: A major change occurred on day 4 of the experiments. The design class attendees presented their work (designs for toasters) in a small exhibition.

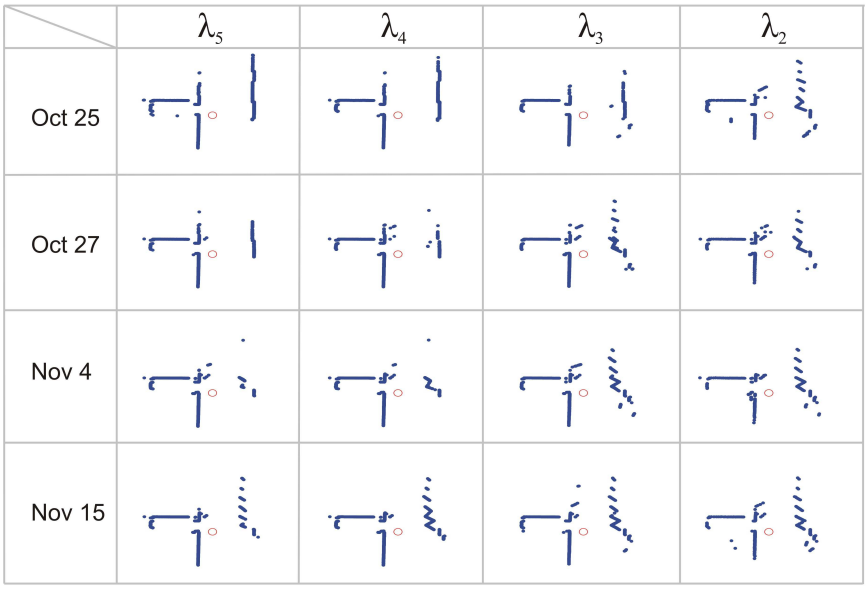

Figure 9: Evolution of a local map after a major change has occurred (the toaster exhibition). Shown are the long-term memory maps λ_2-λ_5 on four different days.

it makes the already difficult problem of simultaneous localization and mapping even more complex and might lead to a less robust and slower solution. Therefore we conducted an experimental comparison of the localization algorithm using the dynamic map against the same localization algorithm using the map created at the end of the first day as a static map. Additionally this static map was tested in two variations: with and without short-term memory (λ_1 in Table 1 activated). A general result is that in all cases there was no serious localization error that the robot could not recover from, so global localization was never required (the start position was always the same). In the absence of ground truth data, the following performance measures were selected:

1. The average likelihood of a range scan reading given the probabilistic model explained in Section 4.2. This measure gives an indication of how expected a scan is.

2. The smallest eigenvalue of the inverse of the covariance matrix that results from scan matching. This measure gives an indication of localization accuracy: if that value is large the corresponding uncertainty is small.

Figs. 10 and 11 show the evolution of these measures. In both cases there is a clear benefit in using the dynamic map. Also, using the short-term memory map alone improves the performance. But the long-term memory map improves the results even more, especially regarding localization accuracy. Both figures also show an expected effect: the static map performs better at the beginning than at the end, while the dynamic map improves performance with time.

5.4 Usage of the different timescales

While the above "black-box" measures characterize the performance of the whole system, they do not show how the internal components of the dynamic map behaved. The exact number of timescales we used (5) was a more or less arbitrary decision, and more timescales would perhaps further improve the overall performance. But the main idea was to cover the full spectrum of possible timescales and the order of the longest timescale ($\lambda_2 \approx 13.5$ days) relates to the duration of the experiment, so for an experiment lasting a year we would probably choose a maximum timescale with a half-life of two or three months.

