Dynamic Maps for Long-Term Operation of Mobile Service Robots

Peter Biber
Graphical-Interactive Systems
Wilhelm Schickard Institute for Computer Science
University of Tübingen, Germany
Email: biber@gris.uni-tuebingen.de

Tom Duckett
AASS Research Center
Department of Technology
Örebro University, Sweden
Email:tom.duckett@tech.oru.se

***Abstract*— This paper introduces a dynamic map for mobile robots that adapts continuously over time. It resolves the stability-plasticity dilemma (the trade-off between adaptation to new patterns and preservation of old patterns) by representing the environment over multiple timescales simultaneously (5 in our experiments). A sample-based representation is proposed, where older memories fade at different rates depending on the timescale. Robust statistics are used to interpret the samples. It is shown that this approach can track both stationary and non-stationary elements of the environment, covering the full spectrum of variations from moving objects to structural changes. The method was evaluated in a five week experiment in a real dynamic environment. Experimental results show that the resulting map is stable, improves its quality over time and adapts to changes.**

I. Introduction

Future service robots will be required to run autonomously over really long periods of time in environments that change over time. Examples include security robots, robotic care-givers, tour guides, etc. These robots will be required to live together with people, and to adapt to the changes that people make to the world, including transient variations at different timescales (e.g., moving people, objects left temporarily, re-arranged furniture, etc.) and long-term modifications to the infrastructure of buildings. If a robot is to adapt to such changes then life-long learning is essential.

The challenge for lifelong SLAM (simultaneous localization and mapping) is that environments can change at different rates, and changes can be gradual or abrupt. Moving people follow a continuous path, while structural changes can occur when the robot is located elsewhere. Changes may not be permanent: an object may have been moved, a package may have been left in the corridor for a while, etc. It is therefore desirable for the robot to remember the old state too in case the change is only temporary. This challenge is related to a more general and well-known problem confronting every life-long learning system, namely the *stability-plasticity dilemma* [5]. Life-long learning demands both *adaption* to new patterns and *preservation* of old patterns at the same time. The escape from this dilemma proposed here is to learn a representation of the world at several timescales simultaneously, in order to cover the full spectrum of possible changes.

A localization and mapping system is presented that maintains a *dynamic map*, which adapts continuously over time. It uses a sample-based representation to handle changes at different timescales. The basic idea is to replenish samples stored for each timescale with new sensor measurements at a timescale-specific learning rate. This representation has a well-defined semantics, derived in section IV. The sample sets are interpreted using robust statistics [7], and a probabilistic model is used to infer the likelihood of new measurements for each different timescale. During localization the robot compares its current sensor data to all timescales in the map and chooses the timescale that best fits the data. Consequently localization is more robust and the map does not go out of date.

We present a working mapping and localization system that uses these concepts, and demonstrate experimentally the usability of this system in a real unmodified and moderately large environment over an extended period of time. With this approach the robot can maintain multiple hypotheses in time about the state of the world. For example, our dynamic map can simultaneously represent the world before, during and after temporary objects are left in a particular place. Thus the concept of a map is extended from a purely spatial representation to a spatio-temporal representation that bears many similarities to human memory.

II. Previous 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 dynamic mapping can be grouped into three categories.

First, some approaches attempt to explicitly discriminate "dynamic" from "static" aspects [3], [10], [11], [6]. For example, the RHINO tour guide robot [3] 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. By contrast our work uses a soft scale of learning rates to handle changes at different timescales.

Second, several authors have investigated aging of the map on a single timescale. Zimmer [13] presented a system that dynamically learns and updates the topology of a map during runtime and and showed the ability of his model to

adapt to changes. Yamauchi and Beer [12] 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 with 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.

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. Some of the previous works use recency weighted averaging ([1], [12]). In the next sections we show that this approach alone is not sufficient to handle the kind of changes that occur in real dynamic environments, but also how it can be modified and extended to cover multiple timescales.