To determine whether all of the timescales applied were really useful, we recorded how often each timescale was selected by the localization algorithm, with the result shown in Fig. 12. Clearly each timescale was actually used, and the relative frequencies are distributed relatively evenly among the long-term memory maps, but with noticably higher usage of the short-term component. A definite temporal trend was also observed: Fig. 13 shows relative usage frequencies for the short-term memory map (λ_1) and the most long-term memory map (λ_5) against time. The usage of the short-term memory map decreased with time, while usage of the long-term memory map increased. An explanation for this behaviour is that the long-term map models the static parts of the scene with increasing reliability over time. With more measurements it becomes

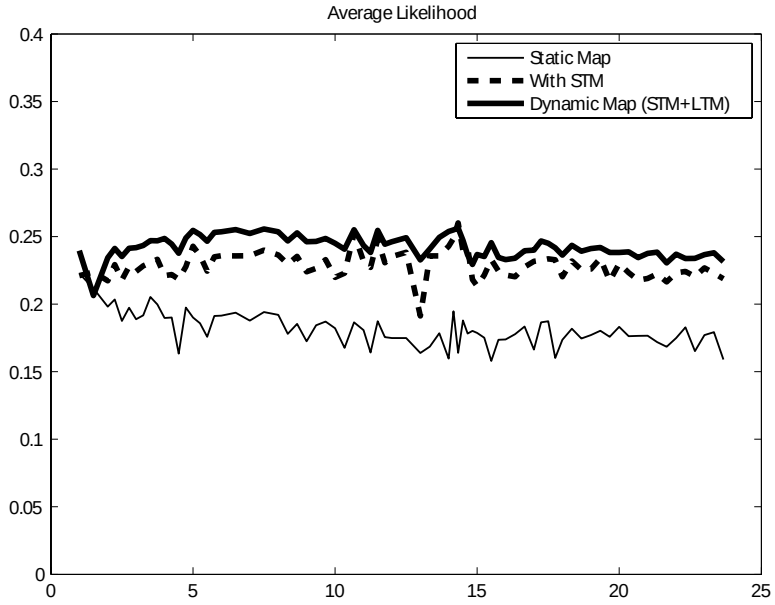

Figure 10: The average likelihood of a measured range value according to the learned perceptual model (Section 4.2). The dynamic map, consisting of a short-term memory map (STM) and long-term memory maps (LTM), is compared to a static map and a static map with added short-term memory. The static map is a snapshot of the dynamic map after the first day.

increasingly unlikely that outliers (e.g. caused by moving people or temporary localization errors) will be used in the local maps. In addition, the median filter becomes more precise as the number of samples in the long-term memory map increases, which also means that the corresponding standard deviation will typically become smaller as the estimate becomes more certain. The localization algorithm will then prefer the long-term memory map if the corresponding part of the environment has really remained static, due to its higher likelihood according to the perceptual model. This result provides compelling evidence that the dynamic map was not only successful in reacting to changes in the environment, but also that it was successful in improving map quality for the static parts of the environment.

6 Conclusion

This paper presented an experimental analysis of sample-based maps for long-term SLAM in dynamic environments. The approach is based on two novel, sim-