III. Issues in Representation for Continuous Map Learning

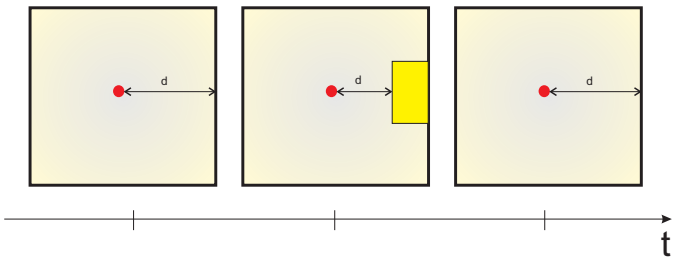

Fig. 1. A simple example environment where a dynamic map may be useful. There is only an empty room. After some time somebody puts a cupboard in front of one wall. Some time later the cupboard is removed. The example map consist only of the distance d.

To introduce, motivate and justify the representation proposed in this paper a simple toy example is considered. Let the example environment be an empty room and the "map" to be learned the distance from an arbitrary point of the room to one of the walls (see Fig. 1). If the environment were static then learning would be simple: all deviations from the true value would be due to Gaussian measurement noise and the map would be perfectly represented by the sample mean and the the sample variance. For stationary distributions these statistics are *sufficient* ([4], chapter 3.6), that is they store the information content of all measurements and it is therefore not necessary to keep them all in memory.

Assume now that the environment is dynamic. After some time somebody puts a cupboard in front of the wall. Obviously the above method is not well suited to this challenge. The time needed for the sample mean to come closer than an arbitrary threshold to the new true value is proportional to the time that the wall has been seen in the old position. This means in particular that if the wall has been seen for an infinite time the estimated distance will never change. This behavior is certainly not desired and the map should react to the appearance of the cupboard independently of how long the wall has been seen.

Request 1 The time that the map needs for adapting 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 ([9], 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 and is given by

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

assuming that measurements are made at regular intervals.

So the influence of old measurements decays according to a well known law that also governs many growing and decaying processes in nature. So it comes as no surprise that there is also a theory in psychology that explains the process of forgetting: decay theory states that forgetting occurs simply because of time passing.

This method fulfills Request 1. In exchange a parameter λ now governs the speed of adaption and leads to an obvious though interesting observation:

Observation 1 The law that governs the update of a dynamic map is inherently dependent on a timescale parameter.

Imagine that our 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 against outliers, and so we state that:

Request 2 The dynamic map should be robust against outliers.

In a dynamic environment this is a most natural requirement for improving a map over time. Consider that the distance to the wall has been measured a hundred times and then a moving person causes one false measurement. All of the learning effort would be in vain if that single measurement would degrade the quality of the estimate.

But after the considerations above it is clear that the inherent timescale determines whether a measurement should be called an outlier. Let us assume that recency weighted averaging would be robust against outliers. Despite that there would still be a complaint: the estimate for the distance would change gradually from the distance to the wall to the distance to the cupboard, but none of these in-between estimates would correspond to any physical reality. In contrast to many other dynamic processes, the changes that occur in the environments 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. These considerations lead to

Request 3 The dynamic map should only yield values that have been actually measured and it should not create interpolated values that can not be legitimized by the data.

The classical answer to requests 2 and 3 is to apply robust statistics. The median of a set of samples fulfills requests 2 (it ignores up to 50 percent of outliers) and 3. But there are no sufficient statistics to describe the median. Sufficient statistics preserve the information content of the whole data set and 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.

The use of a sample-based representation can also be justified by another (and less technical) observation concerning outliers that lies in the nature of the problem: actual changes in the environment appear in the first instance as outliers.

Observation 2 At the moment a measurement is made it is not possible to determine whether it is an outlier or not. Outlier detection is only possible post hoc and for a specific timescale.

Only after more time has passed (how much is dependent on the timescale parameter) and more measurements have been made can outliers be identified. So any method that tries to identify outliers directly after the measurement is in principal not suited to the problem of continuous learning of dynamic maps. It follows that the map representation must be able to track multiple hypotheses until it can be seen whether a change has really happened or only outliers were measured. So any method that represents the environment by a unimodal distribution cannot be considered an adequate solution.

In this section the basic problems and ideas that inspired the dynamic map were presented on a rather abstract level. The next section describes the sample-based representation of a dynamic map, its update rule and its properties, which we propose as a solution to the problems described above.

IV. Sample-based representation

A. 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.

B. 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 for s to be removed is given by 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:

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

This is a pleasing result. The age of the samples is distributed just like the weights in recency weighted averaging. The distribution is dependent on a timescale parameter $\lambda = -\ln(1-u)$. To get an impression of the meaning of λ we state the following well known facts:

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

C. 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:

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

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

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 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.

D. 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 must 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 announced in the introduction this 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.

E. Simulation of a toy example

The behavior 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;

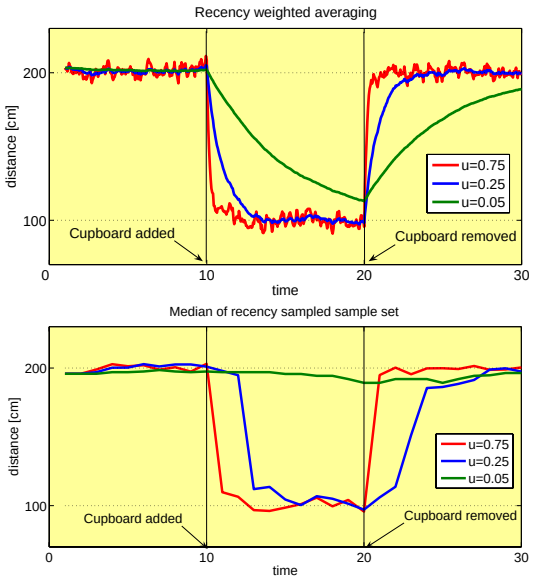

Fig. 2. Using recency weighted averaging on toy example (left) and sample-based dynamic map (right) for three different timescales.

the step-size parameter α is determined to correspond to the respective update ratio.

Figure 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 behavior 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.

V. A COMPLETE SYSTEM FOR CONTINUOUS LOCALIZATION AND MAP LEARNING

This section gives an overview of the localization and mapping system using the dynamic map. 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 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 static SLAM algorithm using laser scans and odometry as input and is based on scan matching. The output is a set of selected laser scans with

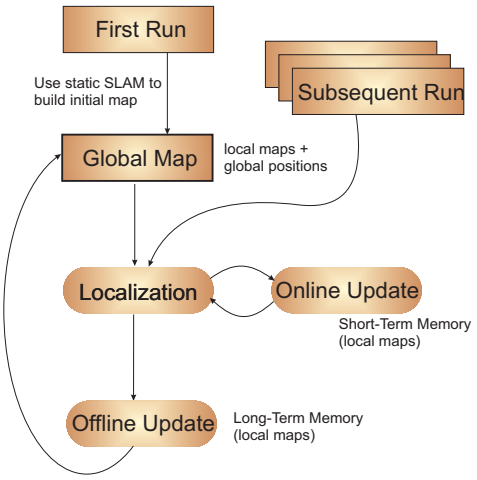

Fig. 3. An overview of the whole localization and learning system. The dynamic map is both updated online during a run and offline after each run.

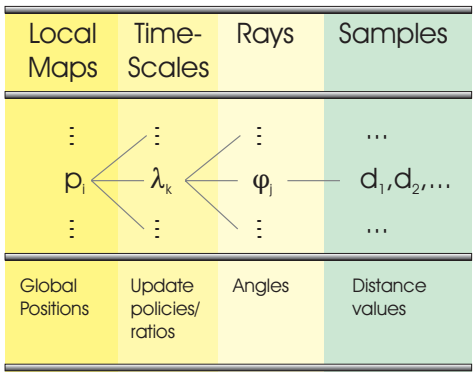

Fig. 4. Internal map representation. The dynamic map consists of a set of local maps. Each local map keeps several sub-maps representing different timescale parameters. Each sub-map in turn is represented by a set of samples for each angle.

relations between them [8]. These laser scans form the initial set of *local maps*.

A local map is like a 360 degree range scan from a constant position: it quantizes the continuous space of all angles of emanating rays from that position into a number of discrete bins. The distances to objects in the direction of these rays (range values) are the values tracked by the dynamic map. A local map is linked to the global map by the position of the projection center for these rays. The range values are represented as sets of samples, one set for each timescale parameter as introduced in the previous section.

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 collected during a run based on the estimated robot trajectory.