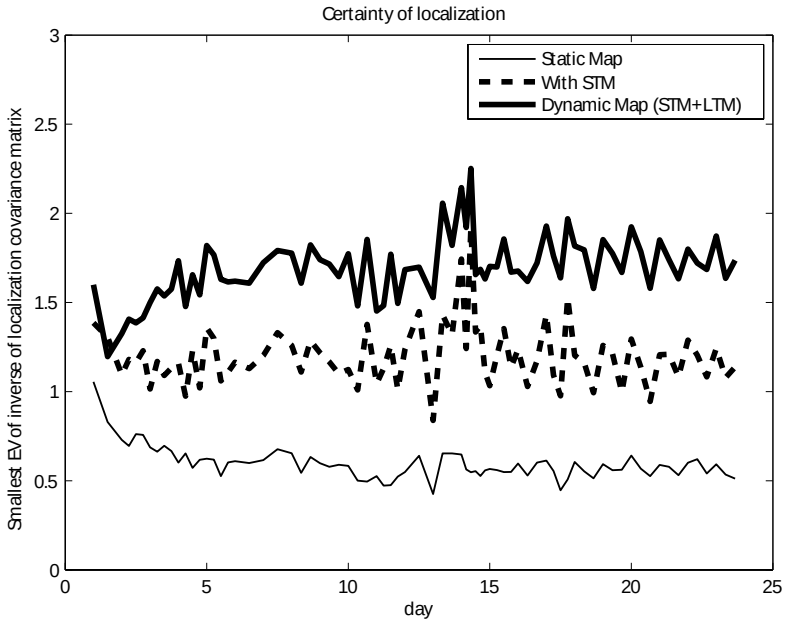

Figure 11: The certainty of the localization estimate. This certainty is measured by the value of the smallest eigenvalue of the inverse of the localization covariance matrix. If that value is large the corresponding uncertainty is small.

ple and powerful ideas: (1) representing the environment at different timescales, with older memories fading at different rates, and (2) using samples and robust statistics to handle contradicting measurements produced by environmental changes. Large amounts of memory are required for the proposed representation, but are available today on standard computers. A further contribution was made at a more abstract level: we investigated the general problems of life-long map learning in dynamic environments and identified the stability-plasticity dilemma as the most important problem. Our solution to the dilemma is to track the state of the world at several timescales simultaneously, and then to let the sensor data select the most appropriate timescale for a given situation. With this approach, the robot can simultaneously represent the world before, during and after changes to the configuration of an environment.

These concepts were verified through a long-term experiment over a period of 5 weeks (one of longest robotic mapping experiments performed), where changes to the environment included the installation and deinstallation of a small exhibition inside the mapped area. Our proposed method separates the well-known problems of static SLAM (error-backpropagation, loop closing) from the dynamic problems, handling the first by a classical SLAM algorithm. This is advantageous, because the difficulty of both parts adds but does not multiply.

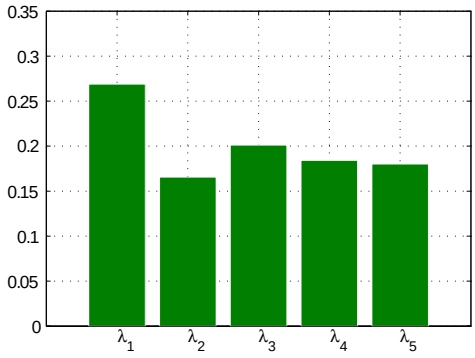

Figure 12: The relative frequency with which each submap was selected. Thereby λ_1 is the short-term memory and λ_2-λ_5 are the long-term memory maps (with update ratios as given in Table 1).

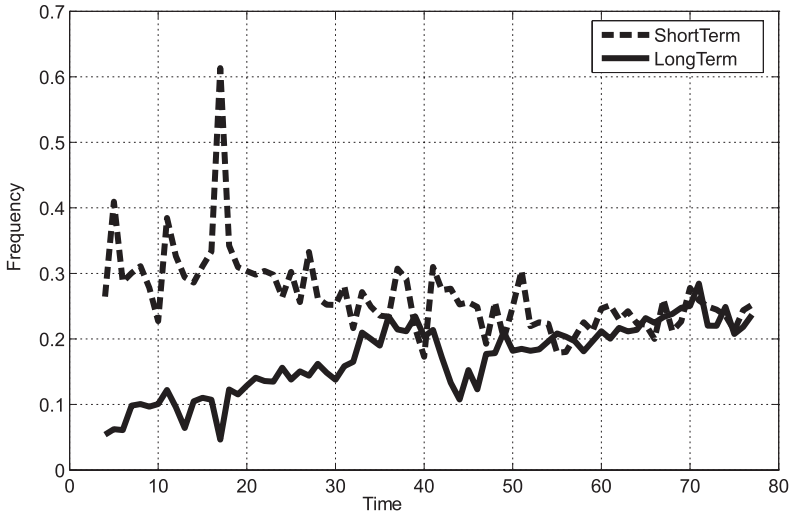

Figure 13: Evolution of usage frequencies for timescales λ_1 and λ_5.