Technical details on local local map selection and localization using scan matching are omitted here due to lack of space, since these are both relatively straightforward. Instead we focus next on the representational issues.

A. Local Maps

In our implementation a local map is a generalization of a laser range scan and is linked to the global map by a position in global coordinates. It holds several sub-maps each corresponding to a different timescale. Each sub-map is parameterized by a number of rays emanating from its position, each ray corresponding to a different angle. Finally each ray maintains a set of distance values: this sample set is the basis of a dynamic map, see Fig. 4.

The emanating rays cover the whole 2D space. An arbitrary 2D point can be mapped to a ray number and range value by finding the closest ray and taking the distance to the position of the local map 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.

B. Perceptual model for local map selection

The sample sets set 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 IV-A 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 by:

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

where

$$p_{\text{normal}} \sim \mathcal{N}(\rho, \sigma^2) \quad (7)$$

$$p_{\text{uniform}} \sim \mathcal{U}(0, \text{maxRange}) \quad (8)$$

That is, p_{normal} is normally distributed and p_{uniform} uniformly distributed with parameters and mixture factor determined as in section IV-A from the samples of the ray considered. 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.

This model considers not only measurement noise but also possible measurements caused by outliers. Altogether, since its parameters are estimated by the median and the MAD operator, it can be considered a robust model whose parameters are estimated robustly.

C. 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 problems of global localization and robot kidnapping are not considered here.

A single localization step consists of two main parts. A *current map* is synthesized by selecting those timescales that best fit the data according to the introduced perceptual model and the current position estimate. This current map is then used to localize the robot in the next time step based on a scan matching scheme that incorporates odometry information. 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.

D. Learning: Online and Offline Update of the Map

As described in section IV-D there are long-term memory maps and a short-term memory map. The short-term memory map is updated after each localization step. All local maps whose centers are near (< 2.5 m) to the current position are informed of the new measurements. The readings of the range scans are converted to polar coordinates as in section V-A and then used to update the sample sets of the local maps as described in section IV-A. The robust estimates for the perceptual model parameters are then updated online. If there is no local map that is nearer than 2.5 m, a new one is initialized immediately, with a global position according to the current position estimate.

For the long-term memory maps the information (triples of local map, range scan and pose estimate) is stored and evaluated only after a run or after a day, but with exactly the same method otherwise.

The next section describes the experimental setup we have employed and details the different timescale and update intervals that were tested.

VI. Experimental evaluation

A. 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 and also 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 an approximate distance covered of 9.6 km. The environment was not prepared in any way nor were people instructed somehow (that would also 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 is shown in Fig. 5.

	Upd. ratio/interval	timescale ($t_{1/2}$)	n_{Rays}	n_{Samples}
λ_1	$u = 0.2$ / always	$t_{1/2} \approx 3.1$	360 (1 ray/°)	5
λ_2	$u = 0.8$ / per run	$t_{1/2} \approx 0.43$ runs	360 (1 ray/°)	10
λ_3	$u = 0.8$ / daily	$t_{1/2} \approx 0.43$ days	720 (2 rays/°)	50
λ_4	$u = 0.2$ / daily	$t_{1/2} \approx 3.1$ days	1440 (4 rays/°)	100
λ_5	$u = 0.05$ / daily	$t_{1/2} \approx 13.5$ days	1440 (4 rays/°)	100

TABLE I

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.

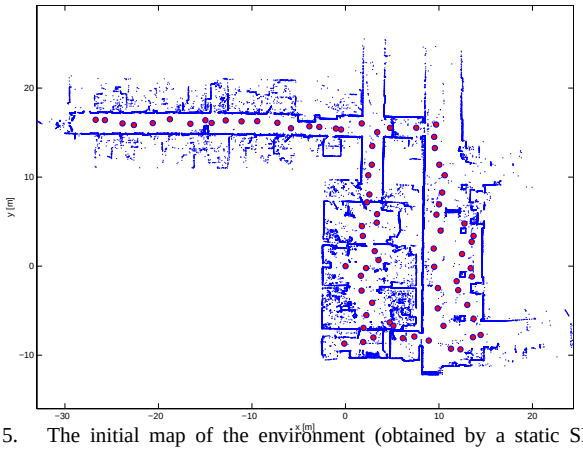

Fig. 5. The initial map of the environment (obtained by a static SLAM approach) in which the experiment was conducted. The filled circles mark the positions of local maps.

Table I shows the different timescales for the sub-maps contained in one local map. The number of sub-maps and their properties were chosen according to the following consideration: a short-term memory should react quickly to changes so only a few data samples should be enough to forget an old opinion. Therefore the number of samples per ray should be small and the update ratio high. This, in turn, entails a relatively low accuracy as all estimates are always calculated on the small amount of available data.

The opposite is true for a real long-term map: such a map should not react at all to temporarily changes, and adapt only if something has really changed consistently. At the same time the static parts of the environment should be modeled with increased accuracy. A large number of samples per ray and a low update ratio are able to provide these properties. The use of robust statistics in combination with the large number of samples then provides both accuracy and robustness against outliers.

The first sub-map is updated online after each new laser scan reading and can be regarded as the short-term memory map. The high update rate effectively always inserts the latest value into the sample set and the sample set is kept small in order to react quickly to changes. The other four sub-maps are then updated offline. With decreasing update ratios both the spatial resolution (number of rays per degree) and the number of samples increases.

B. Qualitative results

The most important result is: 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 6 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.

To assess these statements it is interesting to know what kind of changes actually happened in the environment. 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. 7 and 8. 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. Thus severe changes from one day to the next often posed a good challenge to the dynamic map. These challenges were handled well, and the short-term memory map in particular adapted quickly to the changes. Other frequent changes occurred in the hallway, e.g., chairs were often moved. This hallway was also the most busy place, with a lot of students passing through most of the time.

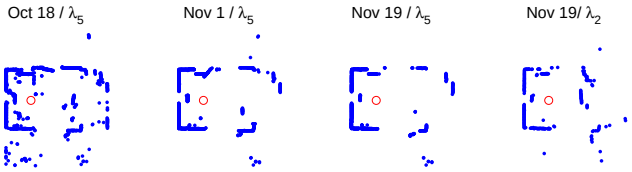

Fig. 6. The most long-term submaps (λ_5 in table I) of an example local map (the middle room of the robot lab). The red circle marks the center of the local map. It can be seen that static aspects improve over time, although the environment sometimes looks quite different, e.g., on the last day as can be seen on the rightmost submap (which is a local map with a small half time, λ_2 in table I)

C. Quantitative results

With the help of some examples is has been shown that the dynamic map successfully adapts to changes in the environment and improves its quality where the environment remains static. 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 it makes the already

Fig. 7. A major change occurred on day 4 of the experiments. The design class attendees presented their work (designs for toasters) in a small exhibition.

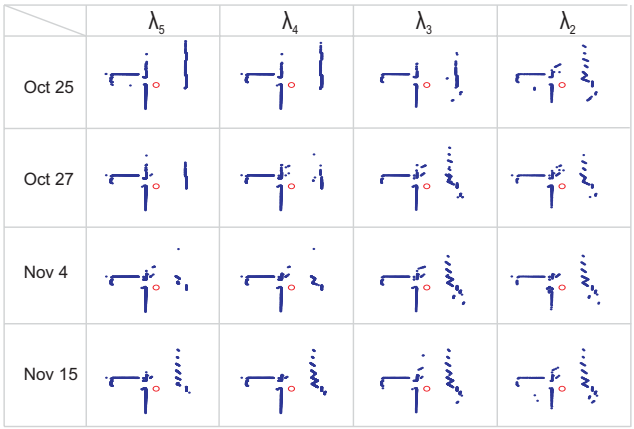

Fig. 8. Evolution of a local map after a major change has occurred (the toaster exhibition). Shown are the long-term memory maps λ_2-λ_5 on four different days.