A key idea to make this possible is to represent the map as a collection of "360 degree scans" relative to global reference points. Thereby the classical SLAM problems such as error propagation and loop closing affect only the reference points and are solved by a classical SLAM algorithm. The dynamical effects affect the individual rays of the 360 degree scans only and are solved by applying 1-D robust statistics (median, median of absolute differences) to each ray. This is crucial, because robust statistics are much harder to generalize to multiple dimensions than least square statistics.

In this paper, we did not consider the problems of "kidnapping" or global localization. Although serious localization errors were never observed in our experiments, this does not guarantee that such errors would never occur in general. In the case of small localization errors or temporary loss of position, wrong samples would be added to the dynamic map, but would then be treated as outliers. In the case of non-recoverable localization error, it would be necessary to switch to global localization and switch off map learning until the robot's position had been recovered. In our system, this could be detected using the perceptual model together with a suitable likelihood threshold to declare possible localization failures.

Future work would also include investigation of other sensor modalities such as vision instead of range-finder sensors. Rather than using a normal distribution in the perceptual model for each angle-bin and timescale, as in this work, a multi-modal distribution estimated from all timescales per bin might remove the need for search over timescales in the localization process. Our experiment in this work covered a five week period in a real environment: further work would still be needed to determine how to scale the approach to operation of service robots for a potentially undetermined period of time, e.g. many years. This would include further analysis on how to choose the timescales and their memory requirements, update ratios, etc. The problem of selecting a minimal set of local maps also remains an open topic for future research. In conclusion, while this work demonstrates a basic solution to the problem of long-term SLAM, it also opens up many interesting avenues for future work on lifelong operation of mobile service robots.

References

[1] J. Andrade-Cetto and A. Sanfeliu, "Concurrent map building and localization in indoor dynamic environments," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 16, no. 3, pp. 361–374, 2002.

[2] D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun, "Learning hierachical objects maps of non-stationary environments with mobile robots," in *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)*, 2002.

[3] P. Biber, "Map building and localization for long-term operation of mobile robots in dynamic environments," Ph.D. dissertation, Wilhelm-Schickard-Institut, University of Tübingen, Germany, 2007.

[4] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[5] P. Biber and T. Duckett, "Dynamic maps for long-term operation of mobile service robots," in *Proceedings of Robotics: Science and Systems I*, Cambridge, MA, USA, June 8-11 2005.

[6] W. Burgard, A. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot," *Artificial Intelligence*, vol. 114, no. 1-2, pp. 3–55, 1999.

[7] J. Dennis and R. B. Schnabel, "Numerical Methods for Unconstrained Optimization and Nonlinear Equations. SIAM Classics in Applied Mathematics," 1996.

[8] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley, Second Edition 2001.

[9] S. Grossberg, *The Adaptive Brain*. North Holland, 1988.

[10] D. Hähnel, R. Triebbel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *ICRA*, 2003.

[11] P. J. Huber, *Robust Statistics*. New York: Wiley, 1981.

[12] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, vol. 4, pp. 333–349, 1997.

[13] C. Stachniss and W. Burgard, "Mobile robot mapping and localization in non-static environments," in *Proc. of the National Conference on Artificial Intelligence (AAAI)"*, Pittsburgh, PA, USA, 2005.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[15] S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "MINERVA: A second generation mobile tour-guide robot," in *ICRA*, 1999.

[16] C.-W. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking if moving objects: Theory and results from a ground vehicle in crowded urban areas," in *ICRA*, 2003.

[17] B. Yamauchi and R. Beer, "Spatial learning for navigation in dynamic environments," *IEEE Transactions on Systems, Man and Cybernetics, Special Issue of Learning Autonomous Robots*, vol. 26, no. 3, pp. 496–505, 1996.

[18] U. Zimmer, "Adaptive approaches to basic mobile robot tasks," Ph.D. dissertation, University of Kaiserslautern, 1995.