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. So three maps were compared: a static map, a static map with short-term memory (λ_1 in table I activated), and the full dynamic map (λ_1–λ_5 in table I activated). Thus it can be determined how much the localization algorithm benefits from the short term memory map alone and what additional value is obtained from the long term memory maps. 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). This does not, of course, mean that the dynamic map is unnecessary, and the benefit was measured as follows. In the absence of ground truth data other indicators must be used for quantitative evaluation, so the following two performance measures were selected:

1) The average likelihood of a range scan reading given the probabilistic model explained in section V-B. This measure gives an indication of how expected a scan is.
2) The smallest eigenvalue of the covariance matrix that

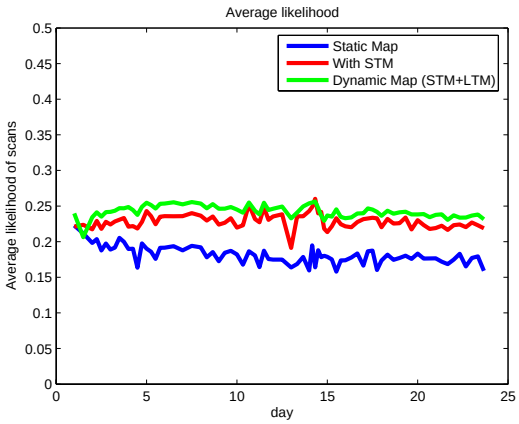

Fig. 9. The average likelihood of a measured range value according to the learned perceptual model (section V-B). The dynamic map, consisting of a short-term memory map (STM) and long-term memory maps (LTM), is compared to a static map and a static map with added short-term memory. The static map is a snapshot of the dynamic map after the first day.

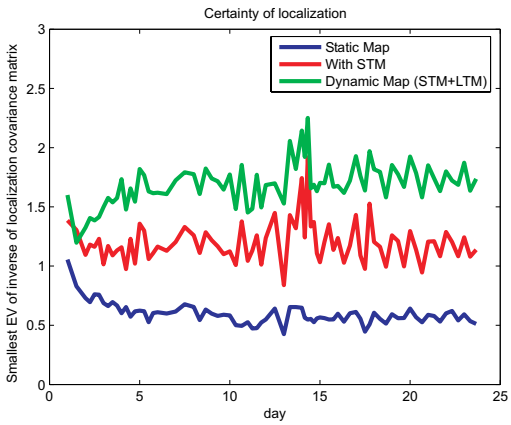

Fig. 10. The certainty of the localization estimate from scan matching. This certainty is measured by the value of the smallest eigenvalue of the inverse of the covariance matrix. If that value is large the corresponding uncertainty ellipse has a small area.

results from scan matching. This measure describes how the localization algorithm estimates the certainty of the result: a large value indicates a small uncertainty.

The second measure is of course specific to our scan matching algorithm, but it can be expected that other scan match algorithms would behave similarly. So this measure can be seen as an indication of localization accuracy. Figs. 9 and 10 show the temporal 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 of the experiments than at the end. With increasing time from the start of the experiment, both indicators show decreasing performance and after a few days the performance seems to more or less stabilize at a considerably lower level. By contrast, the dynamic map improves performance with time and then also stabilizes, but at a higher level.

VII. Conclusion

This paper presented the dynamic map, a way to handle the problem of lifelong map learning in a dynamic and ever-changing world. The key technical contribution is the use of a sample-based representation and its interpretation through robust statistics. Huge amounts of memory are required for the proposed representation, but such amounts 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. This solution is simple and effective, and it is also so general that it can be expected to find application in other areas where life-long learning is necessary. The relevance of such learning abilities in any real autonomous system with very long operation times should be obvious, both from an academic view and from the view of real applications. Here we tried to cover both theoretical and practical aspects of the problem, and also performed the necessary experiments to demonstrate the underlying concepts. This work may be an important step in a promising direction where much research remains to be done.

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*, 16(3):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] 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*, 114(1-2):3–55, 1999.

[4] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, Second Edition 2001.

[5] Stephen Grossberg. *The Adaptive Brain*. North Holland, 1988.

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

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

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

[9] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[10] S. Thrun, M. Bennewitz, W. Burgard, A.B. 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.

[11] 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.

[12] Brian Yamauchi and Randall Beer. Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man and Cybernetics, Special Issue of Learning Autonomous Robots*, 26(3):496–505, 1996.

[13] Uwe Zimmer. *Adaptive Approaches to Basic Mobile Robot Tasks*. PhD thesis, University of Kaiserslautern, 1995.